ACS Poster Collection (2018) 1–15 Published 8/2018; Paper 19

Ordering of Training Inputs for a Neural Network Learner

Xiaotian Wang XIAOTIAN.WANG@VANDERBILT.EDU
Xiaohan Wang XIAOHAN.WANG@VANDERBILT.EDU
Maithilee Kunda MKUNDA@VANDERBILT.EDU
Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN 37235 USA

Abstract

The stochastic gradient descent algorithm for training neural networks is widely used in many machine learning tasks. In typical practice, this algorithm operates by choosing a small fraction of the training data, called a mini-batch, at each iteration to compute an approximation of the gradient of the objective function to be optimized. Researchers tend to use small batch sizes, where each mini-batch is selected uniformly across the underlying distribution of categories present in the training data. While these choices of training procedures are common and often based on empirical observations, i.e., trial-and-error experiments to aid in hyperparameter selection, there are few formal investigations into the question of how the order and composition of training data would affect the training efficiency and generalizability of the neural network. To gain more insight into this problem, we have investigated effects of training order and the composition of a mini-batch by conducting a series of controlled experiments. In our experiments, we retrained an existing neural network model for object recognition with images from the ImageNet data set and from a newly-collected data set called the Toybox data set. Here, we present initial results from our experiments, and discuss implications for future work.

1. Introduction

How does an intelligent agent actively learn from its environment? In other words, if a learning agent can play an active role in selecting its own training data to learn from, what mechanisms might guide the selection process?

This question has been asked in AI at many times in many different forms, from work in distinguishing relevant features and examples from irrelevant ones (Blum & Langley, 1997) to selecting which examples from an unlabeled data set to label, i.e., active learning (Settles, 2012), for example.

A parallel line of research, curriculum learning, assumes the existence of a "teacher" who can provide the learner with certain orderings of training data. In one well known study of curriculum learning, researchers compared the speed of training a network for classifying geometric shapes with and without the curriculum learning technique. The result of the experiment showed that by feeding in the basic shapes first and then feeding in the more complex shapes, researchers could achieve a significantly higher training efficiency, compared to feeding in both kinds of shapes randomly

[redacted]

(regardless of the complexity) (Bengio et al., 2009). Some following experiments also illustrated the possibility of automating the curriculum learning process (with LSTM) (Graves et al., 2017).

Additional related work in AI and cognitive science includes research on information foraging (Pirolli & Card, 1999), ordering in clustering algorithms (Fisher, 1996), and other various types and levels of information ordering during learning (Langley, 1995; Nerb et al., 2007).

Here, we frame the problem of a learner selecting its own training data in the context of the stochastic gradient descent algorithm, currently used to train neural networks in many different task domains. In current practice, each iteration of the training process involves selecting a *mini-batch* of examples randomly from the training set, without replacement, and using error estimates summed over this mini-batch of examples to compute the gradient. The size of the mini-batch, one of many hyperparameters that must be specified by the system designer, is often determined empirically through trial-and-error. Many different mini-batch sizes might be evaluated to find the one that yields the best performance over some validation data set. Random and uniform selection of each mini-batch is the approach currently used in many neural network applications, including visual object recognition, which is the task domain we focus on in this paper (Krizhevsky et al., 2012).

However, if we compare neural network learning to human learning, for the task of visual object recognition, many differences in the "training process" emerge. Human infants learning to recognize objects do not receive uniform, random selections of all possible object categories to learn from at a time. Instead, infants receive coherent sequences of inputs covering just one or a few categories at a time, often through object play (Clerkin et al., 2017). Learning experiments conducted with people across a variety of ages likewise emphasize the importance of similarities and differences (and orderings thereof) within a sequence of training experiences (Carvalho & Goldstone, 2014).

Some computational experiments have shown convergence speed difference when using different methods for randomizing the training data in a mini-batch (Bottou, 2009). However, no experiments have been done (that we are aware of) to systematically investigate the effects of *not* using randomization for mini-batch selection in the stochastic gradient descent algorithm.

We envision the learning problem faced by active intelligent agents operating in real, unstructured environments as a cost-reward tradeoff. For most intelligent agents, like humans, that have limited attentional bandwidth, the choice of attending to any particular stimulus at any given time incurs a cost, in that other sources of information are *not* being attended to at that time. The reward comes in whatever can be learned from that stimulus at that time. Repeated exposure to the same stimulus may enhance learning of that stimulus, though comparisons to other stimuli are likely necessary to learn differentiating characteristics, like category boundaries. Do these principles play a role in human learning, e.g., have infant play behaviors evolved to produce "good" sequences of training inputs? And could these principles be used to design intelligent agents that self-select "good" sequences of training inputs that optimize long-term learning outcomes? These are the primary questions motivating our work.

Our research is not yet at the point where we have defined strategies that learners can use to optimize the learning value of the training inputs they select. However, our initial experiments presented here focus on exploring different distributions of training data, and observing what kinds of differences (if any) might emerge from the training process under these different input conditions.

In this paper, we design various compositions of mini-batches and examine how different compositions of mini-batches could affect the training efficiency for various data sets, where training efficiency is defined as a measure of network convergence and accuracy. In addition, we explore if the different compositions of the training data set, which would explicitly affect the composition of a mini-batch, would indirectly affect the training efficiency.

As mentioned above, we focus on stochastic gradient descent for neural networks, though similar experiments will be valuable to conduct using other learning frameworks as well. As found in many other studies on the ordering of learning experiences, in both AI and cognitive science, we expect that themes of similarity versus difference, and redundancy versus uniqueness, will emerge as important factors in how the learning algorithm captures incoming information and deploys it to deal with new, previously unseen experiences.

