CS 260D Project: F-CRAIG

Tazeem Khan
UID: 105946724

Vamsi Krishna Pamidi
UID: 805945580

Subham Agrawal
UID: 205948930

University of California, Los Angeles

Abstract

In this project, we propose an approach to speedup CRAIG by using the concept of Forgetting Events. Forgetting event is an event when an individual training example transitions from being classified correctly to incorrectly over the course of training. Data points are pruned based on their Forgetting Events score and then CRAIG is applied on the pruned data set to find ideal weighted subset. The idea is that calculating CRAIG on a smaller dataset will provide speedup while expecting little drop in accuracy as data points which are learnt by the model are dropped.

1. Introduction

Training large-scale machine learning systems is widely used with great success, but it requires a large amount of data and computational resources, as well as huge financial expenditures and energy supply[1]. Reducing the cost of training these systems without sacrificing accuracy is one of the major challenges in the field of machine learning and artificial intelligence[2].

In machine learning models, the aim is to find model parameters such that it minimizes convex loss over training data. Stochastic Gradient Descent can find the minimizer of this loss function, but it requires multiple computations of the full gradient i.e. sum of the gradients over all data points and is therefore not ideal for large data sets. This problem gets worse in the cases of deep neural networks where the cost of backpropagation is extremely expensive. Incremental Gradient (IG) methods, like Stochastic Gradient Descent (SGD) and its variants, including SGD with Momentum [3], Adagrad[4], Adam[5], SAGA[6], SVRG[7] iteratively calculate the gradient on random subsets of training dataset. This provides an unbiased estimate of full gradient, but the randomized batches introduce variance in the gradient estimate and therefore are slow to converge. Most of the research in speeding up Incremental Gradient has been in reducing the variance of the gradient estimates or using adaptive learning rates.

The other approach for a solution for this problem is to find a subset of training data such that the gradients of the subset closely represent that of the gradients of the whole training dataset. If we can compute the subset quickly, then this would lead to a speedup per epoch of IG. This approach was developed in the form of Coresets for Accelerating Incremental Gradient descent (CRAIG), for selecting a subset of training data points to increase the speed of large machine learning models. The key idea was to select a weighted subset of training data that is an ideal approximate the full gradient of the training dataset. The selected subset that minimizes an upper bound on the error of estimating the full gradient maximizes a submodular facility location function. Hence, the subset can be efficiently found using a greedy algorithm or lazy greedy algorithm. More importantly, it was also shown that any incremental gradient method (IG) on the subset converges in the same number of epochs as the same IG would on the full training dataset, which means that we obtain a speed-up inversely proportional to the size of the subset. But because every epoch only uses a subset of the data, it requires fewer gradient computations and thus leads to “a number of points in training dataset/number of points in the subset” speedup over traditional IG methods, while still (approximately) converging to the optimal solution.

Our aim was to improve the CRAIG algorithm by speeding up the process of subset selection. The first step in CRAIG is to calculate a similarity matrix for which the time complexity is $O(N^2)$ and as the dataset size increases, the computation power and space required increases as well. This is a limiting step in CRAIG as the data size grows to billions of samples since the lazy greedy itself has a time complexity of O(Nk). In every epoch of CRAIG, the subset selection occurs over the whole dataset and therefore the time to do CRAIG subset selection is constant in every epoch [8].

We leverage the concept of “Forgetting event” to prune datapoints after they are learnt by the machine learning models. A “forgetting event” to have occurred when an individual training example transitions from being classified correctly to incorrectly over the course of learning [9]. After every epoch, a certain % of data is pruned before CRAIG is implemented for subset selection. As the number of epochs increases, the training dataset gets smaller due to pruning and CRAIG is getting a speed up. Our method of improving CRAIG using forgetting events is called “F-CRAIG”.

We demonstrate the effectiveness of F-CRAIG via an extensive set of experiments on MNIST dataset using a simple neural network with different combinations of hyperparameters explained in the experiments section.

2. Methodology

A forgetting is event is when a data point is classified correctly in the previous iterations but then it is misclassified later. The count of such forgetting events is forgettability scores and it depends on the frequency in which this pointwise metric is evaluated.

The values of these forgetting scores can be used to classify whether a point is easy to learn for the model or whether it is very hard. If the value of the forgetting score is high, then it is difficult to learn otherwise it is easier to learn. This score is used as a criterion to evaluate if a data point can be pruned without making the training process inefficient.

If we compute forgetting scores once in every epoch, we will have to wait for few epochs before we start pruning data points. Hence, to be able to start pruning early, we are evaluating forgetting scores with a batch frequency. We further experiment with different batch frequencies to understand the speedups and impact in performance.

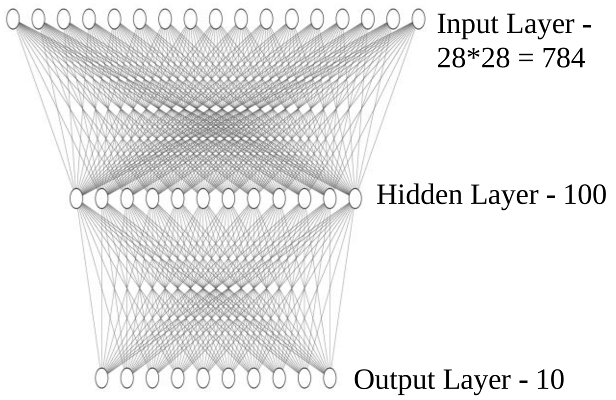
We then use these forgetting scores to prune our dataset by removing the learnt and unforgettable points. The approach we took is to sort data in ascending order based on forgetting scores and then drop the top x % of data points.

