Intelligent Mini-Batch Sampling for Gradient Descent

Aditi Garg
University of Massachusetts, Amherst
Amherst, MA, USA
aditigarg@umass.edu

Abstract

Mini-batch gradient descent is a widely used algorithm to train classifiers. The standard way of selecting these batches is uniform sampling. This paper presents various methods to improve diversity of the mini batch within each iteration while improving informativeness across iterations. Experimental results show that even though these methods show improvement in accuracy in the initial few iterations and lead to lower loss with a quicker convergence, random sampling almost always leads to better accuracy except in the case of non-uniform class distribution in the training set and small training sets.

1. Introduction

Mini-batch gradient descent algorithm uses the selection of a mini batch at every iteration to compute the gradient of the loss function and accordingly update the model parameters. Random sampling to obtain the mini batch works on the assumption that this batch provides a good representation of the entire training set. However, it may be possible that a sampled batch contains examples that are very similar to one another, thereby leading to less learning in an iteration and more jaggedness in the loss curve. Also, it is possible that across iterations, we end up choosing examples that do not add new information to the model, thus delaying convergence.

In this project, we explore various methods to achieve diversity in an iteration and also improve informativeness across iterations. We compare the validation accuracies as well as the convergence rates for all these methods. Additionally, we examine their impact on a small dataset and an imbalanced dataset.

We conclude that for CIFAR-10, various algorithms that improve diversity within a mini batch, do not lead to faster convergence or better accuracy than SGD with random sampling. The methods we use to improve informativeness across iterations lead to a lower loss, but do not translate into better validation accuracy. However, we analyze the impact of these methods on small datasets and imbalanced class distribution and show that they lead to better accuracy and quicker convergence for these cases.

2. Background and/or Related Work

Since stochastic gradient descent is a widely used algorithm, many methods have been proposed to improve its accuracy and convergence rate. As randomness of the mini-batch can lead to slower convergence, multiple algorithms, including stratified sampling[1], Poisson Disk Sampling[3] and Submodular Batch Selection[2] attempt to address this by ensuring that we pick diverse examples in a mini-batch.

Stratified sampling[1] aims at using clusters to form mini-batches, such that intra-cluster variance is minimum and each cluster contains points belonging to the same output class. Poisson Disk Sampling[3] attempts to avoid repeating examples within a given adaptive radius of a previously selected example. Submodular batch selection[2] uses redundancy score to avoid picking similar points in the mini batch.

To maximize learning across iterations, submodular batch selection[2] makes use of uncertainty score, which helps it pick the examples that maximize learning, while also balancing it with the redundancy scores so that the mini-batch does not end up containing all high-entropy similar data points. [4] suggests inclusion of data points that are neither too easy nor too difficult for the model too classify, i.e., it neither includes examples that the model predicts with high confidence nor ones that are too difficult for it to classify (or might be noisy).

In this project, we attempt to combine these approaches to incrementally come up with models that maximize diversity within an iteration and learning across multiple iterations. The clustering techniques that we use can be pre-computed and do not add to the training time. We make use of -

- stratified sampling with clustering
- learning from clusters not used before in the training process
- mingling indices to decide the initial priorities of the clusters

to obtain the most diverse and informative mini-batches across iterations. To obtain mingling indices in this project, we use a slightly different definition from [3], where mingling index isn't the ratio of points (within the neighborhood of a given training point) with a different output class but the number of points with a different output class. We use this definition as it reduces computation and does not impact the cluster priorities.

3. Approach

The project incrementally explores various methods for intelligent mini-batch selection for classifier training. The baselines for all our comparisons are the models learnt by using mini batches formed by –

- Uniform random sampling with repetition.
- Sampling from only one output class per iteration (worst case).

3.1 Diversity within an iteration

Increasing the diversity of the mini batch selected per iteration leads to less jaggedness in the loss curve as the likelihood of selection of extremely different mini-batches across iterations reduces. This helps in faster convergence of the training process.

3.1.1 Training without replacement

This method is similar to random sampling but does not allow repetition of any datapoints in the mini batch. This slightly improves the chances of diversity in a mini batch.

