Sleepbot - examining generalization error

Dhasharath Shrivathsa, Yuzhong Huang
Computational robotics - Spring '17

Abstract

Sleepbot is our project for improvements in generalization error for convolutional neural networks. Inspired by neuroscience, we aimed to build heuristics to speed up training or improve test set performance. We built visualization, logging, graphing, and statistics for a CNN to this effect. The structure of our code allows training processes to be transferred to some extent, through training a secondary network around the training process, though we did not thouroughly investigate this.

Problem framing

Generalization error in Neural networks

Background Initially inspired by neuroscience[2], deep learning have achieved great results in various research area including Computer Vision, Natural Language Processing and Robot Controls. Driven by the demand for high speed performance, most deep learning frameworks are confined to a layer-based form, which is efficient and compatible with the current computing system.[3] There is merit to changing this architecture, as evidenced by [1], and one way is to take a critical look at backpropogation.

The training process -
Convolutional neural networks are trained by

1. Feedforward a batch of images (a tunable "chunk" of the dataset)
2. Backpropogate the errors of the collective batch
3. Repeat for n epochs

In the middle is the most important step, what is considered "training". Backpropogation calculates the error magnitude for each neuron and updates it's weights in closed form. Our method intercepts weights in between training batches, so between each step of backpropogation.

Backpropogation What is wrong with backpropogation? It is a formative algorithim, used to propogate weight update throught the network. Unfortunatley, its biggest strength is also its biggest weakness. A closed form iterateve solution cannot observe time-series steps, and tends to overfit or maintain the current trend (see $>$ batch 30 in Fig. 2), as it only is concerned with error.

Proposed solution Looking at the time-series structure for how the weights of neurons evolve seems like a good proxy for speeding up training. Taking this further can mean training a LSTM-RNN on the layer weights for an approach that appears to be like [7],or [6]. In this work, we only propose simple numerical heuristics as to how to speed training and improve generalization error.

Visualization of performance

Visualization of the training process Given the inherent structure to convolutional neural networks, there exist several ways to visualize what the process looks like. We chose two, the image formed as the neural net is shown all 1's (the magnitude of the contribution of each neuron), and the norm of their weights through time.

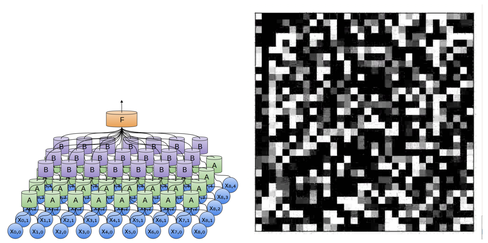

Figure 1: Graphical intuiton and arrangement in a traditional convolutional neural network (CNN). In this, A represents a convolution, B an aggregation/downsampling layer such as a softmax, and F is a fully connected neuron. The right figure is a net trained on MNIST's first layer activation, on an image of all max values. We can see low-level features arising (groups/lines of pixels).

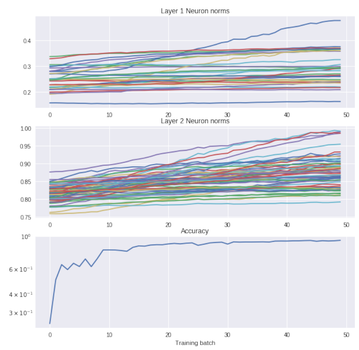

Figure 2: A depiction of neuron weights in time, as training progresses. We see a strong tend towards limit trend in the norms of the neurons as training progresses.

Trends in neuron weights As we view the neurons training, we can see that each has a predetermined distribution across which this is spread. The general widening trend with little reorgainization implies training is done and the network has reached an equilibrium state, with only miniscule incremental gains in performance. However, early identification of the asymptotic limit is a possible heuristic that we can use to speed up training. Another possible metric is the delta in relative spread across the neuron norms. This manifests in the graph as the ratio of the norms staying roughly equal as the network saturates.

Performance evaluation of heuristics

Baseline performance and success measures In Fig 3, we can see what a standard no-sleep run looks like, as well as one that has been corrupted by noise. The down-and-right nature of the random noise curve is indicitave of worse performance, and the "fuzz" along the trained network is random noise being injected, and the network compensating with another step of backpropogation. Geometrically speaking, it is easy to see what success looks like, relative up-and-left shift of the line.

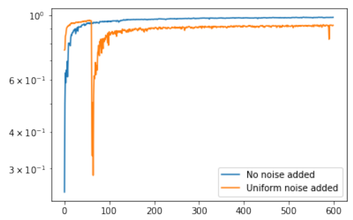

Figure 3: Accuracy on the test dataset for number of batches trained. Comparing random noise in the system (uniform in the $-1 \leq x \leq 1$) to the system with no interruptions in backpropogation

The spike at the beginning deserves mention, as it's a lucky configureation found by chance. We can see it quickly drop off as the opposite happens, a very unlucky sleep operation.

Ideally, we're able to replicate the early boost given by a random sleep (in this case).

Results

We started out with an intent to simulate and implement a sleep phase for deep learning, during which the model takes no external data and change its weight through some signals in the system. As we researched on sleep in neuroscience[5], we found that sleep is a very complex havior that influences mainly the chemical transmission tween neurons[4]. It is very hard to generalize and simul a sleep phase with a single signal neural network system

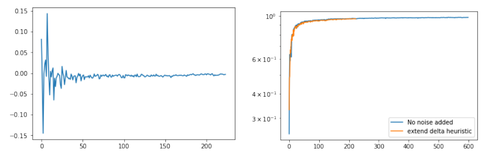

Figure 4: Most of our heuristics looked like this (delta)

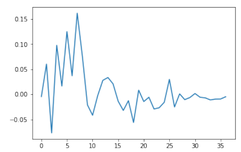

Figure 5: Some of our heuristics (here `delta = last_wts - current_wts; new_wts = current_wts + delta**2`) show promise, improving the training r by 'boosting' the initial process with a square, that roun lower as the training cycle goes on.

Further work

We still have to try transfer learning with a LSTM-RNN, a more heuristsics are always good to try. Now that we ha a toolkit, trying out new things is easy, so we should it ate and try more heuristics. Sleep in a neural network i challenging thing.

Bibliography

References

[1] Nicholas JA Harvey, Michael B Jones, Stefan Saroiu, Marvin Theimer, and Alec Wolman. Skipr scalable overlay network with practical locality properties. *networks*, 34(38), 2003.

[2] J Allan Hobson and Edward F Pace-Schott. The cognitive neuroscience of sleep: neuronal sys consciousness and learning. *Nature Reviews Neuroscience*, 3(9):679–693, 2002.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep con tional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2

[4] CHRISTOPHER A Ross, DAVID A Ruggiero, DONG H Park, TONG H Joh, ALAN F Sved, J Fernandez-Pardal, JUAN M Saavedra, and DONALD J Reis. Tonic vasomotor control by the r ventrolateral medulla: effect of electrical or chemical stimulation of the area containing c1 adrer neurons on arterial pressure, heart rate, and plasma catecholamines and vasopressin. *Journal of roscience*, 4(2):474–494, 1984.

[5] Carlos H Schenck and Mark W Mahowald. Rem sleep behavior disorder: clinical, developm and neuroscience perspectives 16 years after its formal identification in sleep. *Sleep*, 25(2):120,

[6] Zhiyuan Tang, Dong Wang, and Zhiyong Zhang. Recurrent neural network training with knowledge transfer. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International ference on*, pages 5900–5904. IEEE, 2016.

[7] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv pr arXiv:1611.01578*, 2016.