In F-CRAIG, this pruning is done just before the creation of the subset data for the craig, except during the first iteration as forgettability scores would not be present at the start. This pruning decreases the number of data points on which the CRAIG subset selection is performed. As the similarity matrix computation in CRAIG is of order $O(N^2)$, this reduces the time required and speeds up the process.

Below is a complete loop of F-CRAIG. In the initial iteration the entire data will be used for running the CRAIG for subset selection. After subset being selected, model is trained on this subset and the point-wise accuracy is stored in every few batches. After the backpropagation updates are made for first epoch, we calculate forgettability scores using the stored point-wise accuracies and then prune the data based on these scores. This pruned data is then used for subset selection. One important point to note is that the number of points in the subset remain constant even though the set of data points keeps getting smaller and smaller. This pruning is continued until the subset size dataset is remaining but can be stopped early as well.

3. Experimental Setup

We experiment on MNIST dataset containing 60,000 photos of handwritten digits each of shape (28 X 28). We use this data to train a neural network containing one hidden layer of 100 units and the output layer of 10 units. The activation function is sigmoid in each layer except the output layer. We use Softmax activation function in the final layer which outputs probabilities of digits or classes. A schematic of this neural network is given below.



We trained the model with different frequencies for forgettability scores calculation and different pruning percentages. However, the number of points subset points to be selected in CRAIG remains constant i.e. 40% of the training dataset.

4. Observations

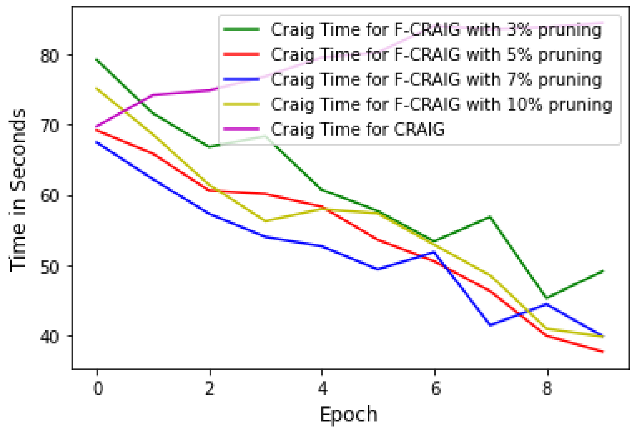
We estimate the time taken for F-CRAIG by executing 3 runs with each configuration. In the below plots, time taken to train the model for 10 epochs is called '**Total Time**' and the time taken for each epoch is plotted as '**Epoch Time**'. Epoch time includes the time taken for subset selection, forgettability score computation, data pruning, and model training.

We further observe time taken at each intermediary step,

1. **CRAIG time** – The total time taken by CRAIG for subset selection. The total time taken is the sum of time taken for all the classes.
2. **Prd time** – The time taken for execution of one forward pass of the entire data in case of CRAIG and pruned dataset in case of F-CRAIG.
3. **Sim time** – The time taken for the computation of the similarity matrix. We assume that the similarity matrix is calculated in parallel for each class and the maximum time taken among all classes gives us 'Sim time'.
4. **Grd time** – The time taken for the execution of the lazy greedy algorithm on the pruned/entire dataset. Similarly, this is also the maximum time taken among all classes by lazy greedy.
5. **GOW time** – The time taken for calculating the subset after we have the logits for each data point. The total time taken is the sum of time taken for all the classes.
6. **FGT score time** – The time taken to compute forgettability score and prune the dataset in each iteration.
7. **Train time** – Time taken for backpropagation at each epoch.

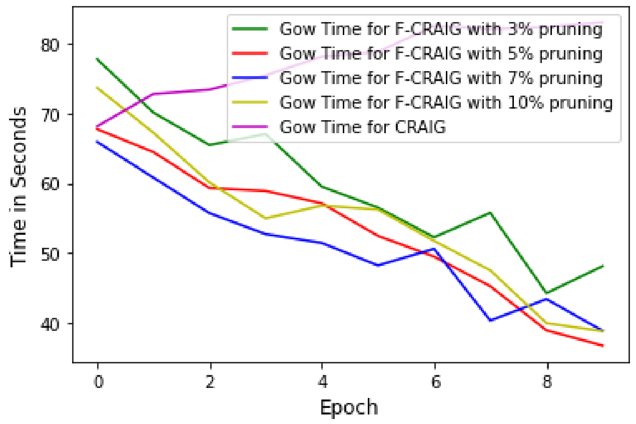
4.1 Pruning %

The frequency of the point-based accuracies calculation was kept constant at a rate of every 100 batches and the pruning % was changed. We ran F-CRAIG for pruning percentages – 3%, 5%, 7%, and 10%. For each of these pruning percentages, we plotted time plots.



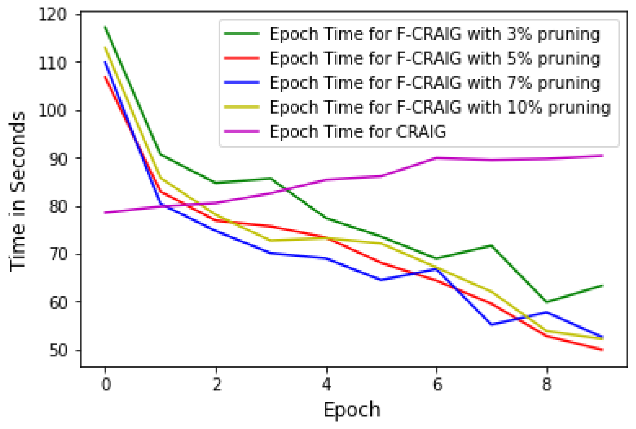
CRAIG Time of CRAIG vs F-CRAIG with Different Pruning %

The CRAIG time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. Initially, CRAIG time is same for both approaches, but we see a significant improvement in the last few epochs as a good proportion of data is pruned.