3.1.2 Stratified sampling

In this algorithm, we divide the training set into strata using the output class such that each stratum contains all points from one output class. In each iteration, we pick equal number of examples from each stratum without replacement, thus enforcing equal representation of all output classes in a mini-batch and hence further diversifying our mini-batch.

3.1.3 Stratified sampling with clusters

Within each stratum of the previous algorithm, we create clusters of similar datapoints. While choosing the mini batch, we not only ensure equal number of examples for each stratum, but also that all examples are picked from different clusters.

3.2 Informativeness across iterations

In the previous section, we improve diversity within each iteration, but training examples can repeat across iterations. Selecting more diverse training examples across iterations could add more information to the model training. To ensure more informativeness (comparable to the uncertainty score[2]), we assume that clusters whose examples have been picked in previous iterations would add less knowledge to the model. Hence, we assign priorities to all clusters and reduce the priority once we pick an example from it. At each stage, we pick examples from the clusters with highest priorities.

3.2.1 Equal Initial Priority

In this method, we assign equal initial priority to all clusters, and reduce the cluster priority once we select an example from it.

3.2.2 Easy Examples First

Here, we assign higher initial priority to clusters farthest from the classification decision boundary. These examples are generally more representative of the class that they belong to, hence considered easier to classify.

To compute distance from the decision boundary, we make use of mingling index[3]. Mingling index[3] for a point is the number of nearest neighbors (using KNN) that do not belong to the same output class. For a cluster, its mingling index is the average of the mingling index of all the points in the cluster.

In easy examples first, we assign highest priority to clusters with lowest mingling index, i.e., ones farthest from the decision boundary.

3.2.3 Hard Examples First

In this method, highest priority is assigned to the cluster with highest mingling index, i.e., ones closest to the decision boundary.

4. Experiments

We test our methods on the –

- CIFAR-10 dataset (with 49,000 training examples and 1000 validation examples)
- a small subset of the CIFAR-10 dataset with 100 training points and 1000 validation points
- an imbalanced dataset containing only two classes in the training and validation sets (derived from the CIFAR-10 dataset)

Using our mini batch sampling techniques, on each of these datasets, we train a -

- softmax classifier
- two-layer neural network [input -> fully connected layer -> ReLU -> fully connected layer -> softmax]

The measures used to quantify the effectiveness of our methods are –

- classification accuracy on the validation set
- number of iterations for convergence

4.1 Hyperparameters

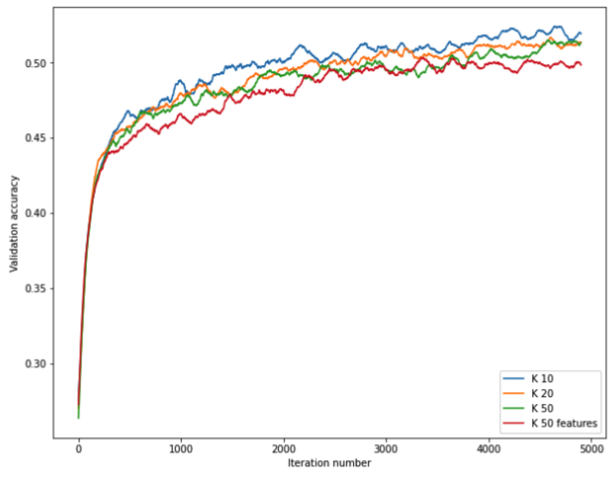
Initially, we perform comparisons for all the algorithms on the same set of hyperparameters [Table 1].

Architecture	Two-Layer Neural Network	Softmax
Number of Iterations	10000	3000
Batch Size	200	200
Learning Rate	1.75e-3	5e-7
Learning Rate Decay	0.95	0.95
Regularization Strength	0.25	2e+4
Hidden Layer Size	100	-

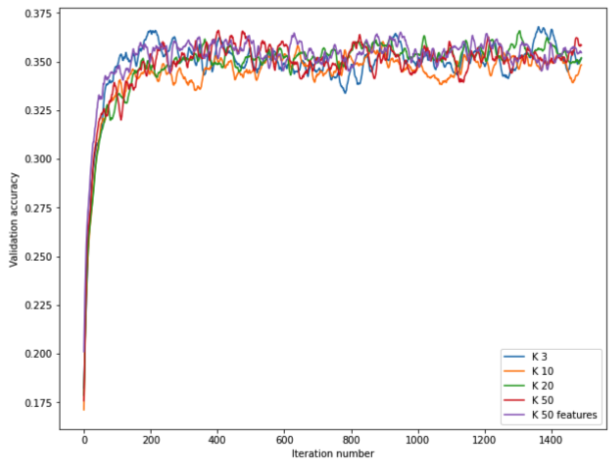
Table 1. Hyperparameters used for training the models

