Simple Baselines for Image Recognition
Group 76
COMP 551

Han Wen Xie, David Kang, Marie-Eve Malette-Campeau
{han.xie, woo.kang}@mail.mcgill.ca
marie-eve.malette-campeau@umontreal.ca

Abstract

In this project, we implement several baseline models for the task of correctly classifying a tiny image into one of ten predefined labels. We compared the test accuracies of baselines against the test accuracy of the Teacher network baseline reported in (Srivastava et al., 2015) on the CIFAR-10 dataset: 88.32%. Among the baseline models, we found that the Deep CNN with ZCA whitening and data augmentation gave us the best test accuracy of 88.10%. After empirically tuning hyperparameters of the architecture, the best Deep CNN has six convolutional layers and dropout rate set to 0.4.

1 Introduction

State of the art machine learning models are within a percent or two of human performance on image recognition tasks. In this report, we take simple baseline image classification models and compare their performance with baseline performances reported in Highway Network (Srivastava et al., 2015) on the CIFAR-10[1] image classification dataset. We validated the baseline models on the simpler MNIST[2] dataset.

Our main task is to correctly classify (against ground truth label) a tiny CIFAR image into one of ten predefined labels. Our implementation of Maxout CNN model is a baseline reported in Highway Networks, known as the Teacher Network (Srivastava et al., 2015). For the MNIST dataset, our task is to recognize a digit from MNIST images, each image containing a single handwritten digit.

We trained many simple baselines using the entire training set and evaluated models based on test accuracy. We explored and implemented 3 simple neural network variants: standard Multi-Layer Perceptrons (MLP), Convolutional Neural Networks (CNN), and another CNN using Maxout activation function (Goodfellow et al., 2013). For the CNN models, we explored two types of depths (shallow or deep). Training hyperparameters were specified in a configuration file, and models were evaluated using 0.8-0.2 training-validation split and a test set. Among the 3 variants, we found that a deep CNN gave us the best results, comparable to the Teacher network baseline of (Srivastava et al., 2015). All experiments except the CIFAR-10 Deep CNN used PyTorch[3] framework (Paszke et al., 2017); the CIFAR-10 Deep CNN used Keras[4] framework.

The most important findings of this report are as follows: the Deep CNN baseline with ZCA whitening and data augmentation preprocessing on CIFAR-10 gave us the best test accuracy of 88.10%. We empirically verified that deep standard CNNs improved performance, as compared to a shallower Maxout CNN. Second, we found that max number of epochs hyperparameter had a great impact on the practical utility of deep networks. The validation loss eventually plateaus out and optimizing the cut off point of the epochs hyperparameter can save significant amount of time for training.

2 Related Work

The field of deep learning in image recognition has witnessed major advances in the last few years. In this section, we discuss relevant papers we used to implement our models and what we learned from them.

[1] https://www.cs.toronto.edu/~kriz/cifar.html
[2] http://yann.lecun.com/exdb/mnist/
[3] https://pytorch.org/
[4] https://keras.io/

2.1 Highway Networks

(Srivastava et al., 2015) is one of the state of the art neural network in image recognition, within 2 percent of human recognition accuracy. Highway Networks is a CNN variant that makes training easier by adding additional non-linear transform units per layer. Using the added units, the CNN variant can be successfully optimized at various depths, whereas standard CNN training is bottlenecked for a deep network. Our simple baselines are attempting to beat the performance of this complex, novel architecture.

2.2 FitNet

FitNet is a baseline model discussed in (Srivastava et al., 2015). (Romero et al., 2014) uses a pair of wide, shallower (teacher) and fit, deeper (student) neural networks; FitNet is the latter neural network, which is trained by using hints from the shallower teacher network. FitNet outperforms the teacher network, whilst using significantly fewer parameters than the teacher network, thanks to the transfer of knowledge via hints. The authors of the Highway Network paper did not replicate FitNet results. Due to FitNet's complexity, we did not implement FitNet (the student network) but we did implement the Teacher network, which is a baseline model discussed in (Srivastava et al., 2015). The Teacher network is a CNN using Maxout activation function (Goodfellow et al., 2013), which reports state of the art accuracy on the CIFAR-10 dataset.

2.3 Maxout & Dropout

Maxout CNN is a CNN that uses a function that outputs the maximum over a set of inputs (called Maxout activation function) as an alternative to the commonly used ReLU activation function (Goodfellow et al., 2013). Maxout function is a flexible and expressive activation function, and empirically achieves strong test accuracy on the CIFAR-10 dataset. We implemented Maxout activation function; two of our baselines use Maxout function: Maxout CNN on both the MNIST and CIFAR-10 dataset and Maxout MLP. Maxout CNN (also known as the Teacher Network) is the baseline in (Srivastava et al., 2015) we chose to implement. As discussed in (Goodfellow et al., 2013), dropout is complementary to Maxout activation and is used in our implementation for regularization. This is reflected in the results we obtained which we will cover in a later section.