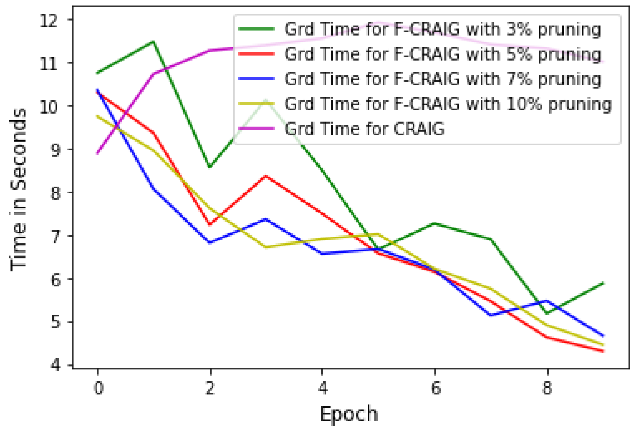
GOW Time of CRAIG vs F-CRAIG with Different Pruning %

The subset calculation time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. Initially, GOW time is same for both approaches, but we see a significant improvement in the last few epochs as a good proportion of data is pruned.



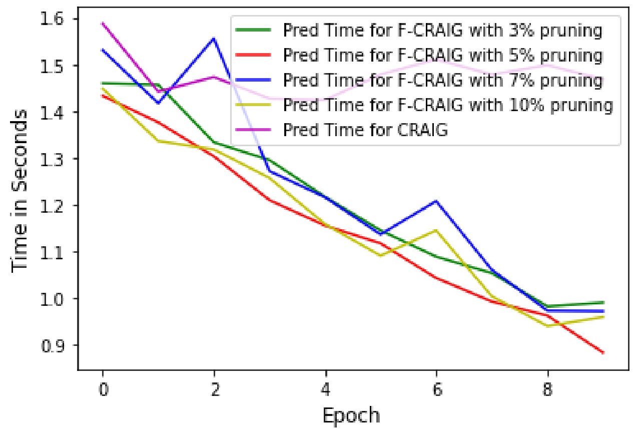
Epoch Time of CRAIG vs F-CRAIG with Different Pruning %

The epoch time for first few epochs is higher because additional time is taken to compute the forgettability metrics. However, starting from third epoch we see that epoch time is lesser from F-CRAIG as compared to CRAIG. This is because the time taken for forget scores is lesser than the time saved by data pruning. Also, epoch time remains contant for CRAIG but reduces with epoch in case of F-CRAIG.

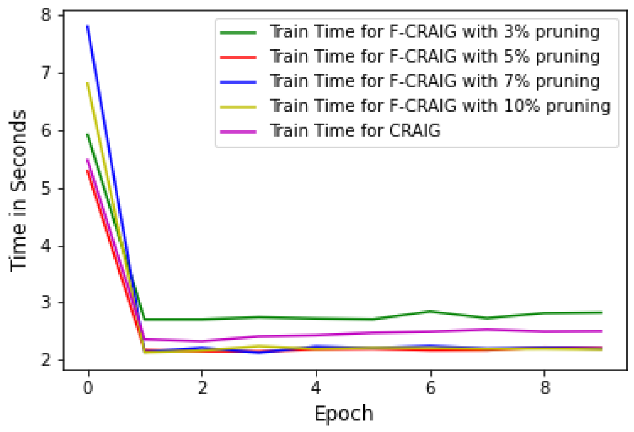


Greedy Time of CRAIG vs F-CRAIG with Different Pruning %

The lazy greedy evaluation time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. Initially, Greedy time is same for both approaches, but we see a significant improvement in the last few epochs as a good proportion of data is pruned. In different pruning%, the time Greedy time eventually converges as there is not a big difference in evaluation time once the data gets pruned.



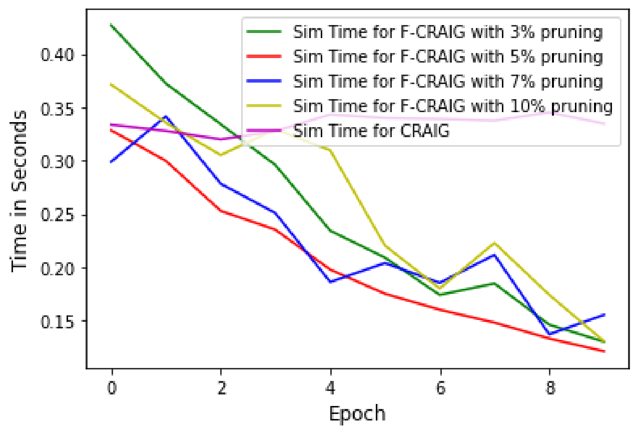
Prediction Time of CRAIG vs F-CRAIG with Different Pruning %



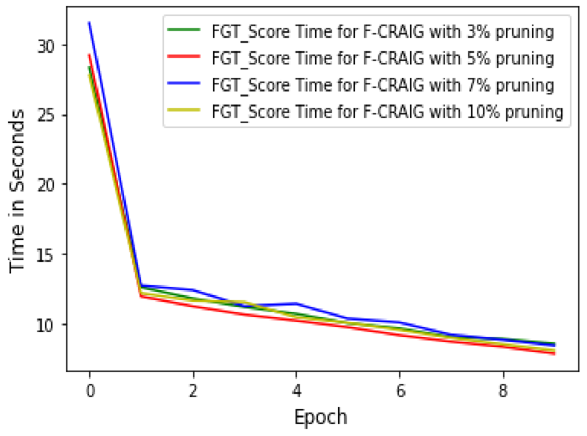
Training Time of CRAIG vs F-CRAIG with Different Pruning%

The Prediction time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. Initially, Prediction time is same for both approaches, but we see a significant improvement in the last few epochs as a good proportion of data is pruned.