2. Experiments, Results and Analysis

2.1 Data Sets

We used two data sets for a series of experiments. One was a subset of the ImageNet data set, a large-scale hierarchical image database for artificial intelligence research (Deng et al., 2009). The other one was a newly collected data set called Toybox that has been created by our research team at Vanderbilt University. The Toybox data set, which contains a collection of first-person views, is used to learn how the human beings would observe objects for image recognition tasks. Researchers used wearable cameras to record videos in first-person views that contain different objects and gathered them into this new data set (Wang et al., 2017). With different frequencies of capturing frames from videos, researchers could get various numbers of images per object. Different from the images in the ImageNet data set, which contains various objects with utterly different backgrounds, the Toybox data set may contain many images of the same object in similar backgrounds. Both data sets contain 12 categories of objects. However, the Toybox data set contains only 30 objects per category, while the ImageNet data set we used contains 1000 objects per category.

2.2 Three Different Compositions of Mini-Batch

To test how compositions of mini-batches would affect the training efficiency, we designed three different modes to form a mini-batch. These three modes were selected purely as examples of modes we thought were likely to lead to different learning outcomes. There are many other interesting modes that we will explore in future work, including modes that more closely match the types of sequences experienced by human infants.

The first mode is called uniform mode. In each mini-batch, we will select the same number of images from each category, either sorting them based on particular criteria or keeping the order inside a mini-batch randomized, as shown in Figure 1 (left); a specific color denotes a specific category of objects. For example, if the size of a mini-batch is 100, and we have 10 categories of objects, then we will select 10 objects from each category for each mini-batch. To make sure that every single image in the training data set will be used in the training process for the same number

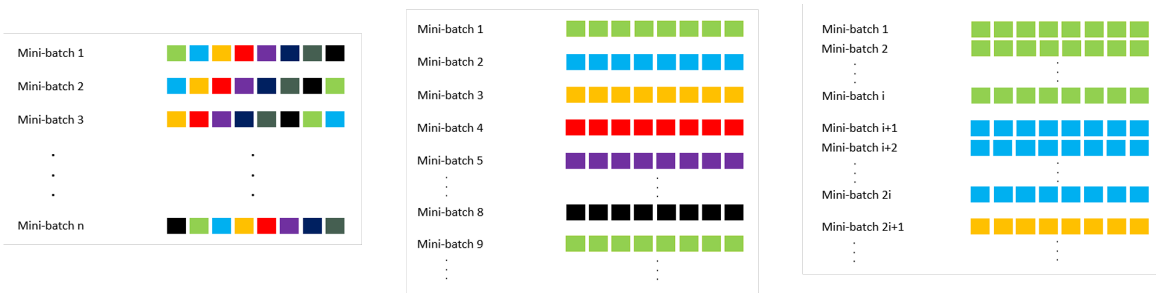

Figure 1. Illustration of three modes used to investigate different compositions of mini-batches: uniform mode (left), alternate mode (center), and sequential mode (right). Different colors are used to indicate training examples from different object categories (e.g., cup, spoon, truck, horse, etc.).

of times, we will at first randomize the order of images for each category, and sequentially select the same number of images from each category.

The second mode is called alternate mode. Images inside a mini-batch are of the same category. However, we will change to another category of objects in the subsequent mini-batch, as shown in Figure 1 (center). For example, if the size of a mini-batch is 100, then at the first step of training, we will feed in 100 images of category A; at the second step, we will feed in 100 images of category B, and so on. For this kind of composition, we also have to apply similar method mentioned above to make sure that every single image in the training data set will be used for the same number of times.

The third mode is called sequential mode. Images inside a mini-batch are of the same category, and we will sequentially pick images from a particular category before we start to use images from the subsequent categories, as shown in Figure 1 (right). For example, if the size of a mini-batch is 100, and each category contains 1000 images, we will keep using 100 different images of the category A at each step, until we have used all images from the category A (which will take 10 iterations). Then we will start to use images from category B, and so on. For the third kind of composition, we also have to make sure that every single image in the training data set will be used for the same number of times during the training process.

2.3 Effects of Mini-Batch Composition

In our experiments for exploring how the composition of a mini-batch would affect the training efficiency, we will train the Inception-v3 model on selected images from the Toybox data set, and evaluate on both selected images from the Toybox data set and ImageNet data set. Specifically, for the training data set, we will select 1080 images per category from the Toybox data set. Each category will contain 12 different objects, and each object will appear in 90 different images in the training data set. For each category, 3 objects, other than those 12 objects used for training, will be used for evaluating the network model. We will make sure that the number of images in the evaluation set to be 100 per category, and keep the number of images for each object in a category roughly the same (about 33 images per object).

For the training process, we will use a batch size of 45, a learning rate of 0.01, and 3600 training iterations in total, so that in every experiment to be completed, every single image in the training

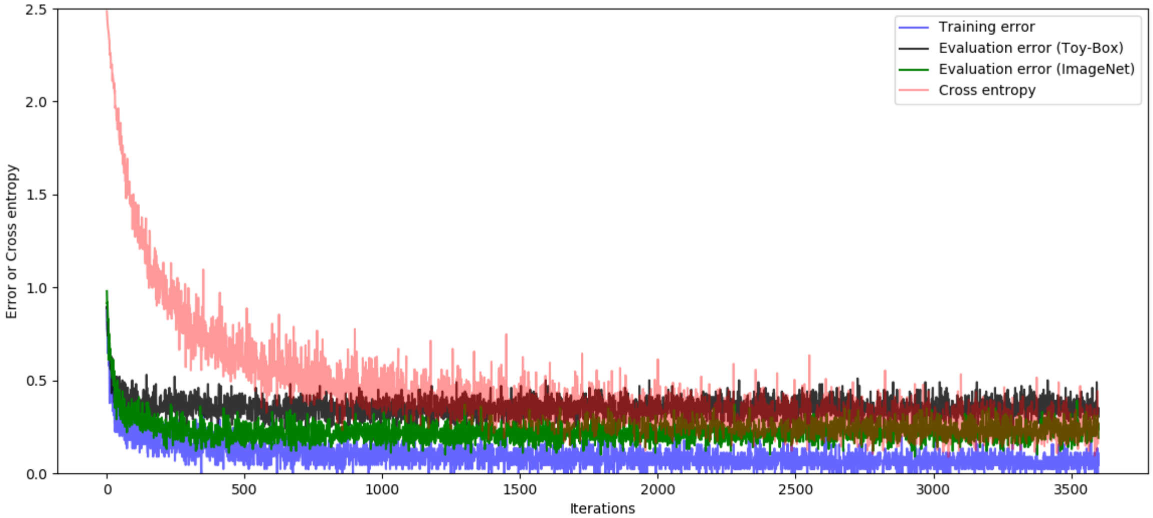

Figure 2. Measures of training and testing error using mini-batches selected using the uniform mode, across 3600 training iterations.

data set will be used for the same number of times. Besides, during each training step, we will calculate the training error right before we feed a mini-batch of data into the network model to train so that we could know how the current model would perform on the current batch of data before the model is overfitting after the current step of training.