2.4 MLP & CNN

Multilayer Perceptrons (MLP) are the simplest neural networks: a feedforward network. We found that the simple MLP baselines performed well on the MNIST dataset, but not on the CIFAR-10 dataset.

CNNs are capable of achieving state of the arts results in the field of computer vision (Chen et al., 2015), and was implemented with varying depths (the deepest CNN being 15 layers deep). We found that simple deep CNN baselines give comparable test accuracy as compared to less deep CNN variants (e.g. Maxout CNN) for the CIFAR-10 dataset.

2.5 All-Convolutional Net (All-CNN)

One of the models we tried for this project was inspired from (Springenberg et al., 2014). The authors propose to simplify a complex CNN model by using a standard convolutional layer with a greater stride instead of a max-pooling layer. They demonstrate that on multiple image recognition benchmarks with small images (i.e. CIFAR-10, CIFAR-100), the simplified CNN is as competitive as state of the art complex CNN model in terms of accuracy. We implemented the fully convolutional network (All-CNN) as a baseline for the handwritten digit recognition task and the CIFAR-10 image recognition task. We found that the All-CNN did not perform as well on the CIFAR-10 dataset.

2.6 Global Average Pooling

(Lin et al., 2013) propose global average pooling to replace the fully connected layers at the end of a convolutional neural network. A global average pooling layer works similar to a max pooling layer: instead of taking the maximum value, it takes the average. These layers are used to drastically reduce the dimensionality of the output before passing it through the softmax function. One clear benefit of global average pooling is that it minimizes overfitting by reducing the total number of parameters in the model. Our baseline All-CNN implementation uses global average pooling right before passing the output to softmax function.

3 Data and Setup

For our baseline validation, we trained our model to recognize the handwritten digit on the MNIST dataset. After validation, our models were evaluated on the CIFAR-10 dataset.

3.1 Dataset

The MNIST dataset is a collection of 60,000 images of handwritten digits (0-9) superimposed to a background noise. Our model's task is to recognize the digit associated with an MNIST image. We validated

our baseline models on the MNIST dataset. For model evaluation, we used a 80-20 training-validation split, 10,000 images being the test set.

The CIFAR-10 dataset is a collection of 60,000 color image. Each CIFAR-10 image belongs to one of ten predefined labels (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck) and has 32x32 pixels. Each label has equal number of examples, and are disjoint (i.e. a CIFAR image belongs to strictly one label). For model evaluation, we used a 40,000-10,000-10,000 data split. The reported test accuracies are on the 10,000 test set.

3.2 Preprocessing

ZCA whitening

This preprocessing step is commonly used for image recognition task (Goodfellow et al., 2013). In images, neighboring pixel raw values are highly correlated. The whitening transformation removes unwanted linear dependencies between features, whilst preserving spatial arrangements amongst neighboring pixels. CNNs heavily rely on spatial arrangements of pixels (Li et al., 2015). ZCA whitening is applied to all models on the CIFAR-10 dataset.

Data augmentation

This preprocessing step generates a new training dataset from an existing dataset by deforming the dataset. This augmentation gives more data to train on, and is empirically found to help with generalization. The intuition is that the deformation technique makes a model more robust against in-class variability while equally sensitive to between-classes variability. Data augmentation is only applied to the Deep CNN on the CIFAR-10 dataset. We used Keras' ImageDataGenerator[5] that allows us to create batches of tensor image data by looping over it. This also deforms the original data using rotation, flip, shifts, reflection, zoom, normalization etc.

4 Proposed Baselines

In this section, we discuss the baselines implemented, starting from the Teacher network in (Srivastava et al., 2015). We've implemented a shallow CNN, and a deep CNN on the CIFAR-10 dataset and the MNIST dataset. Moreover, we also implemented a simple MLP and a Maxout MLP and tested them on the MNIST dataset.

Teacher network: Maxout CNN

We used the same architecture as in (Goodfellow et al., 2013), also known as the Teacher network in (Srivastava et al., 2015) and (Romero et al., 2014). For the MNIST dataset, the Teacher network consists of three convolutional maxout hidden layers with 2 linear pieces each. We used 48, 48 and 24 kernels per layers respectively. Each convolutional layer is followed by a max-pooling layer with kernel size 4x4, 4x4 and 2x2 and with a stride of 2 pixels. Finally, the third layer is followed by a fully-connected layer using a softmax function. For the CIFAR-10 dataset, the Teacher network also has three convolutional maxout hidden layers with 2 linear pieces each. We used 96, 192 and 192 kernels per layers respectively. Each convolutional layer is followed by a max-pooling layer with kernel size 4x4, 4x4 and 2x2, and with a stride of 2 pixels. Finally, the third layer is followed by a fully-connected maxout layer with 5 linear pieces and 500 units. Lastly, a second fully connected layer using a softmax function ends the forward pass.

