Batch normalization and only Batch normalization

August 9, 2022

Alessandro Torri

Abstract

In this report I will analyze the results obtained by training only batch normalization and freezing all the other model parameters at their random initialization. All the code regarding this project can be found here: https://github.com/Alessandro1999/BatchNormOnly

1. Introduction

The aim of this project is to investigate the differences in terms of performances and behaviour of models that are trained in the standard way and models with all the parameters freezed at their random initialization except for the batch normalization ones.
In order to see that, I have trained a sequence classification model that uses a LSTM(Hochreiter & Schmidhuber, 1997) to perform sentiment analysis and a standard MLP to perform image classification. However, it is important to say that the focus of this project was to see the differences between the two kinds of training described before rather then optimize these networks to achieve SOTA performances.

2. Related works

This project takes strongly inspiration from the work of Frankle, Schwab and Morcos(Frankle et al., 2020), that tested the idea of training only batch normalization on different ResNet CNN trained on CIFAR-10, CIFAR-100 and ImageNet. Indeed, in order to explore the ideas of their work, I have tried to test them on a different architecture like MLP (even if not suitable as CNNs for image classification) and on a completely different kind of task like sentiment analysis, where we deal with text data.

Email: Alessandro Torri <torri.1835715@studenti.uniroma1.it>.

Deep Learning and Applied AI 2022, Sapienza University of Rome, 2nd semester a.y. 2021/2022.

3. Method

In this section, I will describe more in detail the experiments that I have tried.

3.1. Sentiment analysis

The task of sentiment analysis has been tackled with the dataset that can be found here; A binary classification (positive/negative) of tweets has been performed with a training set of 90k tweets, a validation set of 10k and a test set of 25k. The model architecture is composed of:

- An embedding layer, in which the words are converted in word embeddings; the words that have their own word embedding are the ones that appear at least twice in the training set whereas the others have associated an Out of Vocabulary word embedding.
- A LSTM that analyzes sequentially the word embeddings and from which the last hidden state is taken.
- A linear layer followed by a softmax to obtain the probabilities for the two classes.

Between every component of the network there is a batch normalization layer; the network has a total of 5.5M parameters, while the batch normalization ones are just 604 (only the 0.01%!).
For this task, I have tried to see the behaviour of batch norm only both in the case of word embeddings learned from scratch and fine-tuned word embeddings starting from GloVe(Pennington et al., 2014) pre-trained ones (as a standard practice of NLP to start with pre-trained layers).

3.2. Image classification

As for the image classification task, I have used the CIFAR-10 dataset with a MLP with two hidden layers and batch normalization between every layer. The model has a total of 3.8M parameters while the batch normalization ones are just 6.2K (the 0.16%).
Of course, since we are using a MLP the images of the dataset are linearized into a one-dimensional vector.

4. Results

It is very interesting the fact that the findings of the article of (Frankle et al., 2020) seem to be confirmed by my results; indeed, training only batch norm leads to surprisingly high results (considering the small number of trainable parameters), not so distant from the ones achieved with all parameters.

4.1. Sentiment analysis

The first attempt I tried was with SA (sentiment analysis) and I observed the following results:

Figure 1. As we can see, training all parameters leads to 78% accuracy while training only batch norm (as we said, only the 0.01% of the parameters) leads to 67% accuracy. Of course, since we have way less parameters with batch norm only, the model requires a lot more training epochs, that are by the way faster than the epochs of the complete model.

Anyway, it is a standard practice in NLP to start with pre-trained word embeddings, in order to have words in a space that is meaningful of their meaning and/or context; that's why I decided to run a second experiment, in which the only difference is that the model word embeddings are not initialized randomly, but with pre-trained GloVe embeddings, and here are the results:

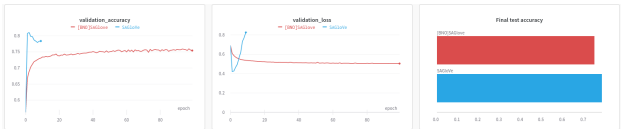

Figure 2. Differently from before, the network with batch norm only achieves 74% accuracy, while training all parameters still leads to 78%; so here we have only a difference of 4%!

4.2. Image classification

For image classification on CIFAR-10, we observe the same difference (around 9% points) as SA without GloVe:

Figure 3. The MLP has 54% accuracy while training only batch norm gives around 45%.

4.3. γ and β distributions

What is really interesting is how the distribution of the γ and β parameters changes when training all parameters and when training only batch norm. Indeed, the difference in these distributions is a clear hint on how training only batch norm achieves these high results; in particular, by looking at the gamma distribution, we can observe how it basically performs feature selection over the random features and it does so by having a lot of γ values set to 0.

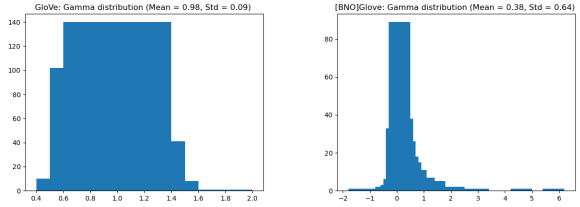

Figure 4. On the left the γ distribution of the SA network (with GloVe) when all parameters are trained while on the right the γ distribution of the same network with batch norm only.

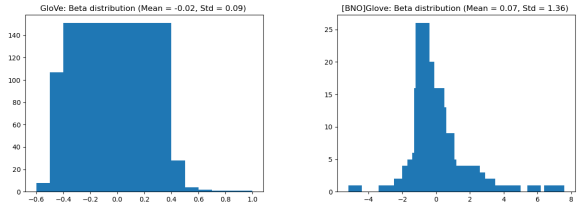

Figure 5. As in figure 4, here we can see the β distribution of the network.

As seen in figure 4, the γ distribution has a peak at 0 when training only batch norm (actually also the β distribution), meaning that when there is a big enough number of parameters (even if they are at their random initialization) batch norm is able to learn which of them to select in order to obtain the best performances.

5. Conclusions

We have seen how the findings of (Frankle et al., 2020) seems to be confirmed also on other task and architectures; Indeed, I showed how the batch normalization parameters seems to be particularly expressive and alone they can achieve surprisingly high results (not so distant from the ones achieved training all the model). For future research, it might be interesting to see if their expressive power is simply given by the pre-activation shifting and rescaling of the features or if it is also important the batch component; for example, would Instance normalization lead to the same results?

References

Frankle, J., Schwab, D. J., and Morcos, A. S. Training batchnorm and only batchnorm: On the expressive power of random features in cnns, 2020. URL https://arxiv.org/abs/2003.00152.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pp. 1532–1543, 2014.