We plot curves of the training error, evaluation error (from both the Toybox and ImageNet data set), and the cross-entropy for each mode of mini-batch composition in Figure 2, Figure 4 and Figure 6. For each mode, we also trained for another 240 iterations, so that we can get three extra plots to show more detailed trends of curves at the beginning of the training process, as shown in Figure 3, Figure 5 and Figure 7.

Figure 2 and Figure 3 show training process when we were using the uniform mode for the batch composition. We can see that the training error, evaluation error, and the cross-entropy drop quickly during the first few iterations, and then slowly decrease, until the evaluation error of both the Toybox and ImageNet data set reach 0.4 and 0.3 respectively. The convergence of the evaluation curves is quite chaotic, and it seems that at the later part our training process, the training accuracy of the network model is not further improved, or the improvement is too small to be visible from the plot.

Figure 4 and Figure 5 show training process when we were using the alternate mode for the batch composition. During the first few training iterations, the training error is pretty high. The reason is that we calculate the training error right before we train the network model, and the model is overfitting after each step since we would feed in the model with images of the same category every mini-batch.

Alternating the category of objects and the overfitting problems are also reflected in the sustained and substantial fluctuation of the cross-entropy, which is an indication that the way we feed

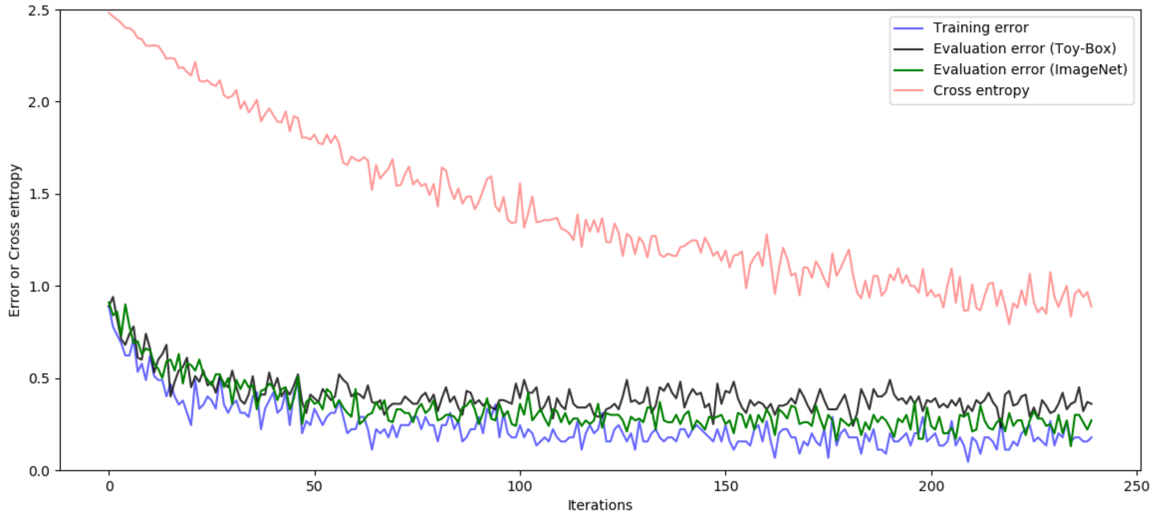

Figure 3. Measures of training and testing error using mini-batches selected using the uniform mode, across first 240 training iterations.

in the training images is problematic. We can see that the cross-entropy, training error and evaluation error dropped the most in the first 1000 iterations, yet failed to maintain this trend later on.

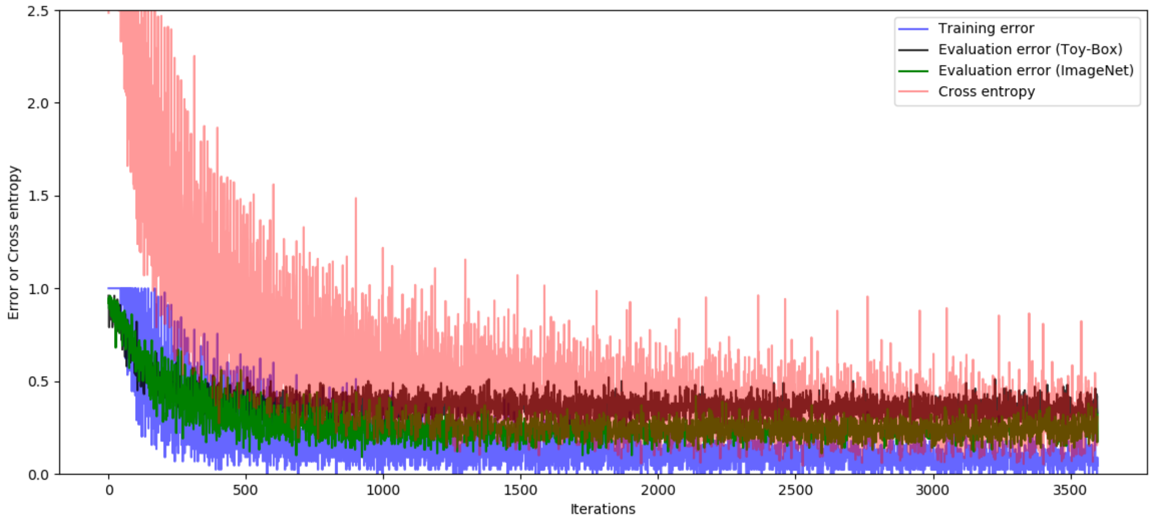

Figure 4. Measures of training and testing error using mini-batches selected using the alternate mode, across 3600 training iterations.

