**[Mixed Precision Training]**

1. Introduction

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| This | paper | proposes | a | method | for | maintaining | model | accuracy | using | reduced | precision | |
| representations and arithmetic in deep learning. Specifically, we introduce a method to train different | | | | | | | | | | | | |
| neural networks using the FP16 format, and introduce three techniques to maintain a master copy | | | | | | | | | | | | |
| of the weights with FP32, apply loss scaling to ensure that the gradient value is non-zero, and | | | | | | | | | | | | |
| accumulate FP16 arithmetic in FP32. Using these techniques, we experimentally show that we can | | | | | | | | | | | | |
| train different neural network structures and applications with equivalent accuracy to FP32. | | | | | | | | | | | |  |

1. Related Work

|  |  |
| --- | --- |
| We analyze studies that train CNNs and RNNs with reduced precision in various ways. We propose | |
| a novel method to maintain accuracy using the FP16 format, which introduces three aspects that | |
| distinguish it from existing methods: it uses reduced precision for all tensors and arithmetic, it trains | |
| without hyperparameter adjustment, and produces an accuracy-free model. Experimental results | |
| show that the technique works effectively on large datasets with state-of-the-art models in various | |
| applications. |  |

1. Implementation
   1. FP32 Master copy of weights

|  |  |
| --- | --- |
| In this section, we maintain and update a copy of FP32 master weights to maintain the accuracy of | |
| the FP32 network while using weights, activations, and gradients stored with FP16. The experimental | |
| results match the results of FP32 training when updating the copy of FP32 weights, but updating | |
| the FP16 weights reduces the accuracy by 80%. Additional copies of weights increase the memory | |
| requirement by 50%, but their impact on overall memory usage is small. |  |

* 1. Loss Scaling

|  |  |
| --- | --- |
| Loss scaling of FP16 increases model accuracy by extending the gradient value and keeping it within | |
| the FP16 expressible range. This preserves the critical values of the network and prevents the | |
| gradient value from becoming zero. To avoid overflow, we inspect the weighted gradient and skip | |
| the weight update when overflow is detected. |  |

* 1. Arithmeric Precision

While neural network arithmetic uses FP16 tensors in memory, arithmetic is performed on FP32.

Point wise operations are limited to memory bandwidth and are not affected by arithmetic precision,

so both FP16 and FP32 mathematics can be used.

1. Results
   1. CNNs for ILSVRC Classification

Several CNNs were trained with mixed precision on the ILSVRC classification task. For several models, we were able to achieve the same accuracy as the baseline FP32 training.

* 1. Detection CNNs

Object detection is a regression task, where the network predicts bounding box coordinates. The SSD detector failed to train on FP16 without loss scaling, but achieved similar performance to FP32 through loss scaling.

* 1. Speech Recognition

We explore the results of mixed-precision training with DeepSpeech 2 model on speech data. We find that the semi-precision storage format can play a role in normalizing the training.

* 1. Maching Translation

For language translation, we trained a model of the TensorFlow tutorial from English to French with several variations. The mixing precision with loss scaling matched the FP32 results, but the results were slightly degraded when loss scaling was not used.

* 1. Language Modeling

We trained a bigLSTM English language model on a billion-word dataset. We trained with FP16 using loss scaling to match it with FP32.

* 1. DCGAN Results

DCGAN is a generative adversarial neural network for image generation. The outputs of FP32 and mixed precision training look similar.

1. Conclusions and Future Work

Mixed precision training is an important technique to reduce memory consumption and task time for deep learning models. Using this technique, we demonstrate that different deep learning models can be trained without loss of accuracy. In the future, studies will be conducted on generative models and automated loss scaling element selection.