4.2 Clustering

To form clusters, we employ the K-means clustering algorithm. We explore multiple values of K while clustering on raw pixels. We then compare it with clustering on feature vectors for each image, formed by concatenating the HOG (Histogram of Oriented Gradients) and color histogram feature vectors [Figure 1] and select the clustering that gives the highest accuracy for comparison with other sampling methods.



(a)

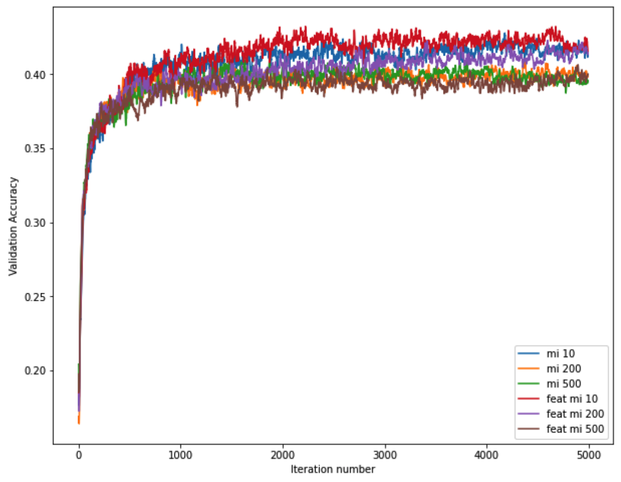


(b)

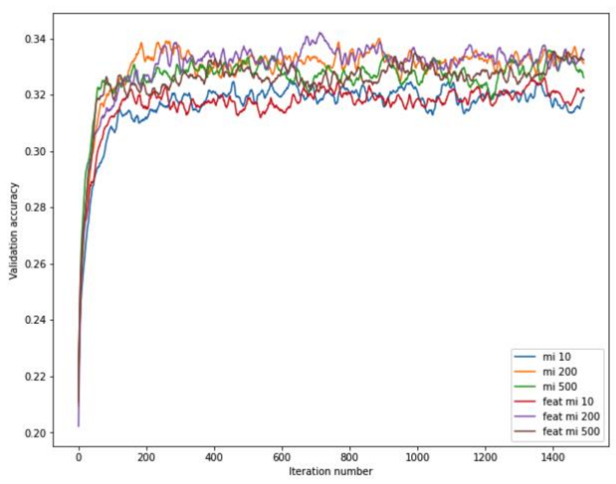
Figure 1. Validation accuracies for *stratified sampling with clusters* using different values of K during K means clustering on raw image pixels and image features (*K 50 features*) for – (a) Two-layer neural network (averaged across 100 iterations), (b) Softmax classifier (averaged across 10 iterations)

4.3 Mingling Index

We experiment with multiple values of K, while finding the K nearest neighbors using raw images and image features (formed by concatenating the *histogram of oriented gradients* and *color histogram* feature vectors) to compute the mingling index [Figure 2].



(a)



(b)

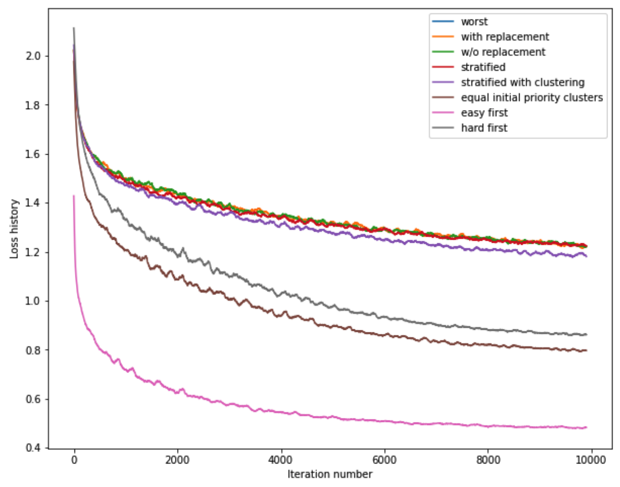
Figure 2. Validation accuracies for *easy examples first* using different values of K to compute mingling indices using KNN on raw image pixels and image features (*feat mi 10* etc) for – (a) Two-layer neural network, (b) Softmax classifier (averaged across 10 iterations)