We used the same training stopping rule as described in (Romero et al., 2014): stop training after 100 epochs without any improvement on the validation set.

Shallow CNN

The goal behind this model was to compare (Goodfellow et al., 2013) results on a Maxout CNN with a shallow CNN. Shallow CNN's architecture has identical components as the Maxout CNN architecture, except the Maxout units are replaced with ReLU activation functions.

All-CNN

We implemented All-CNN from (Springenberg et al., 2014) as a baseline model. It includes 4 blocks of 3 convolutional layers. Each layer uses filters of dimension 3 by 3 and padding of 1. The first 2 layers of each block use a stride of 1, where as the 3rd layer uses a stride of 2, therefore replacing the max-pooling operation typically used in CNNs. Before passing the output through a softmax function, we perform a global average pooling. Except for the first layer, each subsequent layer performs batch normalization before passing through the ReLU activation function. The initial block uses 32 filters and this number doubles at each subsequent block, ending with a depth of 256. We did not use any dropout for the baseline, and chose Adam (Kingma and Ba, 2014) as an optimizer.

MLP & Maxout MLP

The architecture for the MLP has 2 hidden layers of 500 units each. ReLU was used as activation

[5] https://keras.io/preprocessing/image/#imagedatagenerator-class

function, as well as dropout with a probability of 0.5. We kept the same architecture for the Maxout MLP but using a Maxout activation function instead of a ReLU. We refer to these models as MLP and MaxMLP in the rest of this write-up. Both models are evaluated on the MNIST dataset only.

Deep CNN

We found that neither a shallow CNN architecture, nor the All-CNN architecture perform well on the CIFAR-10 dataset. Hence, we explored how varying the depth of CNNs affects performance, and implemented a Deep CNN architecture. The Deep CNN uses 6 convolutional layers with ReLU activation function and a ℓ_2 regularizer. Each layer is followed by batch normalization to increase training speed. After every two layers, we vary the filter size from 32, 64, 128 respectively and all have kernels of size 3X3. Additionally, We used max-pooling layers of size 2x2 and varying dropouts from 0.2, 0.3, 0.4 to regularize our data. Finally, the output dense layer has 10 nodes with a softmax activation. This model gave the best baseline results on the CIFAR-10 dataset.

5 Results

MNIST

The MNIST results serve to validate the baselines we've implemented. Although most of our models achieved strong and comparable performances, the Maxout CNN (the Teacher network) had the best results on test set: 98.89%, followed by 98.75% with a CNN that has the same architecture, but uses a ReLU activation function instead of a maxout unit. Next, the All-CNN model achieved 98.32% on test set. For all three models, a learning rate of 1e-4 and batch size of 64 examples was used.

For the MNIST dataset, we also implemented a basic MLP with ReLU activation function and a similar Maxout MLP. The former achieved a surprisingly high performance of 97.38% on test set. However, the Maxout MLP did poorly with a 91.33% test accuracy. Although our results on the Maxout CNN are strong, we are still below those reported by (Goodfellow et al., 2013) using a Maxout CNN and those reported by (Romero et al., 2014) using the FitNet: 98.89% vs 99.55%, 99.49% respectively. Results on the MNIST dataset for all the models implemented and those reported in the literature are shown in Table 1.

	Accuracy (%)			
Model	Train	Val.	Test	Note
MLP	98.37	97.43	97.38	Relu
Maxout MLP	93.3	91.81	91.33	Maxout
Shallow CNN	99.84	98.80	98.75	Relu
Teacher	99.78	99.0	98.89	Maxout
All-CNN	98.95	98.51	98.32	-
FitNet*	-	-	99.49	-
Teacher*	-	-	99.55	-

Table 1: **MNIST.** Training, validation, and test accuracies of proposed models. All models use Adam optimizer for training. The starred FitNet and Teacher test results were taken from (Romero et al., 2014) and (Goodfellow et al., 2013), respectively.

