Review on “In-Place Activated BatchNorm for Memory-Optimized Training of DNNs”

Authors: S. Bulò, L. Porzi, P. Kontschieder,

# Short Summary

In this paper, the authors explore inefficiencies with batch normalization (BN) and activations during training in deep neural networks (DNNs). The key issue is that BN and activation output values calculated on the forward pass need to be stored for re-use in the backwards pass. Therefore, there is an inherent trade-off between memory and speed.

One pre-existing solution is Checkpointing which only stores the input into the BN layer in the forward pass and then re-computes the input to the activation on the backwards pass. This introduces some computation overhead at the expense of requiring less memory for each BN and activation block. In this paper, the authors point out that storing the output of the batch normalization, as opposed to the input, in the Checkpointing method will improve computation speed at little cost.

They then propose an alternative to the checkpointing method: a new layer called In-Place Activated Batch Normalization (Inplace-ABN) and a variation. Both can be used as an end-to-end replacement for existing activation and batch normalization layers. The original Inplace-ABN proposes that it is better to store the output of the activation layer during the forward pass, which can then be inverted during the backwards pass. Although conceptually similar to checkpointing, it is shown that the gradient calculation requires less parameters and thus this solution uses less memory. The variant rewrites the gradient calculations be taken with respect to one of the intermediary outputs which reduces computation complexity.

Due to the inversion in Inplace-ABN, it is important that an invertible activation function such as Leaky RELU be used in place of its popular variant RELU. The paper shows that such a change does not have a detrimental effect on performance when used with the ResNeXt and WideResNet architectures.

Inplace-ABN is ultimately shown to free almost 50% of the memory associated with the base implementations of batch normalization and activations with minimal computational overhead.

# Main Contributions

* A computational efficiency improvement to Checkpointing
* A self-contained Inplace-ABN layer that halves DNN memory requirement during training
* Experiments on image classification and semantic segmentation showing how additional memory can be utilized to improve model performance

# High-Level Evaluation of Paper

The authors give comprehensive insight in the problem that the paper is intending to tackle. The explanation of checkpointing also made it easier to understand how their proposal takes a novel approach to the issue. However, it’s not clear whether there are solutions other than checkpointing that tackle the issue. The change proposed to improve checkpointing is very briefly discussed and not well-developed in terms of quantifying any performance benefits. This may be because the proposal is more akin to an implementation detail than a novel idea. Figure 2 on the third page of the paper provides high-level overview of their proposals and the original model. This comparison is particularly effective because it is clear how model is iteratively improved with each change. Furthermore, the authors pre-emptively identified the weaknesses in their proposal (invertibility & zero values) and demonstrate that solutions to these newly introduced problems do not impact performance.

The final section in the paper focusing on demonstrating how the additional GPU memory, obtained via implementing the Inplace-ABN, can lead to better performance. While the additional memory is a benefit of their proposal, I didn’t feel that experimenting with how that memory is used was relevant to the overall focus of the paper. Furthermore, it is already well-established that deeper networks and larger batch sizes can show performance gains.

# Discussion on Evaluation Methodology

While most papers use mean average precision (MAP) or intersection over union (IoU) to quantify image classification results, it seems that the authors of this paper opted for using regular accuracy for all of their experiments instead. I’m not sure how semantic segmentation is quantified using accuracy.

While experiments are well designed and performed on multiple distinct datasets, they are seldom compared to external baselines. A notable exception is the case where the model exceeded the performance of the LSUN 2017 winner achieving state of the art performance. I think the omission of comparing to other state-of-the-art is purposeful in that the focus of this paper is to explore the speed versus memory trade-offs. With that said, charts quantifying the difference in memory utilization for the different models are missing the chart that describes computational overhead omits Inplace-ABN II. Personally, I would have liked to see more emphasis on this aspect of the paper as it seems to be more relevant to the major contribution.

# Possible Directions for Future Work

A possible direction for future works is to implement the Inplace-ABN layer in popular network architectures to investigate better quantify memory versus speed trade-offs.