Deep Resnets for 10 Class Image Classification

Aniruddha Ingle*
Department of Computer Science
Texas A&M University
College Station, TX, 77801
aniingle@tamu.edu

Abstract

Multi class image classification involves assigning unseen images to one of the target classes. In computer vision, using deep learning models is a common approach to solve this problem. It is known that training deep networks have certain barriers that involve propagating error through out the network, ensuring that there is coherent learning as the depth increases and computational feasibility. This work uses the well known residual framework to provide a pathway for information, unsullied by transformations, to propagate throughout the network along side information that is transformed using linear and non-linear transformations that are common to deep neural networks. Additionally, to improve on the computational burden of the deep layers, the bottleneck technique is used. The data set used is the well-known CIFAR 10 data set. This report consists of a brief history of the problem space, the realisation of Resnets as a successful solution paradigm for such problems, motivation for the techniques used, and a summary of the results that were obtained.

1 Introduction

Image classification is a problem that is best solved using the supervised learning paradigm. The conceptual idea is to train a model using well labeled images to learn patterns that are specific to the distribution of interest. Once, this model is trained, it can be used to make predictions on images that belong to a similar distribution as compared to the training data set.
Convolutional Neural Networks are a specialised type of deep learning models that have excellent utility in computer vision. Convolutional Neural Networks are able to encode spatial information within their operations. Spatial information is very important in the tasks related to computer vision like image classification because the pixel values relative to their neighbouring pixels form the important information needed to recognise image structures like boundaries and other patterns that could help in completing the computer vision task.
The following section outlines the proposed methodology for this project which uses a specialised convolutional neural network architecture called the Residual Network.
It also outlines the auxiliary strategies like data augmentation, hyper parameter tuning, learning rate scheduling, and so on.
In the final section of the report the results of the model are discussed in terms of the accuracy of validation and test. Training loss is also provided to provide deeper insight into the convergence of the model.

*

Preprint. Under review.

2 Proposed Methodology

This section of the report outlines the proposed methodology for this project. It is a brief review of the methods used to facilitate the training of a deep ResNet model.

2.1 Data Augmentation

Data Augmentation is the process of using available data to cover a larger distribution of the possible input space. In other words, data augmentation are a set of techniques that modify the currently available data to mimic the availability of a larger data set.

1. Rotation
Rotating an image is a useful data augmentation technique because it allows the network to learn to recognise images that are spatially different but materially similar. In other words, if a network is trained on a data set that has fewer instances of a cat with its head tilted to the right or the left, an unseen image like that could disrupt the prediction process of the model. However, the simple process of feeding rotated images of a cat can counter this issue.

2. Cropping
As Convolutional Neural Networks are good at encoding spatial image data, it is understood that they can understand the relative pixel positions and the amount of information present in these neighbouring regions. However, a network has the tendency to learn features that may or may not be important in prediction time. Hence, feeding cropped images to the model can force the mode to learn multiple paths towards the correct classification. To continue talking about the example of a cat. If the model begins to over-fit the prediction of a cat depending upon the shape of its ears and an image is provided such that the ears of a cat are obscured during test time. The model will have a difficult time predicting this instance correctly. Hence, randomly cropping and augmenting your data set is a good idea to make sure there are redundant learning paths for your model that lead to correct classification.

3. Flipping
The idea behind this technique is similar to the idea behind rotation, in data augmentation. We want to make sure that our model can identify and learn from various versions of our current data set. This allows the model to learn from a richer feature space and cover more of the possible input distribution in test time.

2.2 Residual Framework

The Residual Framework enables training a deep neural network whilst maintaining a similar number of parameters as compared to 'plain' version of these networks. The key ideas of the residual networks are using projection shortcuts that pass on the information from one chunk of the architecture to another chunk of the architecture in the forward pass with identity transformations. This allows for two flows for information. One of these flows carries the linear and non-linear transformations of the traditional neural networks. The other path short circuits the network by providing an older version of the information that enables deeper persistence in the network.

2.3 Bottleneck Block

As mentioned earlier, the residual framework relies on a deep model. As models get deeper the number of parameters increase and the number of computations become extremely computationally taxing. However, the bottleneck block is a solution to challenge these complexities. The bottleneck block reduces the feature space and then performs the computations. It then goes on to re-establish the feature space by up scaling it. This allows for good results. More information about the specifics of the bottleneck block is available in the implementation section.