Figure 6 and Figure 7 show training process when we were using the sequential mode for the batch composition. Both plots show apparent periodical patterns on both the training error and cross-entropy curves. Peaks of both curves are corresponding to the first batches of each category

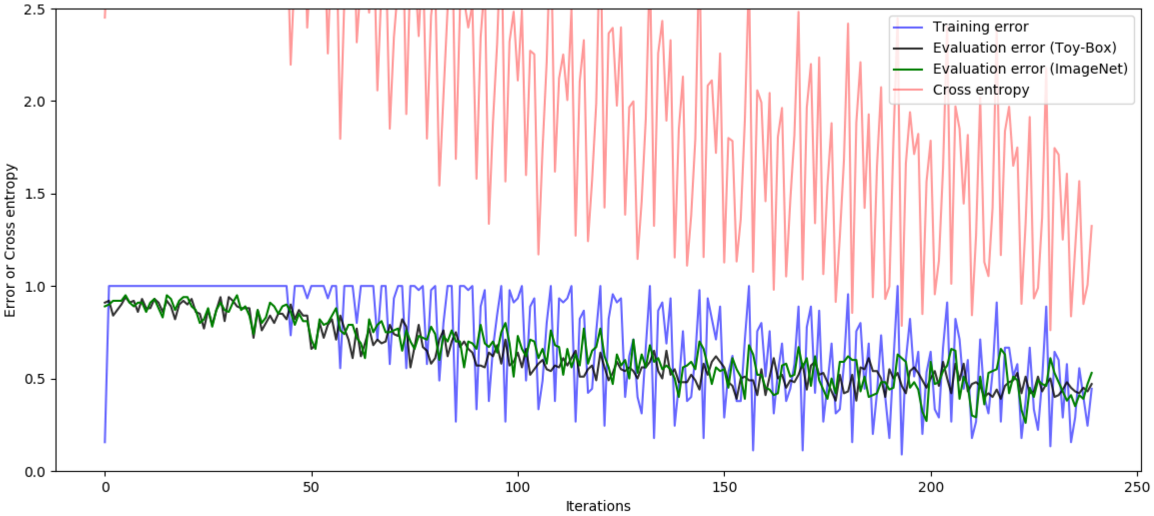

Figure 5. Measures of training and testing error using mini-batches selected using the alternate mode, across first 240 training iterations.

used for training. In this mode, we also feed in the model images of the same category at each step. However, when we start to feed in a batch of images from a new category, the current network model is highly likely to categorize these images to be of the previous category, which might cause an extremely high training error and cross-entropy. The network model then quickly learn from the new category of objects and start to be overfitting again.

Comparing to the cross-entropy in the previous experiment, cross-entropy we got using the sequential mode showed even more substantial fluctuation, and it had a severe impact on the training efficiency. We can see that with the same number of training iterations, the network model could not even achieve the same evaluation error comparing to models we trained using two other modes.

To more directly compare the training efficiency between three different modes, for each mode, we completed the training process 50 trials using 1080 images per category from the Toybox data set, with a batch size of 45 and ran 240 iterations of the training process. For reference, we also ran another 50 trials using the random mode in which we randomly selected training images for every mini-batch. Then we plotted probability density estimation curves of average evaluation errors (computed using the evaluation errors against Toybox validation data set) for all four modes. Distributions of average evaluation errors are shown in Figure 8.

Just as reflected in the previous plots, for a training process of only 240 iterations, we got the largest average evaluation errors with the sequential mode, followed by the alternate mode. We got the smallest average evaluation errors with the uniform mode and random mode. Surprisingly, comparing to the random mode, we got even smaller average evaluation errors on average using the uniform mode. The reason uniform mode could outperform random mode is that it is guaranteed in the uniform mode our network model could learn from each category equally at each step so that the network model could more easily overcome the initial overfitting problem, while it is not guaranteed when using the random mode. Thus, with our current hyperparameter settings, it is not

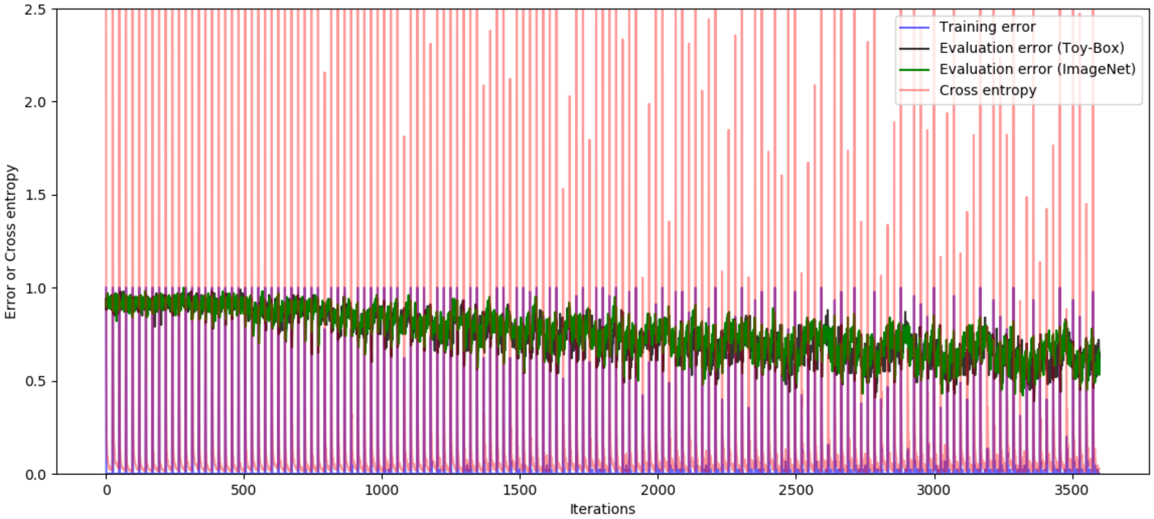

Figure 6. Measures of training and testing error using mini-batches selected using the sequential mode, across 3600 training iterations.