4.4 Results

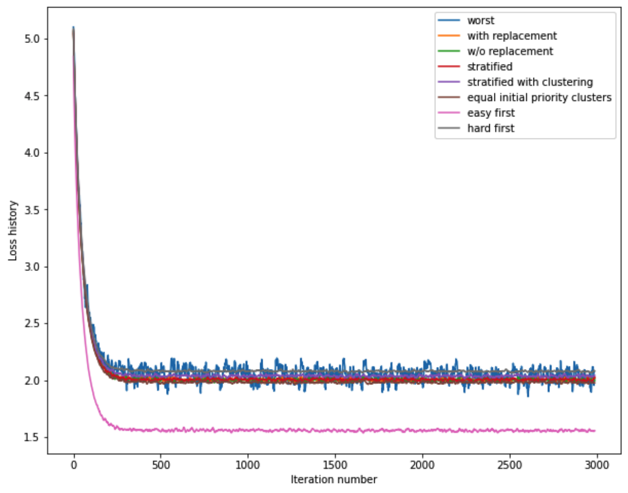
We report the classification accuracies and iterations for convergence obtained for various mini batch sampling techniques.

4.4.1 CIFAR-10 dataset

Figures 3 and 4 show the loss and the validation accuracy curves for all the above methods for both softmax and two-layer neural network averaged across 3 runs. We see that for the entire CIFAR-10 dataset, there is almost no change in the loss and accuracy for methods that diversify the mini batch within an iteration. The methods for more informativeness across iterations lead to lower loss, but that does not translate to higher validation accuracy. The training, validation accuracies and number of iterations for convergence (averaged across 3 runs) for the best model (one with highest validation accuracy) for each of these sampling techniques are tabulated in Table 2.

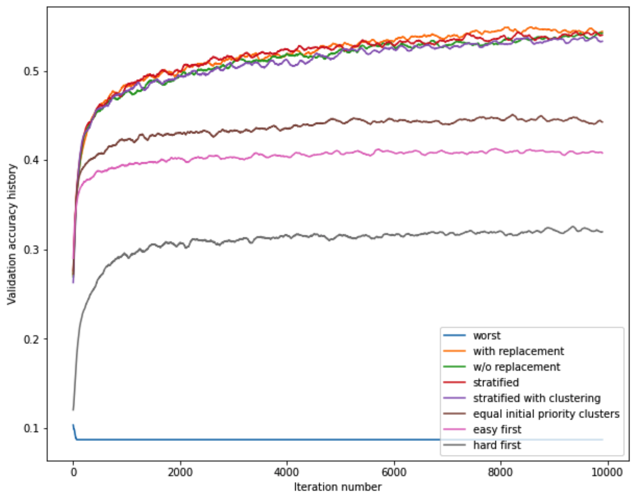


(a)

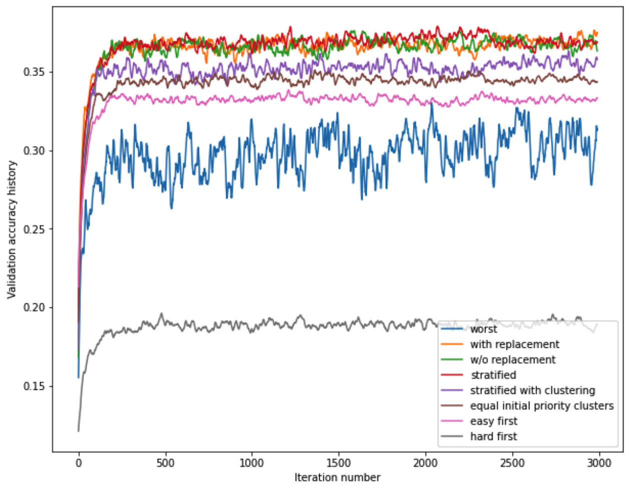


(b)