CIFAR-10

The reference baseline is the Teacher network in (Goodfellow et al., 2013), with 88.32% test accuracy. Our Maxout CNN (Teacher) test accuracy was 5.34% lower than the reference (82.98% vs 88.32%). We trained the Maxout CNN using a learning rate of 3e-4 and batch size of 128 examples for almost 500 epochs. The shallow CNN performed worse than the reference by 9.33% (78.99%). We found that the shallow CNN overfit the training set, even with a dropout rate of 0.5. This benefit of Maxout CNN over Shallow CNN is discussed in more details in (Goodfellow et al., 2013). Essentially, Maxout activation function acts as a regularizer and prevents overfitting.

All-CNN also reported lower test accuracy, as a result of overfitting: 82.30%, 69.18%, 69.75% on training, validation and test set. All-CNN was trained on only 10 epochs, using a 1e-4 learning rate and batch size of 64 examples. Lastly, the Deep CNN obtained a test accuracy of 88.10%, compared to the reference 88.32%. The best Deep CNN has dropout hyperparameter set to 0.4. Although our results with the Deep CNN are strong, we are still below those reported by (Romero et al., 2014) using the FitNet and those reported by (Srivastava et al., 2015): 88.10% vs 91.61%, 92.24% respectively. Results on CIFAR-10 dataset for all the models implemented and those reported in the literature are shown in Table 2.

6 Discussion and Conclusion

For the task of correctly classifying a tiny image into one of ten predefined labels, Deep CNN with ZCA whitening and data augmentation gave us the best validation accuracy of 89.13% and test accuracy of 88.10%. We observed that without data augmentation, our Teacher network were 5.34% off of the Teacher network baseline mentioned in Highway Networks. We believe that data augmentation is crucial to better generalization results on the CIFAR-10 dataset. For future work, we would like to extend our preprocessing methods to include global

Accuracy (%)				
Model	Train	Val.	Test	Note
Shallow CNN	87.16	79.35	78.99	Relu
Teacher	86.08	82.52	82.98	Maxout
Deep CNN	89.41	89.13	88.10	Relu
All-CNN	82.30	69.18	69.75	Overfit
Highway*	-	-	92.24	-
FitNet*	-	-	91.61	-
Teacher*	-	-	88.32	-

Table 2: **CIFAR-10.** Training, validation, and test accuracies of proposed models. All models use Adam optimizer for training. The starred Highway Network, FitNet and Teacher test results were taken from (Srivastava et al., 2015), (Romero et al., 2014) and (Goodfellow et al., 2013) respectively.

contrast normalization mentioned in (Goodfellow et al., 2013). We noticed that, in general, ZCA whitening and global contrast normalization is used in conjunction on the CIFAR-10 dataset.

We realize that beating the Teacher network is easy, using state of the art models on CIFAR-10. We noted that training parameters like max number of epochs had a great impact on practical utility of using the models. We noticed that the validation loss plateaus out after a critical number of training iterations. Optimizing a cut off iteration and maximizing the validation accuracy is important for practical utility of using deep networks.

For the current report, we considered many simple models. We would like to extend our baseline to include ensemble methods with even simpler models such as Support Vector Machines (SVMs) and CNNs. CNNs automate feature construction, and makes it difficult to interpret learnt features. Using simple SVMs would give us an easier time to interpret higher order features outputs from the CNN and may be able to give higher performance results (Niu and Suen, 2012).

7 Statement of Contributions

D.K. suggested the project. H.W.X. implemented the DeepCNN on CIFAR-10. M.-E. M.-C. implemented all the other baselines on MNIST and CIFAR-10. All three members contributed to the report writing.

References

[Chen et al.2015] L. Chen, S. Wang, W. Fan, J. Sun, and S. Naoi. 2015. Beyond human recognition: A CNN-based framework for handwritten character recognition. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 695–699, November.

[Goodfellow et al.2013] Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. 2013. Maxout networks. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1319–1327, Atlanta, Georgia, USA, 17–19 Jun. PMLR.

[Kingma and Ba2014] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

[Li et al.2015] Zuhe Li, Yangyu Fan, and Weihua Liu. 2015. The effect of whitening transformation on pooling operations in convolutional autoencoders. *EURASIP Journal on Advances in Signal Processing*, 2015(1):37, Apr.

[Lin et al.2013] Min Lin, Qiang Chen, and Shuicheng Yan. 2013. Network in network. *arXiv preprint arXiv:1312.4400*.

[Niu and Suen2012] Xiao-Xiao Niu and Ching Y. Suen. 2012. A novel hybrid CNNSVM classifier for recognizing handwritten digits. *Pattern Recognition*, 45(4):1318–1325, April.

[Paszke et al.2017] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

[Romero et al.2014] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2014. Fitnets: Hints for thin deep nets.

[Springenberg et al.2014] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. 2014. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.

[Srivastava et al.2015] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *CoRR*, abs/1505.00387.