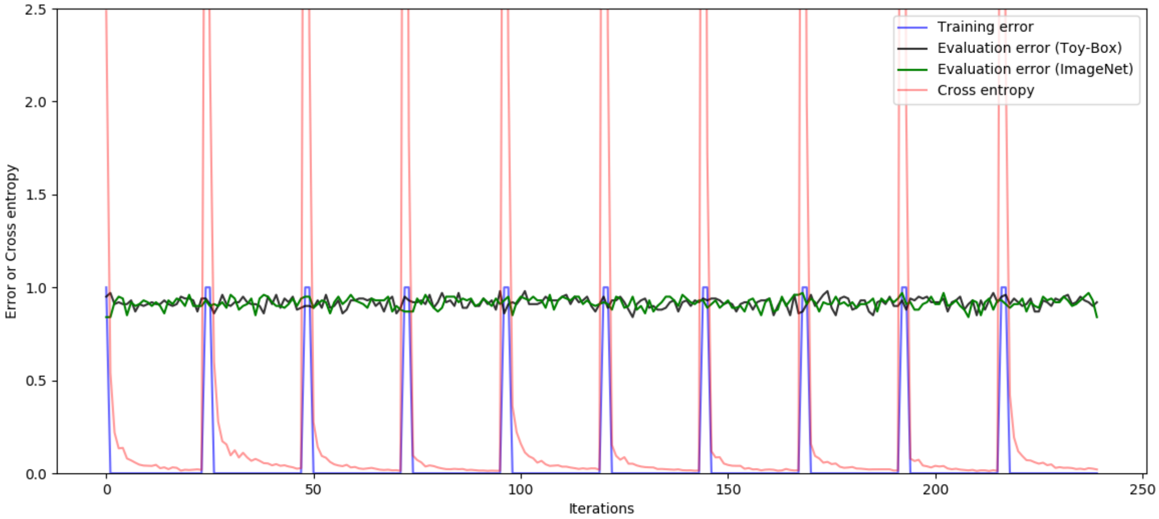

Figure 7. Measures of training and testing error using mini-batches selected using the sequential mode, across first 240 training iterations.

difficult to observe that the more uniformly training data of each category we distribute within and across mini-batches, the higher training efficiency we can get.

It is an interesting question to consider whether uniform selection would continue to outperform random selection in additional experiments and/or task domains. Considering that the network model (Inception-v3 model in our case) has a considerable amount of neurons, the model may end up learning something about uniformly distributed batches, just as it may pick up other trivial

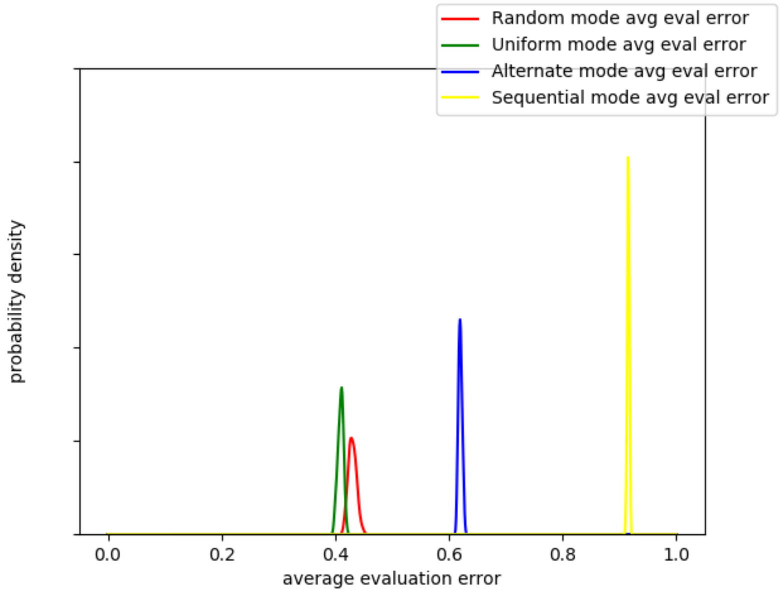

Figure 8. Probability density estimations of average evaluation errors for training regimes using four different modes.

information (i.e., overfitting) in various ways. Additional experiments are needed to better elucidate the differences we have identified between uniform and random training data selection methods.

2.4 Effects of Training Data Composition

Besides the way we form a mini-batch, to explore factors that may affect the training efficiency, we also have to take into account the overall composition of the training data set. In previous experiments, we used 12 objects per category for training and got a reasonably good training accuracy and efficiency (using the random mode or uniform mode) considering that the diversity of our training data set is much worse than that of the ImageNet data set. In the following experiments, we will explore how the diversity of objects (number of different objects in the training data set) and the number of different images per object in the training data set would affect the training efficiency.

To explore the effects of objects' diversity, we will keep the number of images per category to be 1080, and change the number of objects each category contains in the training data set. We will use the random mode (randomly select training images from the entire training data set) to form the mini-batch, with a batch size of 45, and run the training for 2880 iterations. We will investigate how the training accuracy and efficiency would change when there are 2, 4, 6, ..., 12 objects per category (so that each object would appear in roughly 540, 270, 180, ..., 90 different images) in the training data set. For each experiment, we ran 50 trials and plotted all the probability density curves of average evaluation errors (computed using the evaluation errors against ImageNet validation data set) in Figure 9.