The Training time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. F-CRAIG seems to outperform CRAIG in training time for all different pruning % except that of 3% data pruning.



SIm Matrix Time CRAIG vs F-CRAIG with Different Pruning %

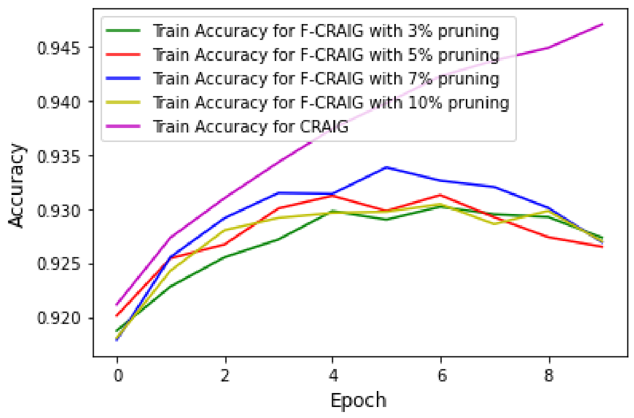


Forgetting Score Time of CRAIG vs F-CRAIG with Different Pruning%

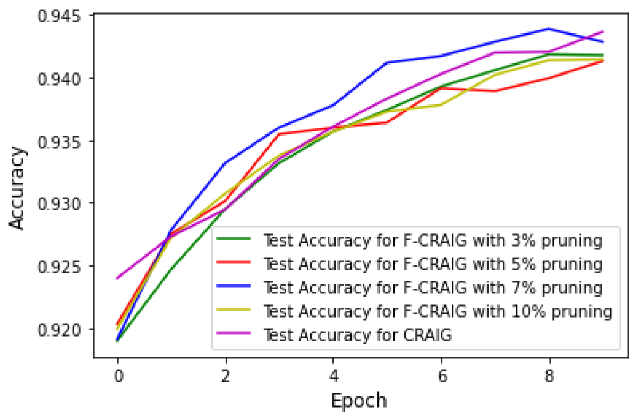
The Similarity matrix calculation time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. Initially, Similarity time is same for both approaches, but we see a significant improvement in the last few epochs as a good proportion of data is pruned

The Forgetting Scores calculation time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. FGT_Score time is same for different pruning%, and we see a similar performance times throughout the runs

From all the above plots, we do not see significant differences in run time for different pruning percentages. Even though we expected to see higher speedup for 10% pruning at each epoch as compared to 3% pruning, the results do not support the same. The results could show more changes when changing the dataset size or the training architecture and further experiments must be conducted.



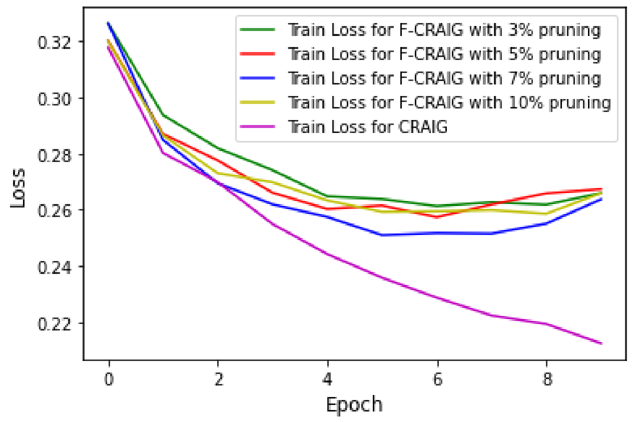
Train Accuracy of CRAIG vs F-CRAIG with Different Pruning %



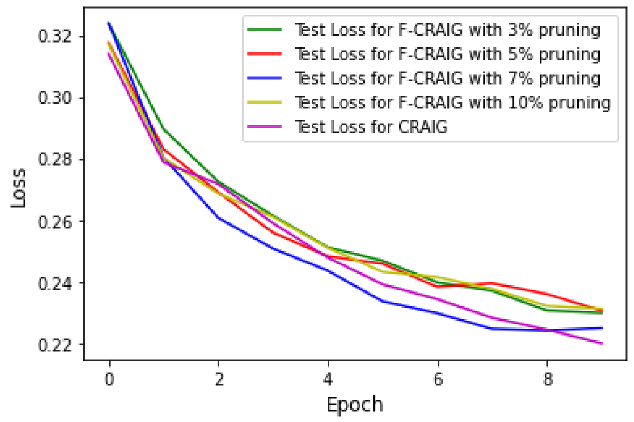
Test Accuracy of CRAIG vs F-CRAIG with Different Pruning %

The training accuracy of CRAIG seems to outperform that of F-CRAIG by around 2%. However, as it will be seen in the testing accuracy, it seems that F-CRAIG is underfitting instead of giving lower accuracy.

The testing accuracy of CRAIG is very similar to F-CRAIG by around. Even for different pruning %, it seems that F-CRAIG is giving similar performance on testing data meaning we can prune more data without any significant drops in accuracy.