Figure 3. Loss curves for models training using different mini-batch sampling techniques for – (a) Two-layer neural network (averaged across 100 iterations), (b) Softmax classifier (averaged across 10 iterations)



(a)



(b)

Figure 4. Validation accuracy curves for models training using different mini-batch sampling techniques for – (a) Two-layer neural network (averaged across 100 iterations), (b) Softmax classifier (averaged across 10 iterations)

Mini-batch selection method	**Training Accuracy (in %)**	**Validation Accuracy (in %)**	**Iterations for convergence (approx.)**
Worst case	19.62	20.9	1
With replacement	63.87	56.85	9000
Without replacement	65.06	56.2	9000
Stratified	65.08	56.6	9000
Stratified with clustering	62.19	55.75	9000
Equal initial priority clusters	54.52	47.05	1300
Easy examples first clusters	46.93	42.95	1100
Hard examples first clusters	41.46	34.9	1500

(a)

Mini-batch selection method	**Training Accuracy**	**Validation Accuracy**	**Iterations for convergence**
Worst case	31.135	31.45	Not achieved within 3000
With replacement	35.65	37.2	220
Without replacement	35.27	36.45	220
Stratified	35.65	37.2	220
Stratified with clustering	35.25	35.25	220
Equal initial priority clusters	33.04	34.6	200
Easy examples first clusters	32.77	33.45	200
Hard examples first clusters	18.5	19	180

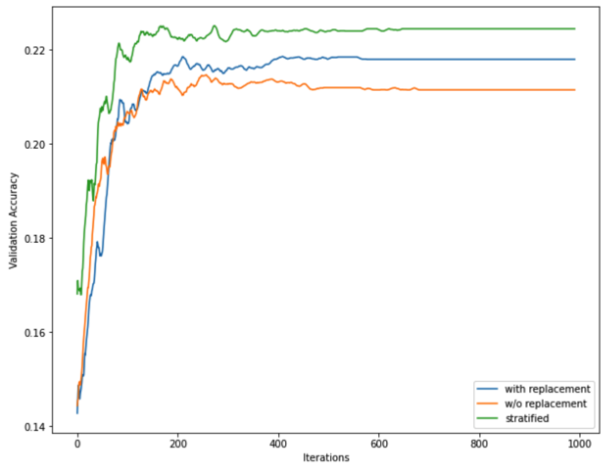
(b)

Table 2. Training and validation accuracies for the best model for each method on – (a) Two-layer neural network, (b) Softmax classifier

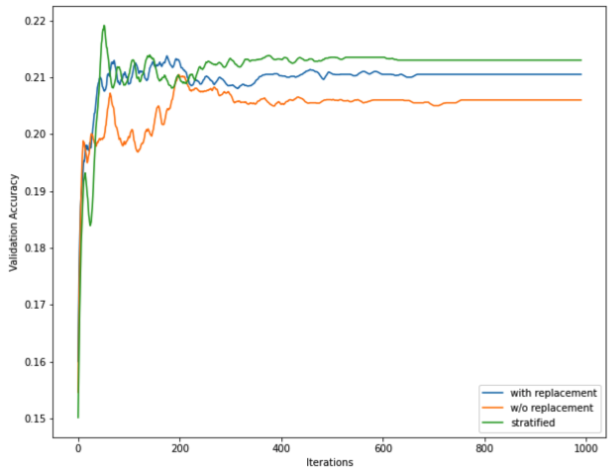
For two-layer neural network, learning for the worst case stops in the initial few stages leading to a constant validation accuracy in the later iterations. The convergence is the fastest for stratified sampling with clusters with assigned priority. However, this quicker convergence does not lead to a higher accuracy and the validation accuracy remains the highest for training with replacement.

4.4.2 Small dataset

We test out the impact of training without replacement and stratified sampling on a subset of CIFAR-10 dataset containing 100 training points. Contrary to the large dataset results, small datasets show better accuracy and quicker convergence for stratified sampling as shown in Figure 5. The hyperparameters used are listed in Table 3.