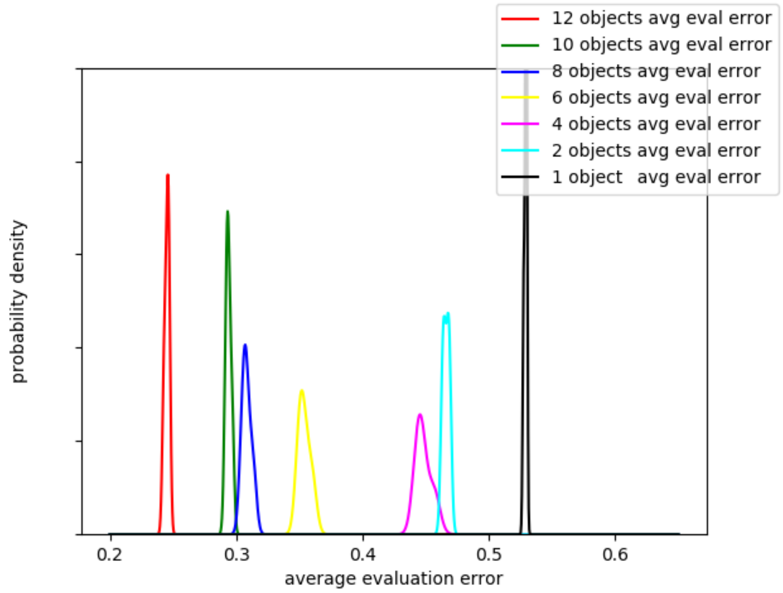

Figure 9. Probability density estimations of average evaluation errors for training regimes using different distributions of object diversity.

We can see from the plot that when the diversity of objects in our training data set is relatively large, with the same number of iterations in our extended training process, the network model will reach lower average evaluation error, an indication of higher training efficiency. However, this result is corresponding to the training process with our current settings, which has a large number of training iterations, and we also want to know how the average evaluation error would change when we change the length of the training process (number of training iterations).

We plot the change of average evaluation error up to a specific training step in our training process. From Figure 10, we can see that the relationship of average evaluation errors in the first 300 training iterations for various numbers of objects in the training data set is not stable. Especially in the first 100 iterations, all curves intertwine with each other. However, after a certain number of training iterations (1500 iterations in our case), as shown in Figure 11, we know that the relationship of average evaluation errors becomes stable. The more different objects we use in our training data set, the smaller average evaluation errors we could achieve in the long run.

To explore how the number of images per object would affect the training efficiency, we only have to keep the total number of images used in the training process the same (batch size times number of training iterations) and change the number of images per object used in the training data set. We will still use the random mode (randomly select training images from the entire training data set) to form the mini-batch, with a batch size of 45, and run the training for 2880 iterations. In the training data set, there will be 12 objects per category, and we will investigate how the training efficiency would change when there are 2, 4, 10, 20, 40, 60, 80, and 90 images for each object. For

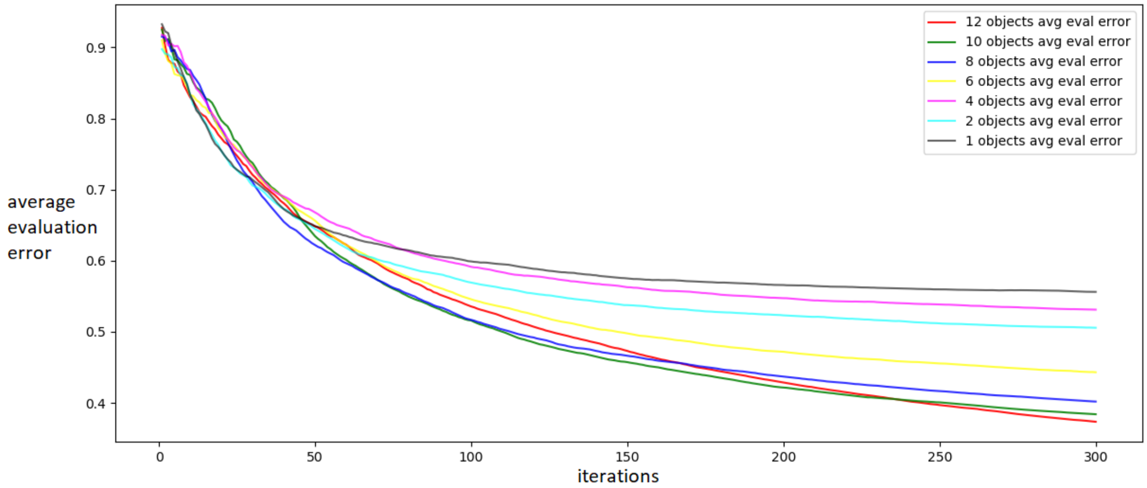

Figure 10. Average evaluation errors for training regimes using different distributions of object diversity, across 300 training iterations.

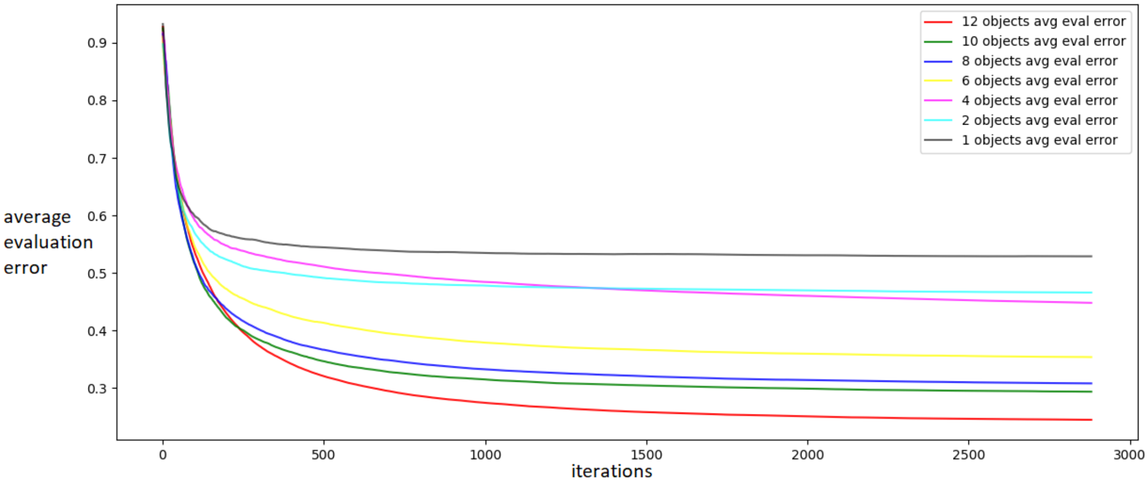

Figure 11. Average evaluation errors for training regimes using different distributions of object diversity, across 2880 training iterations.

each experiment, we ran 50 trials and plotted all the probability density curves of average evaluation errors (computed using the evaluation errors against ImageNet validation data set) in Figure 12.