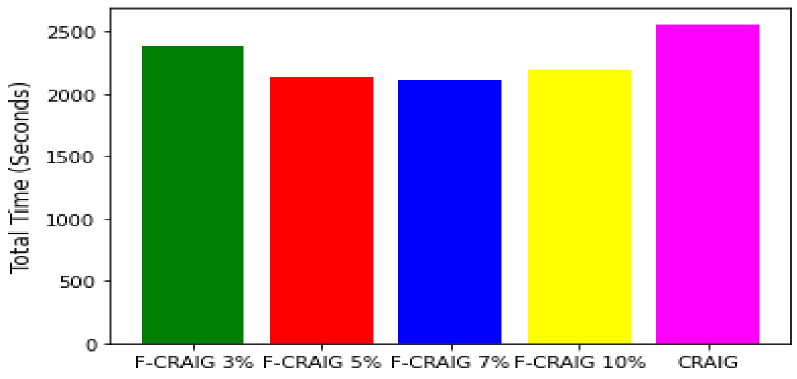
Train Loss of CRAIG vs F-CRAIG with Different Pruning %



Test Loss of CRAIG vs F-CRAIG with Different Pruning %

The training loss of CRAIG seems to outperform that of F-CRAIG by around 0.06. However, as it will be seen in the testing loss, it seems that F-CRAIG is underfitting instead of giving higher loss.

The testing loss of CRAIG is very similar to F-CRAIG by around. Even for different pruning %, it seems that F-CRAIG is giving similar performance on testing data meaning we can prune more data without any significant increase in loss.

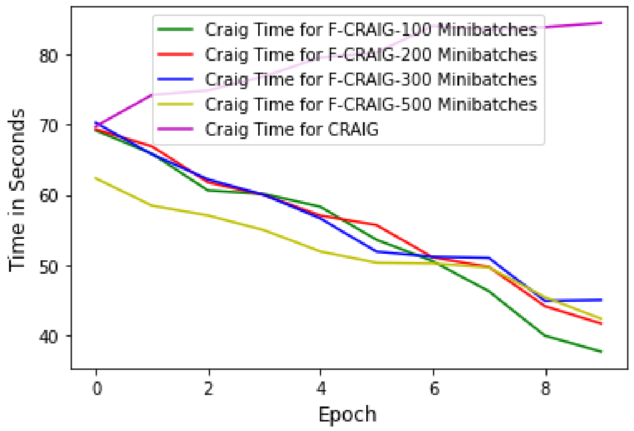


Total Time taken over 10 epochs

Overall, we observe that the time taken for training 10 epochs and 3 runs is almost 20% less in case of F-CRAIG. Also, we observe that the total time reduces for 5% pruning and 7% pruning but then increases for 10% pruning, even though it is counter intuitive.

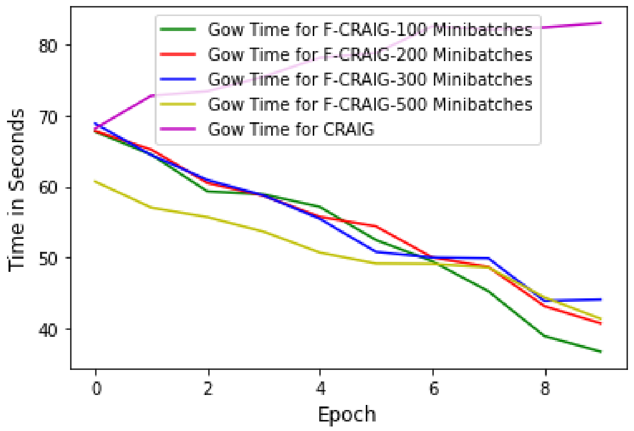
4.2 Forgettability Score Frequency

In this section, we fix the pruning percentage at 5 and the frequency of the forgettability score calculation varies. We experimented for every 100, 200, 300, and 500 mini batches. The same metrices were observed as in the previous case and the time plots are plotted below.



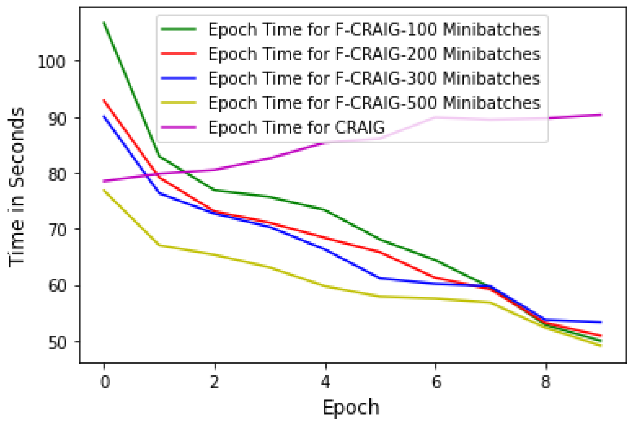
CRAIG Time of CRAIG vs F-CRAIG with Different Minibatch Size

The CRAIG time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. Initially, CRAIG time is same for both approaches, but we see a significant improvement in the last few epochs as a good proportion of data is pruned.



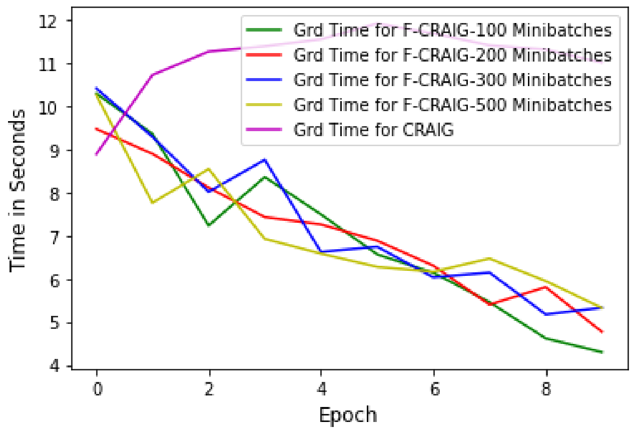
GOW Time of CRAIG vs F-CRAIG with Different Minibatch Size

The subset calculation time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. Initially, GOW time is same for both approaches, but we see a significant improvement in the last few epochs as a good proportion of data is pruned but GOW times remains same for different minibatches.