(a)



(b)

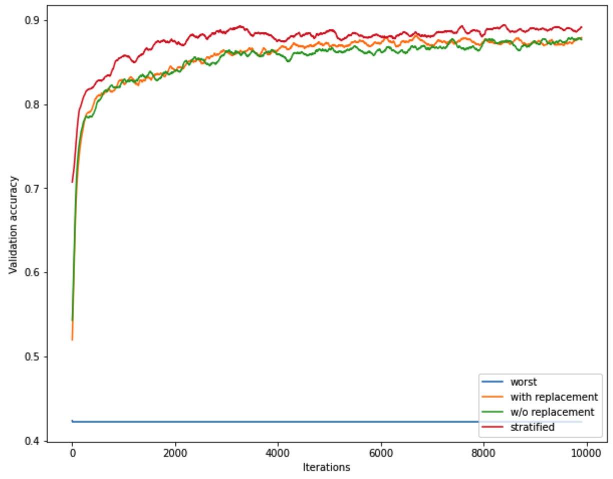
Figure 5. Validation accuracy for diversified mini batch training on a small dataset of – (a) Two-layer neural network, (b) softmax classifier

Architecture	Two-Layer Neural Network	Softmax
Number of Iterations	1000	1000
Batch Size	20	20
Learning Rate	1.75e-3	5e-7
Learning Rate Decay	0.95	0.95
Regularization Strength	0.3	2e+4
Hidden Layer Size	100	-

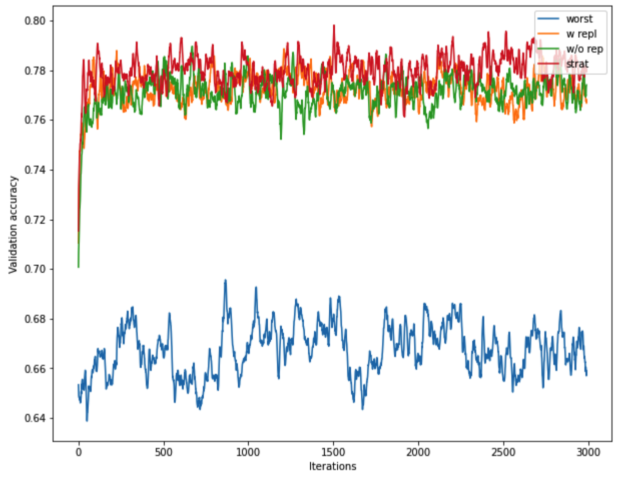
Table 3. Hyperparameters for training small dataset

4.4.3 Imbalanced dataset

We create an imbalanced dataset from the CIFAR-10 dataset using all datapoints with class label – *0*. If there are *n* datapoints in class *0*, we choose *n/2* datapoints for output class *1*. We modify the validation set to only include points from output classes *0* and *1*. For this imbalanced distribution, we obtain the highest validation accuracy and quickest convergence for stratified sampling. The hyperparameters used for this training are listed in Table 4.



(a)



(b)

Figure 6. Validation accuracy for diversified mini batch training on a imbalanced dataset of – (a) Two-layer neural network, (b) softmax classifier

Architecture	Two-Layer Neural Network	Softmax
Number of Iterations	10000	3000
Batch Size	50	40
Learning Rate	1.75e-4	5e-7
Learning Rate Decay	0.95	0.95

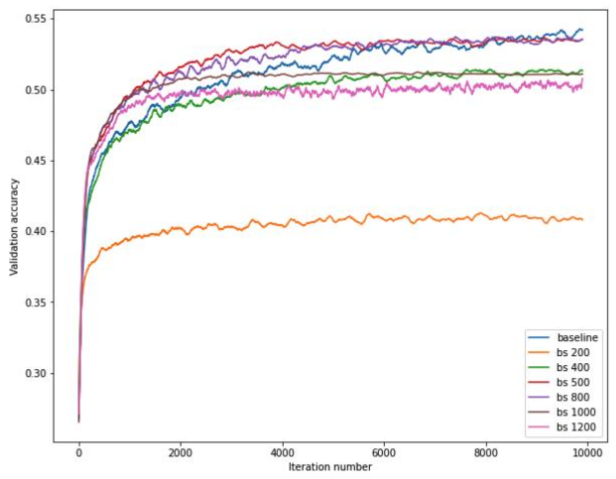
Regularization Strength	0.25	2e+4
Hidden Layer Size	100	-

