Faster Training by Automatically Selecting the Best Training Data for Computer Vision Tasks

Tony Cai, Dennis Duan, and Ananth Agarwal
6/4/2022
Motivation: Increasingly Expensive Model Training

AlexNet to AlphaGo Zero: A 300,000x Increase in Compute (Log Scale)

Previous Approaches

- **Grad-Match**: Periodically choose subsets of training data to train on

- **Selective-Backprop**: Only perform backpropagation on high-loss examples

---

**Image Credits:**


Our Contributions

- Key observation: Most often, model training occurs multiple times, for example for hyperparameter and model search

- Can we use training statistics from prior training runs to help train models on the same dataset more efficiently with minimal loss in performance?
Our Contributions

- For the image classification task:
  - Reproduce the training speedups reported by Grad-Match on the standard CIFAR-10 dataset, evaluate it against a new random selection baseline, and propose and evaluate a variant that reuses subsets chosen from previous training runs.

- For the object detection task:
  - Evaluate if the benefits of Selective-Backprop translate to object detection on the popular MS COCO dataset
Image Classification - Methods

- Reproduce the training speedups reported by Grad-Match on the standard CIFAR-10 dataset, **evaluate it against a new random selection baseline**, and propose and evaluate a variant that reuses subsets chosen from previous training runs.
  - Random subset selection: choose random examples to fill budget
  - **RandomPB**: divide train set into fixed batches, then randomly choose batches
Image Classification - Methods

- Reproduce the training speedups reported by Grad-Match on the standard CIFAR-10 dataset, evaluate it against a new random selection baseline, and **propose and evaluate a variant that reuses subsets chosen from previous training runs**.
  - **Cached-Grad-Match**: run an initial training run with Grad-Match, and on subsequent runs “replay” the same subsets selected.
    - Saves computation time, but how robust is it?
Image Classification - Experiments

- Cached-Grad-Match
  - Reuse Grad-Match’s data selection on ResNet training for different learning rates
  - Reuse Grad-Match’s data selection on ResNet training for optimizers
  - Reuse Grad-Match’s data selection on MobileNetV2 training for ResNet training

- Baseline
  - RandomPB
  - Training on full dataset
# Image Classification - Fixed Accuracy Speedup

<table>
<thead>
<tr>
<th>Learning rate</th>
<th>0.001</th>
<th>0.003</th>
<th>0.03</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GRAD-MATCH</strong></td>
<td>0.81</td>
<td>0.82</td>
<td>2.72</td>
</tr>
<tr>
<td><strong>CACHED-GRAD-MATCH</strong></td>
<td><strong>1.00</strong></td>
<td><strong>0.99</strong></td>
<td><strong>7.92</strong></td>
</tr>
</tbody>
</table>

Table 1. Fixed-accuracy speedups over full training using different learning rates.

<table>
<thead>
<tr>
<th>Optimizer</th>
<th>Adam</th>
<th>RMSProp</th>
<th>SGD</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GRAD-MATCH</strong></td>
<td><strong>0.83</strong></td>
<td>0.81</td>
<td>3.07</td>
</tr>
<tr>
<td><strong>CACHED-GRAD-MATCH</strong></td>
<td>0.67</td>
<td><strong>0.97</strong></td>
<td><strong>4.68</strong></td>
</tr>
</tbody>
</table>

Table 2. Fixed-accuracy speedups over full training using different optimizers.

<table>
<thead>
<tr>
<th>Model Architecture</th>
<th>Speedup on ResNet</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GRAD-MATCH</strong></td>
<td>3.86</td>
</tr>
<tr>
<td><strong>GRAD-MATCH-warm</strong></td>
<td><strong>4.69</strong></td>
</tr>
<tr>
<td><strong>CACHED-GRAD-MATCH</strong></td>
<td>2.24</td>
</tr>
<tr>
<td><strong>CACHED-GRAD-MATCH-warm</strong></td>
<td>4.51</td>
</tr>
</tbody>
</table>

Table 3. Fixed-accuracy speedups over full training for different model architectures.
Image Classification - Training Speed vs Error

Figure 1. Speedup-accuracy tradeoffs of subset selection algorithms using different learning rates.