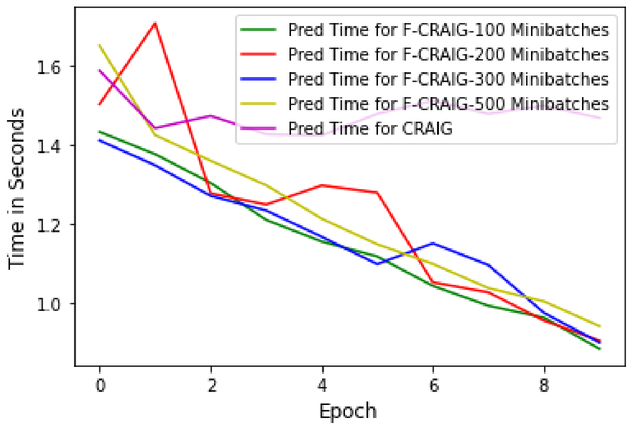
Epoch Time of CRAIG vs F-CRAIG with Different Minibatch Size

The epoch time for first few epochs is higher because additional time is taken to compute the forgettability metrics. However, starting from third epoch we see that epoch time is lesser from F-CRAIG as compared to CRAIG. Also, epoch time remains contant for CRAIG but reduces with epoch in case of F-CRAIG. We can also see that initially the calculating forgetting scores afer 500 minibatches is fastest as less computations are being done but as data gets pruned, the epoch time converges for different F-CRAIG minibatches.



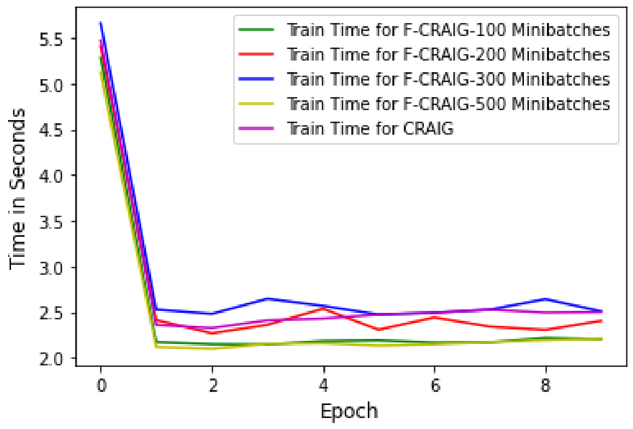
Greedy Time of CRAIG vs F-CRAIG with Different Minibatch Size

The Greedy evaluation time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. Initially, Greedy time is same for both approaches, but we see a significant improvement in the last few epochs as a good proportion of data is pruned but lazy greedy time remains same for different minibatches.



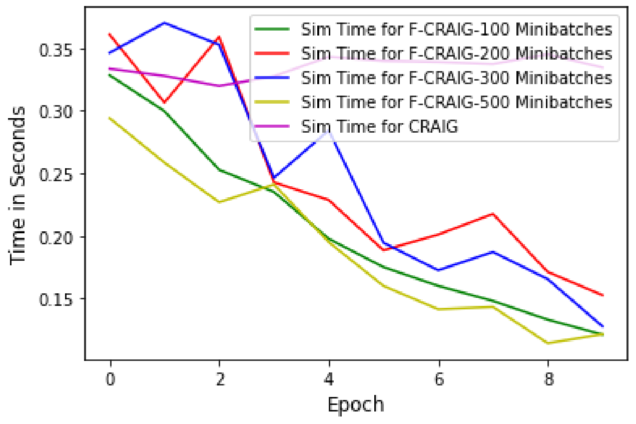
Pred Time of CRAIG vs F-CRAIG with Different Minibatch Size

The Prediction time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. Initially, Prediction time is same for both approaches, but we see a significant improvement in the last few epochs as a good proportion of data is pruned. However, there are no major differences in the prediction time for calculation of after different minibatch sizes.



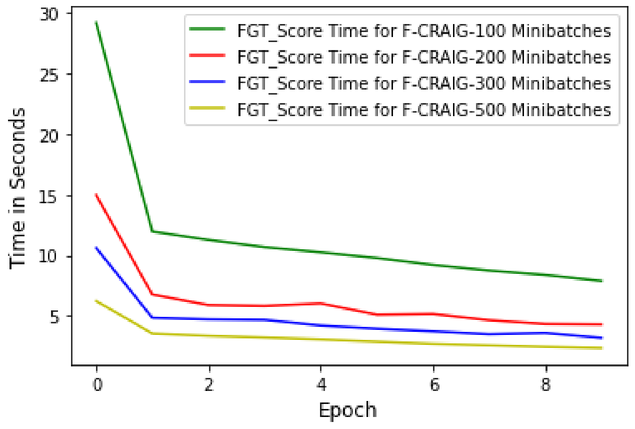
Train Time of CRAIG vs F-CRAIG with Different Minibatch Size

The Training time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. F-CRAIG seems to outperform CRAIG in training time for all different minibatch calculations times except that of 200 minibatches.