2.4 Learning Rate Scheduling

The learning rate defines the magnitude of influence of the gradient on the weights. This in turn guides the convergence of the model towards the possible optima of of loss. Learning rate scheduling is used to dynamically influence this rate of influence. The scheduler requires identification of the number steps, i.e., the number of epochs at which the learning rate will be reduced by a constant factor.
The reduction of learning rate lies in the idea that- early on- the model has a lot of ground to cover, in terms of reaching the optima. This allows for larger steps in the optimisation process. However, as the epochs increase, the learning rate must decrease to avoid overshooting out of the optima, i.e., the minima in this case. Additionally, a higher learning rate in the later stages of the optimisation process could lead to weight oscillation. The model will jump back and forth between non

2.5 Experiments to Identify Key Hyper-parameters

The final step in the methodology is to identify the best performing hyper-parameters. Keeping in mind the depth of the architecture and the availability of data. The key hyper-parameters to tune were the number of epochs, the learning rate schedule, and the size of the ResNet. Identifying the learning rate schedule involved observing the training process of multiple models. The key indicator was saturating or repeating loss margins for recurring epochs. This influenced the choice of the step size used for the schedule and the factor of reduction. Additionally, the size of the ResNet was identified by bench-marking results on a subset of the data set. The final size of the ResNet was identified to be 18.
The Experimentation section discusses the results on the validation and test set, additionally, it provides further insight into how the final model was chosen.

3 Implementation Details

3.1 DataReader

DataReader is used to load the data from CIFAR Data Set. The data is available in five separate files. These files were individually loaded into a dictionary using **pickle**. Additionally, each dictionary consists of the image data and the labels. These two elements were added to a numpy array. These numpy arrays were concatenated to generate the total train data. This test data is later split into train and validation data.
The CIFAR data set consists of a **test_batch** This file was used to prepare the test data in a similar way.

3.2 Image Utils

ImageUtils is used for two major purposes. Normalising the image data and to perform augmentation operations like cropping and flipping.
To normalise the image data the image is subtracted pixel_wise with the mean of the channel. Then the image data is scaled downwards by the standard deviation. Again this is performed pixel_wise and across each channel. The image is cropped using a random seed value to generate a 32x32 image. Additionally, the image is also flipped on the probability of a coin toss.

3.3 Network

Network is where the magic happens. This is where the entire architecture is laid down for the convolutional networks. There are two convolutional networks, namely, Version 1, viz., this uses the standard residual block and Version 2, viz., this uses the bottle neck residual block.

Class Resnet
In Class Resnet, the entire architecure is laid out. Here, I implemented the start layer which is a 3x3 Convolution which operates on 3 input channels to give 16 output channels.

Class Batch Norm Relu Layer
In this Class we implement the combination of Batch Normalisation and Relu Activation. This layer is used intermittently between convolutions inside the standard and bottleneck blocks of the resnet.

Class Bottleneck Block
The bottleneck block is made up of three convolutional components. The first component is the bottleneck layer which reduces the feature depth by a factor of four. However, it is important to note that the bottleneck layer is pre-activated. This means that before this first down sampling component is applied, the input tensors are sent through the batch normalisation process and relu activation process.
This convolved output is then sent to a 3x3 convolution which maintains the input volume. Again, the process is preactivated, i.e., batch normalisation and relu activation are used before the convolution process.
This is followed by another preactivated convolution layer. In this convolution layer, the input volume is upsampled to the original input volume that came into the bottleneck block.

Class Stack Layer
The stack layer is used to stack multiple blocks to form a coherent part of the architecture. Depending upon whether the block type is standard or residual, the blocks are stacked and the projection shortcut is manipulated to provide the correct configuration.
The relationship between first_num_filters and filters_out is used to manipulate the input, output filters of the projection shortcut and the stride that should be used. Majorly, the complexity of the stack layer was in ensuring the volumes are provided as expected by the architecture. These volumes are a funciton of the stride used and the number of filters used in the convolution defined by the projection shortcut.

Output Layer
This layer is responsible for the conversion of the convolutional process to a fully connected neural network. To do so, the feature volume must be reduced. Averaging pooling is used to reduce the feature volume with stride two and kernel size 2 (in case of standard block). Average pooling is used to reduce the feature volume with stride four and kernel size 4 (in case of bottleneck block). Then the new output volume, in both cases, is flattened. This is then used to construct a fully connected neural network layer with ten output nodes. Each output node resembles a class. This is followed by a a softmax activation for multi class prediction.

3.4 Class Model

The model class is where the architecture comes to life. In this class, I define the optimizer, the error measure and a neat scheduler that can help implement the variable learning rate.
The error measure is cross entropy loss. The optimizer is SGD. Additionally, the weight decay is also added to the model here. The implementation of the mini batch sampling, as well, is produced in this class. The back propagation process is conducted here using the selected solver, i.e., SGD.

