CS4487 Machine Learning

Course Project

Technical Report

Name: LI Haotian

SID: 54780576

Kaggle ID: 4487



Table of Contents

[1. Introduction 3](#_Toc27605531)

[1.1. Dataset – CIFAR 10 3](#_Toc27605532)

[1.2. Course Project 3](#_Toc27605533)

[2. Literature Study 3](#_Toc27605534)

[2.1. Fractional Max-Pooling 3](#_Toc27605535)

[2.2. Striving for Simplicity: The All Convolutional Net 4](#_Toc27605536)

[2.3. All you need is a good init 4](#_Toc27605537)

[2.4. Generalizing Pooling Functions in Convolutional Neural Networks: Mixed, Gated, and Tree 4](#_Toc27605538)

[2.5. Workflow 4](#_Toc27605539)

[3. Result Analysis 5](#_Toc27605540)

[Reference List 7](#_Toc27605541)

# Introduction

## Dataset – CIFAR 10

The CIFAR-10 dataset (Canadian Institute For Advanced Research) is a collection of images that are commonly used to train machine learning and computer vision algorithms. It is one of the most widely used datasets for machine learning research. The CIFAR-10 dataset contains 60,000 32x32 color images in 10 different classes. The 10 different classes represent airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. There are 6,000 images of each class.

Evaluation Criteria:

A picture containing object

Description automatically generated

## Course Project

In this project, the previous dataset was adopted for training multiple Deep Neural Networks(DNN) and improve the model accuracy on which is tested on Kaggle (<https://www.kaggle.com/c/cs4487cp>).

# Literature Study

## Fractional Max-Pooling

This paper introduces a new method of max-pooling which is good to be implemented during model training. The amazing by-product of discarding 75% of the data is that can build into the network a degree of invariance with respect to translations and elastic distortions. However, if simply alternate convolutional layers with max-pooling layers, performance is limited due to the rapid reduction in spatial size, and the disjoint nature of the pooling regions. It also formulated a fractional version of max-pooling where alpha is allowed to take non-integer values. Fractional max- pooling is stochastic as there are lots of different ways of constructing suitable pooling regions. It also greatly reduces overfitting .

## Striving for Simplicity: The All Convolutional Net

This Paper re-evaluate the state of the art for object recognition from small images with convolutional networks, questioning the necessity of different components in the pipeline. It is found that max-pooling can simply be replaced by a convolutional layer with increased stride without loss in accuracy on several image recognition benchmarks. To analyze the network a new variant of the “deconvolution approach” is introduced for visualizing features learned by CNNs, which can be applied to a broader range of network structures than existing approaches.

## All you need is a good init

In this paper, layer-sequential unit-variance (LSUV) initialization – a simple method for weight initialization for deep net learning – is proposed. The method consists of the two steps. First, pre-initialize weights of each convolution or inner-product layer with orthonormal matrices. Second, proceed from the first to the final layer, normalizing the variance of the output of each layer to be equal to one. Experiment with different activation functions (maxout, ReLU-family, tanh) show that the proposed initialization leads to learning of very deep nets that (i) produces networks with test accuracy better or equal to standard methods and (ii) is at least as fast as the complex schemes proposed specifically for very deep nets such as FitNets (Romero et al. (2015)) and Highway (Srivastava et al. (2015)). Performance is evaluated on GoogLeNet, CaffeNet, FitNets and Residual nets and the state-of-the-art, or very close to it, is achieved on the MNIST, CIFAR-10/100 and ImageNet datasets.

## Generalizing Pooling Functions in Convolutional Neural Networks: Mixed, Gated, and Tree

In this paper, it introduces two primary directions: (1) learning a pooling function via (two strategies of) combining of max and average pooling, and (2) learning a pooling function in the form of a tree-structured fusion of pooling filters that are themselves learned. In the experiments every generalized pooling operation explore improves performance when used in place of average or max pooling. It experimentally demonstrate that the proposed pooling operations provide a boost in invariance properties relative to conventional pooling and set the state of the art on several widely adopted benchmark datasets

## Workflow

* Step 1: Load dataset with torchvision and normalize the dataset
* Step 2: Define a network with pytorch / keras / tensorflow
* Step 3: Define loss function
* Step 4: Training Models on GPU (Colab) and save to local
* Step 5: Test Accuracy on Kaggle
* Step 6: Refine params-tuning

# Result Analysis

* 1. ResNet

Even though ResNet is much deeper than VGG16 and VGG19, the model size is actually substantially smaller due to the usage of global average pooling rather than fully-connected layers which reduces the model size down.

| **Model** | **n** | **200-epoch accuracy** | **sec/epoch GTX1080Ti** |
| --- | --- | --- | --- |
| ResNet20 v1 | 3 | 92.16 % | 35 |
| ResNet32 v1 | 5 | 92.46 % | 50 |
| ResNet44 v1 | 7 | 92.50 % | 70 |
| ResNet56 v1 | 9 | 92.71 % | 90 |
| ResNet110 v1 | 18 | 92.65 % | 165 |

| **Model** | **n** | **200-epoch accuracy** | **sec/epoch GTX1080Ti** |
| --- | --- | --- | --- |
| ResNet56 v2 | 6 | 93.01 % | 100 |
| ResNet110 v2 | 12 | 93.15 % | 180 |
| ResNet164 v2 | 18 | - % | - |
| ResNet1001 v2 | 111 | - % | - |

* 1. VGG

During Pre-training The smaller networks converged and were then used as initializations for the larger, deeper networks — this process is called pre-training.

While making logical sense, pre-training is a very time consuming, tedious task, requiring an entire network to be trained before it can serve as an initialization for a deeper network.

In summary, it is painfully slow to train. And the network architecture weights themselves are quite large compared to other models.

* 1. DenseNet

DenseNets have several compelling advantages: they alleviate the vanishing-gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters.

* 1. Inception\_v3

Similar to GoogleNet, it used to be called googlenet. The Inception V3 architecture included in the Keras core comes from the later publication by Szegedy et al., [Rethinking the Inception Architecture for Computer Vision](https://arxiv.org/abs/1512.00567)(2015) which proposes updates to the inception module to further boost ImageNet classification accuracy. The weights for Inception V3 are smaller than both VGG and ResNet.

# Reference List

1. STRIVING FOR SIMPLICITY: THE ALL CONVOLUTIONAL NET, <https://arxiv.org/pdf/1412.6806.pdf>
2. Fractional Max-Pooling, <https://arxiv.org/pdf/1412.6071.pdf>
3. All you need is a good init, <https://arxiv.org/abs/1511.06422>
4. Generalizing Pooling Functions in Convolutional Neural Networks: Mixed, Gated, and Tree, <https://arxiv.org/abs/1509.08985>