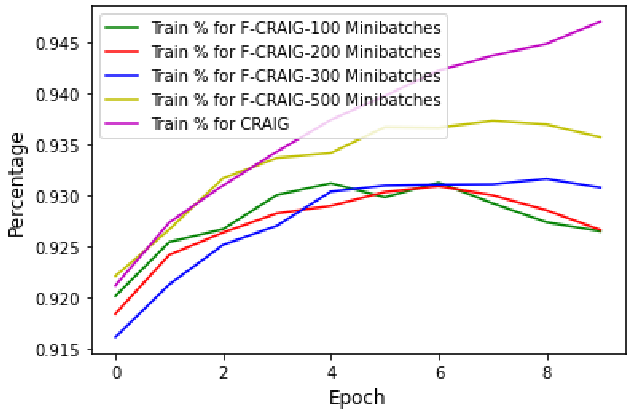
Similarity Time of CRAIG vs F-CRAIG with Different Minibatch Size

The Similarity matrix calculation time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. Initially, Similarity time is same for both approaches, but we see a significant improvement in the last few epochs as a good proportion of data is pruned

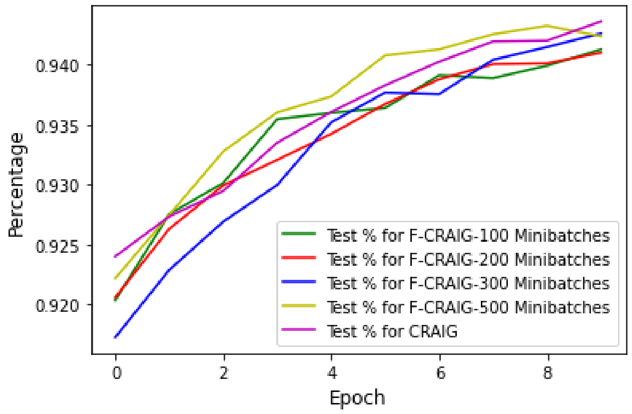


FGT Time of CRAIG vs F-CRAIG with Different Minibatch Size

The Forgetting Scores calculation time reduces for F-CRAIG with epoch as the dataset is getting pruned eventually. FGT_Score time is clearly in an order as calculating forgetting scores after 100 minibatches is clearly slower than the other minibatch calculation frequencies



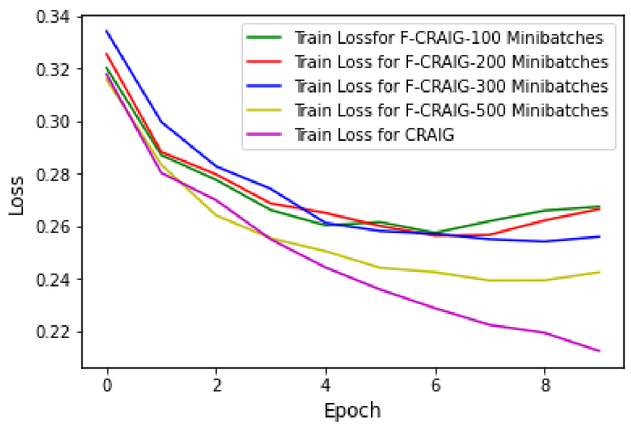
Train Accuracy of CRAIG vs F-CRAIG with Different Minibatches



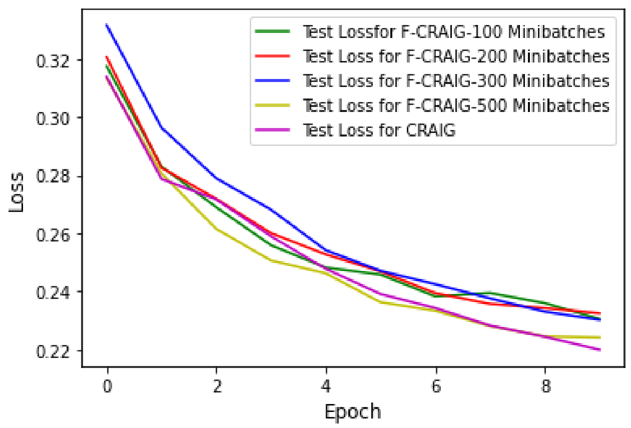
Test Accuracy of CRAIG vs F-CRAIG with Different Minibatches

The training accuracy of CRAIG seems to outperform that of F-CRAIG by around 2%. However, as it will be seen in the testing accuracy, it seems that F-CRAIG is underfitting instead of giving lower accuracy

The testing accuracy of CRAIG is very similar to F-CRAIG by around. Even for different minibatch frequencies, it seems that F-CRAIG is giving similar performance on testing data meaning we can prune more data without any significant drops in accuracy



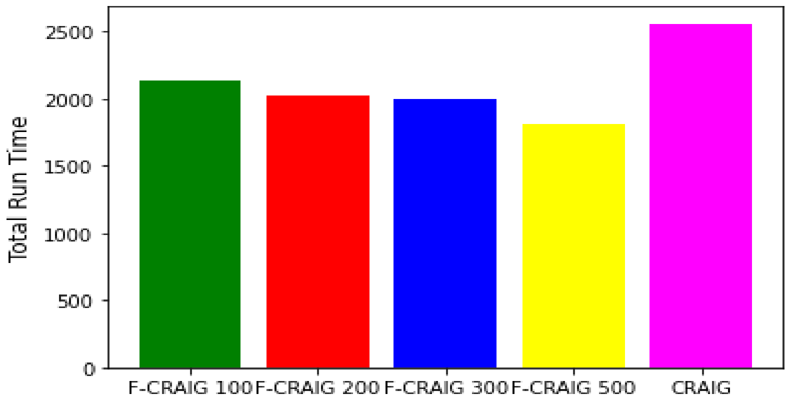
Train Loss of CRAIG vs F-CRAIG with Different Minibatche s



Test Loss of CRAIG vs F-CRAIG with Different Minibatches

The training loss of CRAIG seems to outperform that of F-CRAIG by around 0.08. However, as it will be seen in the testing loss, it seems that F-CRAIG is underfitting instead of giving higher loss.