3.5 Loading Private Data Set and Reporting Predictions

The private data set needed to be transformed to get appropriate predictions. Initially, the model was giving out predictions that were not consistent with the performance of the model. It was realised through visualising images from the training data and the private data set that there was a difference in representation. This was fixed by reshaping the private data set images and transposing the image channels to match the representation of the images that were used to generate predictions. The predictions were converted to probability scores using a softmax non-linearity. These predictions were

4 Experiments and Results: Training, Validation, and Testing

There is a total training data set of 50,000 images that is provided in the CIFAR Data Set with a test batch of 10,000 images that are provided to benchmark the results of a deep learning model. This

section summarises the results of the three phases, i.e., Training, Validation, and Testing. The total training data is split into training and validation data with 45,000 images for training the model and 5,000 images for validating the model.

After the validation phase, the model is used to make predictions on the labeled testing data set of 10,000 images.After which the model is used to make predictions on the private testing data set. The private testing data set does not have labels, hence, the performance on the private testing data set cannot be reported. However, a distribution of the predicted classes was observed and reported.

4.1 Training Cross Entropy Loss:

The following are the results for the progression of the cross entropy loss with respect to epoch check points. The checkpoints are set at 10 epochs.

The loss was monitored to identify the point of stagnating of the metric. Stagnating loss is an indicating of reduction in the learning process.

10 Epochs, Loss: 0.408796
20 Epochs, Loss: 0.236620
30 Epochs, Loss: 0.088671
40 Epochs, Loss: 0.037238
50 Epochs, Loss: 0.001709
60 Epochs, Loss: 0.012359
70 Epochs, Loss: 0.006104
80 Epochs, Loss: 0.005638
90 Epochs, Loss: 0.001755
100 Epochs, Loss: 0.000983

The above results indicated at that the learning was succeeding. However, further experimentation was required to identify the success of this model and the choice of the best model as differentiated by the number of epochs. The idea is to identify if the model might be over-fitting as epochs progress.

4.2 Validation Accuracy:

The availability of the validation set and a test set that are disjoint to the final prediction task allowed for a two fold validation selection process. The results of this experiment informed the choice of the model but that decision would be further identified by the performance on the test set.

This validation set has a total of 5000 images.

Model with Epochs: 70 , Validation Score: 92.48%
Model with Epochs: 80, Validation Score: 92.40%
Model with Epochs: 90, Validation Score: 92.20%
Model with Epochs: 100, Validation Score: 92.64%

This experimentation has two learning outcomes:

1. The model performs reliably on unseen data. The Deep ResNet works and has a strong prediction ability. However, the best of these models can be further decided by performing experiments based on confidence and their performance on the test set.

2. The model performance does not increase tangibly over the four measured checkpoints. This could be an indication of learning within the model, as epochs progress, that may cause over-fitting or unfruitful weight modifications.

Why is the confidence of a prediction important?

In the process of making a prediction an algorithm generates probability scores for each class. This means that the algorithm generates a fuzzy set which is the score for each class. The mechanism that follows this process of generating a fuzzy set is taking the class with the highest confidence. However, an ideal algorithm would be able to classify an image with high confidence for the correct class and assign really low values for classes that are not correct. Hence, an algorithm that makes correct classifications with high confidence is valuable.

Analysing Confidence of Predictions on Validation

An exploratory analysis was performed on the predictions of the models that indicated that the model which was trained on 80 epochs might have better confidence. However, this was not a metric that could be believed right away. This is where the test set accuracy- which in this case can be used as the second fold for validation due to a hidden test set- was leveraged to perform more exhaustive validation.

4.3 Test Set Accuracy:

The test set- which is being used as another round of validation- has a total of 10,000 images.

Model with Epochs: 70 , Test Score: 91.88%
Model with Epochs: 80, Test Score: 92.29%
Model with Epochs: 90, Test Score: 91.87%
Model with Epochs: 100, Test Score: 92.19%

It can be seen that even though the difference is marginal the model with 80 epochs outperforms the other models. This adds confidence to the exploratory analysis mentioned above.

4.4 Private Test Data Set:

The private set consists of 2000 images. There are no labels provided of these images. The requirement is to submit predictions for this data. Initially, the results of this model seemed to be skewed. However, it was realised that the representation of the images was different as compared to the representation of the images on which the models were trained. Applicable transformations and reshaping were implemented to perform successful predictions.

The predictions on the private test set are attached in the submission.

4.5 Appendix

The final submission consists of the code, a read me file on how to interact with the main.py file to perform important functions like training, testing, and predictions. For the interest of space, the best four models are submitted for verification of the results.