We can see from the plot that the distribution of average evaluation errors is much fuzzier than expected. We can still find out, though, when there are more different images per object in a category, we will get overall lower average evaluation errors, an indication of higher training efficiency. However, there are also exceptions. For example, when there are 10 images per object in our train-

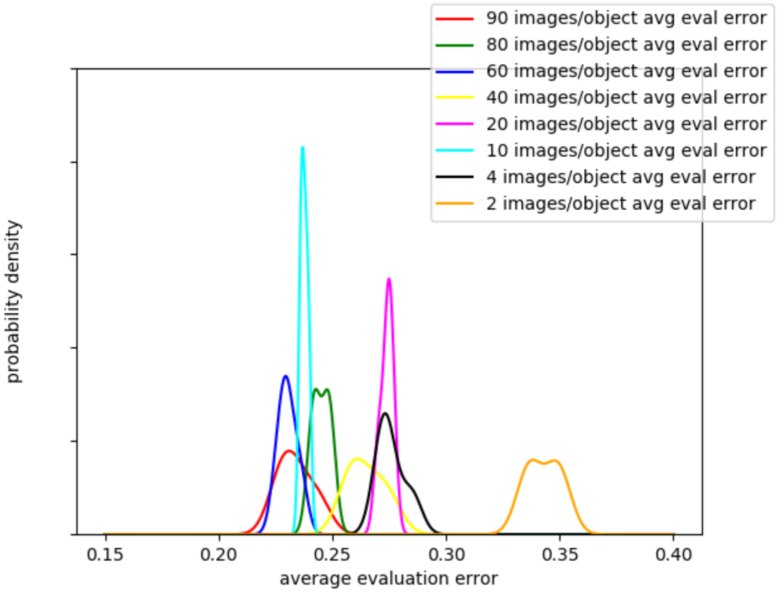

Figure 12. Probability density estimations of average evaluation errors for training regimes using different distributions of images per object (view diversity).

ing data set, we will get lower average evaluation errors than if there are 20, 40 or even 80 images per object in the training data set.

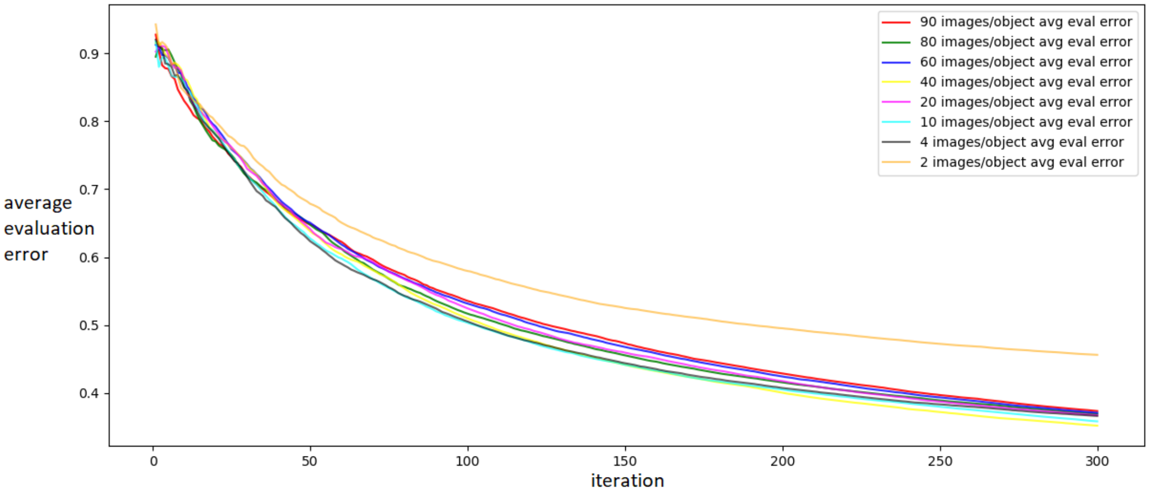

Figure 13. Average evaluation errors for training regimes using different distributions of images per object (view diversity), across 300 training iterations.

To get more insight into the change of average evaluation errors during the training process, we plot trends of average evaluation error up to a specific training step for the different number of images per object in Figure 13 and Figure 14 (for 300 iterations and 2880 iterations respectively).