The testing loss of CRAIG is very similar to F-CRAIG by around. Even for different minibatch frequencies, it seems that F-CRAIG is giving similar performance on testing data meaning we can prune more data without any significant increase in loss.



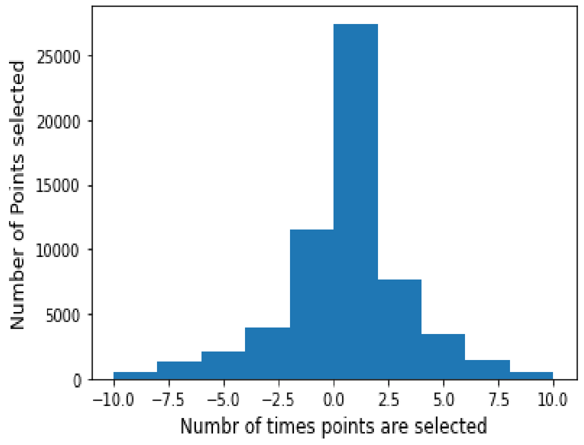
Total Time taken over 10 epochs

Overall, we observe that the time taken for training 10 epochs and 3 runs is almost 20% less in case of F-CRAIG. Also, we observe that the total time reduces in F-CRAIG as the number of forgetting score calculations decreases as Minibatch frequency increases.

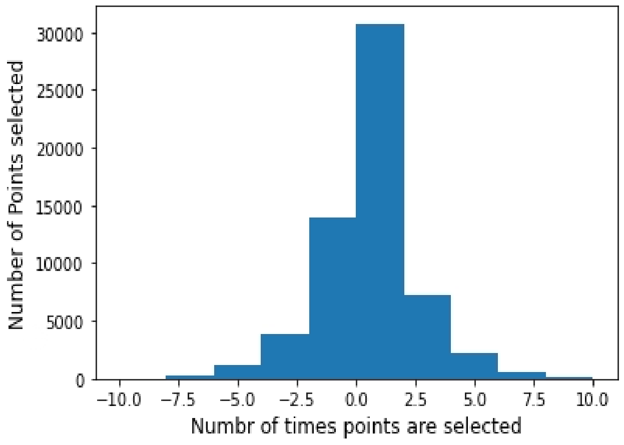
4.2 Comparison of Data Selection

We compared the points selected by CRAIG in two runs to observe the stochasticity in the method. From the left plot, we can see that most of the points remain same as the histogram peaks at 0. However, there is some stochasticity in the process as observed by the tails of the histogram.

The idea was to check if the points selected from F-CRAIG and CRAIG are same which means that we are not dropping points which are of high importance in the training. So, we plotted a similar histogram to see the differences between F-CRAIG and CRAIG.



F-CRAIG VS CRAIG



CRAIG RUN 1 vs CRAIG RUN

We observe that the left and the right plots are similar. This helps us conclude that F-CRAIG was faster when compared to CRAIG and F-CRAIG is not removing important point that are being selected for back-propagation.

5. Conclusion

The CRAIG time, Epoch time, GOW time, Grd time, Pred time, and Sim time stays nearly same for every epoch in CRAIG. But these run times decrease with epochs for F-CRAIG. This is because the data gets trimmed, and the time taken for these computations reduces. Also, the additional time for forgettability scores computation and pruning decreases as epoch number increases, but it is around 15 seconds. In terms of performance, CRAIG performs slightly better in the training data, but no significant difference can be observed in the test dataset.

The additional time taken for data pruning using forgettability scores increases as the frequency of its calculation is increased. This is clearly because forgettability scores are calculated lesser number of times. However, we do not observe any significant differences for different pruning ratios and forgettability score frequencies.

F-CRAIG is faster than CRAIG and takes approximately 20% lesser time. Overall, we can conclude that the points pruned by F-CRAIG doesn't get selected in CRAIG and we achieve a speedup without compromising on the performance.

References

[1] Strubell, E., Ganesh, A., and McCallum, A. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243, 2019*

[2]Asi, H. and Duchi, J. C. The importance of better models in stochastic optimization. *arXiv preprint arXiv:1903.08619, 2019.*

[3] Qian, N. On the momentum term in gradient descent learning algorithms. *Neural networks, 12(1):145–151, 1999.*

[4] Asi, H. and Duchi, J. C. The importance of better models in stochastic optimization. *arXiv preprint arXiv:1903.08619, 2019.*

[5] Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980, 2014*

[6] Defazio, A. and Bottou, L. On the ineffectiveness of variance reduced optimization for deep learning. *In Advances in Neural Information Processing Systems, pp. 1755–1765, 2019*

[7] Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. *In Advances in neural information processing systems*, pp. 315–323, 2013.

[8] B. Mirzasoleiman, J. Bilmes, and J. Leskovec. "Coresets for data-efficient training of machine learning models". *In: Int. Conf. Machine Learning (ICML)*. PMLR. 2020, pp. 6950–6960. arXiv: 1906.01827.

[9] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. An empirical study of example forgetting during deep neural network learning. *In 7th International Conference on Learning Representations, ICLR 2019*.