Table 4. Hyperparameters for training imbalanced dataset

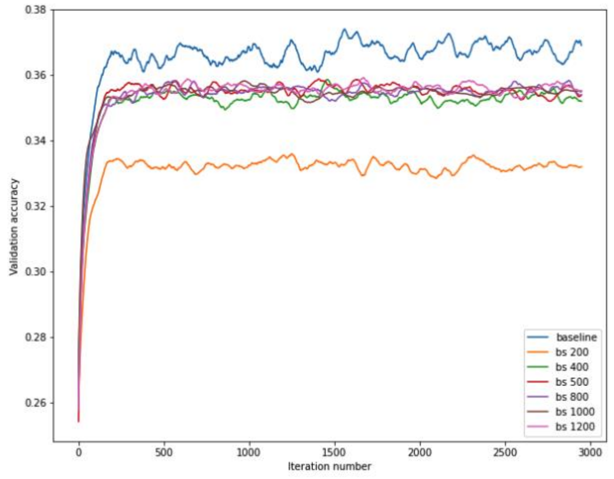
5. Conclusions

This analysis done for the mini-batch selection on CIFAR-10 data and its subsets suggests that –

- For large datasets, introducing diversity in a minibatch per iteration does not have any significant impact on the accuracy or convergence speed. This might be because of the large size of the dataset, which significantly reduces chances of selecting similar examples in an iteration. This is validated by selecting a small subset of the CIFAR10 dataset, for which stratified sampling gives both – a quicker convergence and a higher accuracy [Figure 5].
- For an imbalanced dataset, stratified sampling works much better than uniform sampling, as the training picks same number of examples from all the output classes despite unequal representation in the training set [Figure 6].
- Assigning priorities to clusters to avoid repetition of similar examples across iterations leads to a lower loss but does not translate to a high validation accuracy. The reason for this might be that even though the correct class probability increases, the increase isn't enough to perform accurate classification, implying that the model needs multiple examples of the same kind to learn, before moving on to examples that are of a different kind. This is validated in Figure 7, where we try stratified sampling with clustering (with clusters assigned equal initial priorities) using different batch sizes. The hyperparameters for all the curves in Figure 7 are listed in Table 1, with only the batch size being updated for every curve. In general, a higher batch size leads to more repetition of similar examples in successive iterations, leading to higher accuracy. Here, we see that even though clustering methods show high accuracy in the initial iterations, random sampling with replacement (baseline) outperforms them in the long run.



(a)



(b)

Figure 7. Validation accuracies for stratified sampling with clustering (with equal initial priority) using different batch sizes for the training of – (a) two-layer neural network (averaged across 100 iterations), (b) softmax classifier (averaged across 10 iterations).

Hence, we can say that in asymptotic time, for a high number of training examples, mini batches selected using random sampling act as regularisers, often leading to the best classification accuracies. But, when the dataset is small and/or imbalanced, stratified sampling results in quicker convergence and higher accuracy.

In the future, we can try these models on deeper and more complex network architectures. We can also explore different loss functions that coincide better with the validation accuracy in such cases. Since stratified learning performs well on smaller datasets, we can verify if such sampling methods can be used for few-shot learning. We can also explore the scope of these training methods in naturally imbalanced training sets, such that the model is fairer to classes with less representation.

6. References

[1] Zhao, P., & Zhang, T. Accelerating Minibatch Stochastic Gradient Descent using Stratified Sampling. ArXiv, abs/1405.3080, 2014

[2] K. J. Joseph, Vamshi Teja R., Krishnakant Singh, and Vineeth N. Balasubramanian. Submodular batch selection for training deep neural networks. In Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19). AAAI Press, 2677–2683, 2019

[3] Cheng Zhang, Cengiz Öztireli, Stephan Mandt, and Giampiero Salvi. Active mini-batch sampling using repulsive point processes. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI'19/IAAI'19/EAAI'19). AAAI Press, Article 704, 5741–5748, 2019

[4] Chang, Haw-Shiuan and Learned-Miller, Erik and McCallum, Andrew. Active Bias: Training More Accurate Neural Networks by Emphasizing High Variance Samples. ArXiv, abs/1704.07433, 2018.