Figure 2. Speedup-accuracy tradeoffs of subset selection algorithms using different optimizers.

Figure 3. Speedup-accuracy tradeoffs of subset selection algorithms for different model architectures.
Image Classification - Results

Low-priority images

High-priority images

Priority histogram
Object Detection - Methods

- Single Shot Multibox Detector (SSD)
  - ResNet backbone + conv layers + classification, regression heads

![Image of object detection](image)

- Loss Function

\[
L_{obj}(x, c, l, g) = \frac{1}{N} \left( L_{conf}(x, c) + \alpha L_{loc}(x, l, g) \right)
\]

Object Detection - Methods

- Selective-Backprop
  - Combine Jiang et al. original approach with MosaicML suggestions

- Nvidia SSD PyTorch library: ResNet backbone, input size 300x300
  - Integrates with COCO’s Python library

Algorithm 1 SELECTIVE-BACKPROP for Object Detection

1: for epoch in range(epochs) do
2:   for i, $X_{b\text{size}}$ in enumerate(data_loader) do
3:     if epoch in [sb_start, sb_end] then
4:       losses = forward($X_{b\text{size}}$)
5:     if i % interrupt == 0 then
6:       backward(losses)
7:     else
8:       $X_{s} = P_{select}$(losses)
9:       backward(forward($X_{s}$))
10:   end if
11: else
12:   backward(forward($X_{b\text{size}}$))
13: end if
14: end for
Object Detection - Methods

- Evaluation Metrics
  - Total Training Time
  - Mean average precision: mAP [0.50:0.95]
    - All object sizes
    - Small
    - Medium
    - Large

- Intersection over Union (IoU):
Object Detection - Experiments

1. Baseline: No Selective-Backprop
2. Selective-Backprop, interrupt = 2
3. Selective-Backprop, interrupt = 3

- Common parameters:
  - Batch size = 32
  - Epochs = 30

- Selective-Backprop (variables defined in Alg. 1)
  - Start (sb_start) = 0.5
  - End (sb_end) = 0.9
  - Keep (s) = 0.5
The AWS EC2 instance was stopped and restarted between running SB3 and the baseline and SB2, so the SB3 absolute training time is not directly comparable to the other two due to potential GPU environment differences.

<table>
<thead>
<tr>
<th>mAP IoU 0.50:0.95</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>All</td>
</tr>
<tr>
<td>Base</td>
<td>9.68</td>
</tr>
<tr>
<td>SB2</td>
<td>9.09</td>
</tr>
<tr>
<td>SB3</td>
<td>8.79</td>
</tr>
</tbody>
</table>

$t_{diff} = \frac{\bar{t}_i}{\forall i \in [1, \text{epochs}]} - \frac{\bar{t}_j}{\forall j \in [1, \text{epochs}] \setminus \{\text{start, end}\}}$
Object Detection - Discussion

- Speedup, but accuracy lost

- Alg. 1 has a visible effect on model parameters in one layer of the ResNet50 backbone:
Future Work

- Image Classification
  - Convergence analysis
  - Other data selection heuristics

- Object Detection
  - Multiple re-runs, Hyperparameter sweeps
  - Larger batch size
  - Grad-Match and Cached-Grad-Match
Conclusion

- Found Grad-Match speedup for image classification, but its performance varies substantially with different hyperparameters.

- Reusing prior training (Cached-Grad-Match) shows performance gains without extra cost of adaptive data selection.

- Benefit of data selection is less pronounced for object detection.
References


Krishnateja Killamsetty, Dheeraj Bhat, Ganesh Ramakrishnan, and Rishabh Iyer. CORDS: COResets and Data Subset selection for Efficient Learning, 3 2022.


Nvidia. SSD300 v1.1 For PyTorch, 3 2019.