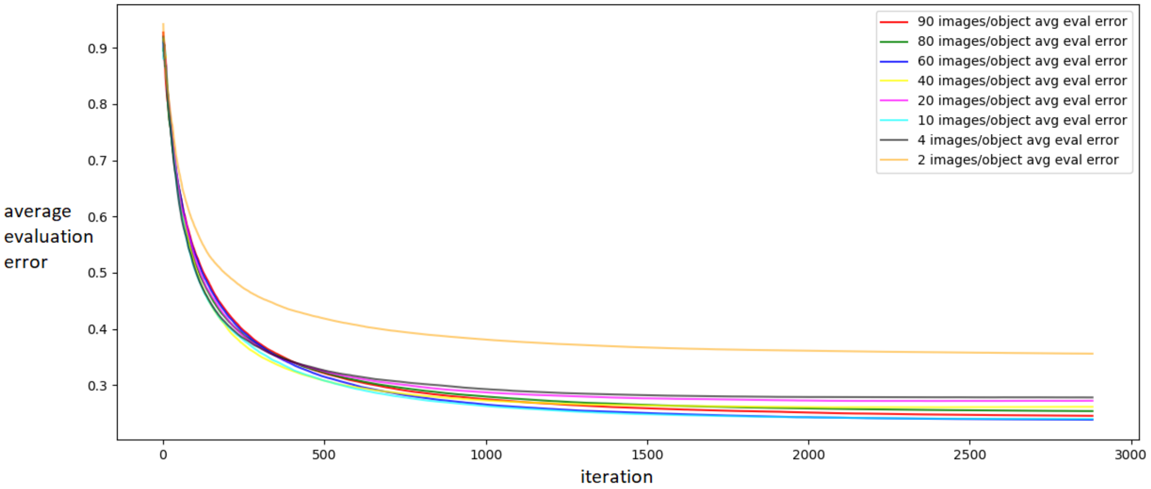

Figure 14. Average evaluation errors for training regimes using different distributions of images per object (view diversity), across 2880 training iterations.

From both plots, we can see that when we used at least four images per object in out training data set, there would be no significant difference regarding the training efficiency. Thus, with our current hyperparameter settings, the number of different images per object in the training data set does not have a significant impact on the training efficiency as long as that number is above some threshold, although it is still unclear what would contribute to that threshold.

One possible reason that increasing the number of images per object may not always help to improve the training efficiency could be that different images of an object are not distinguished enough for the network model to learn more features, a problem caused by the nature of the Toybox data set. We might be able to provide cure to this problem, however, by manually selecting higher quality images for each object so that each of an object can be much more distinctive.

3. Conclusions

From all experiments we have done, we know that not only the composition of a mini-batch but also the composition of entire training data set will have a substantial impact on the training efficiency. Using mini-batches, in which images of all different categories are distributed evenly, we can in general, achieve higher training efficiency. In fact, all non-uniformly distributed mini-batches can impair the training efficiency to some extent, depending on how severe the overfitting problems they can cause in the training process. Specifically, we found that when we ensure that images of all categories in the data set appear in every single mini-batch during the training process, the network model will achieve a training efficiency as high as, if not higher than, the training efficiency of randomly assigning images in a mini-batch.

Generally speaking, current training approaches suffer from overfitting when we only allow a single or limited number of categories to be in a mini-batch. The situation gets worse if we further limit the diversity of categories between every mini-batch of data we feed into the network model. To ensure that the training efficiency is as high as possible, we have to distribute images of different categories evenly within and between mini-batches.

These results raise several important research questions for the study of learning, both in humans and in artificial intelligent agents. Do humans have a "visual learning" algorithm that is drastically different from stochastic gradient descent, such that extended sequences of exposure to just one or a few object categories can still support successful learning? If so, what does this algorithm look like, and does it represent a class of algorithms somehow fundamentally different from stochastic gradient descent? And could it provide learning advantages to intelligent agents that are exposed to human-like learning environments? For instance, as we start to move from explicitly designing the mechanisms of AI systems targeting specific problems towards artificial general intelligence, it may be well worth exploring not only how agents would react towards human-like input signals but also how humans' choices of interaction with agents would affect these cognitive or learning processes.

Our research will continue to investigate these questions using more detailed empirical experiments of the kind presented in this paper, as well as experiments that use automated methods of selected training input orderings and compositions, instead of relying on manually defined distributions. In the long run, we hope to implement these learning algorithms on real-world systems in order to study learning under more complex and difficult conditions.

Acknowledgements

This research was funded in part by a Vanderbilt Discovery Grant, titled New Explorations in Visual Object Recognition. We thank James Ainooson, Fuxin Li, Azhar Molla, Alan Peters, Jim Rehg, Linda Smith, and Chen Yu for their contributions to this project.

References

Bengio, Y., Louradour, J., Collobert, R., & Weston, J. (2009). Curriculum learning. *Proceedings of the Twenty-Sixth Annual International Conference on Machine Learning* (pp. 41–48).

Blum, A. L., & Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, *97*, 245–271.

Bottou, L. (2009). Curiously fast convergence of some stochastic gradient descent algorithms. *Proceedings of the Symposium on Learning and Data Science*.

Carvalho, P. F., & Goldstone, R. L. (2014). Putting category learning in order: Category structure and temporal arrangement affect the benefit of interleaved over blocked study. *Memory & Cognition*, *42*, 481–495.

Clerkin, E. M., Hart, E., Rehg, J. M., Yu, C., & Smith, L. B. (2017). Real-world visual statistics and infants' first-learned object names. *Philosophical Transactions of the Royal Society B*, *372*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Li, F.-F. (2009). Imagenet: A large-scale hierarchical image database. *Proceedings of the IEEE Conference on Computer Vision and Pattern*

Recognition (pp. 248–255).

Fisher, D. (1996). Iterative optimization and simplification of hierarchical clusterings. *Journal of artificial intelligence research*, *4*, 147–178.

Graves, A., Bellemare, M. G., Menick, J., Munos, R., & Kavukcuoglu, K. (2017). Automated curriculum learning for neural networks. *Proceedings of the Thirty-Fourth International Conference on Machine Learning* (pp. 1311–1320).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Proceedings of the Annual Conference on Neural Information Processing Systems* (pp. 1097–1105).

Langley, P. (1995). Order effects in incremental learning. *Learning in humans and machines: Towards an interdisciplinary learning science. Pergamon*, *136*, 137.

Nerb, J., Ritter, F. E., & Langley, P. (2007). Rules of order: Process models of human learning. *In order to learn: How the sequences of topics affect learning*, (pp. 57–69).

Pirolli, P., & Card, S. (1999). Information foraging. *Psychological review*, *106*, 643.

Settles, B. (2012). Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *6*, 1–114.

Wang, X., Eliott, F. M., Ainooson, J., Palmer, J. H., & Kunda, M. (2017). An object is worth six thousand pictures: The egocentric, manual, multi-image (emmi) dataset. *Proceedings of the IEEE International Conference on Computer Vision Workshop on Egocentric Perception, Interaction, and Computing* (pp. 2364–2372).