VGU
Vietnamese-German University

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

ACADEMIC REPORT

AI FOR RENEWABLE ENERGY

Summary of Convolutional Neural Networks

Author:
Thai Hoang Tam - 17387

Instructor:
Prof. Bui Minh Duong
Prof. Tran Hong Ngoc

April 26, 2023

Contents

Chapter 1

Introduction

The report shows a summary of Convolutional Neural Networks (CNN). Most of the material will be from A survey of Convolutional Neural Networks: Analysis, applications, and prospects [1]. The main part of the summary will be about briefly explaining what is CNN, what it is used for, and some key concepts on how to choose the correct hyperparameters, and activation functions. And finally, briefly introduce the CNN application and energy-related situation.

Chapter 2

Summary

2.1 Overview

CNN is a type of Artificial Neural Network (ANN) and can perform some amazing tasks such as face recognition, autonomous vehicles, and image classification. The advantage of CNN compared to the older model such as single-layer perceptron networks is that it can handle linear inseparable problems (such as XOR problems). CNN works by extracting features from the input with convolution structures. It is inspired by visual perception (making humans see things). CNN has some advantages over Fully Connected networks such as local connection (a node does not connect to all nodes from previous layers) which help reduce parameters.

2.1.1 Deformable Convolution

General neural networks have fixed geometric structures so it is not robust to the geometric transformation such as rotation, and scaling. This can be overcome by data augmentation (changing data conditions such as rotating an image, changing lighting, etc) but this needs more work done before training. Deformable convolution can adapt to some geometric transformation.

2.1.2 Group Convolution

Group convolution can help build wider networks, and reduce parameters, it can even improve accuracy. Group convolution is less likely to lead to overfitting by reducing the number of parameters.

2.1.3 Steerable Convolution

An invariant network can not perform accurate recognition with a rotated image. Steerable convolution helps to increase the flexibility of equivariant CNN.

2.1.4 Graph Convolution

Standard CNN convolves based on pixels on an image but Graph neural network (GNN) convolves based on graph relation.

2.2 Discussion

This section will be about activation functions, loss functions, optimizers, hyperparameters, and some guides to choose the suitable one for the model and data. The summary will mainly about this section.

2.2.1 Activation Function

The activation function determines which information should be transferred to the next layers. With nonlinear activation functions, deep neural networks can approximate any function. Some popular activation functions:

- Sigmoid function
- Tanh function
- ReLu function
- Other versions of ReLu like Leaky ReLu, PReLu

The sigmoid function will map any value from (-∞, ∞) to the range (0, 1), suitable for binary classification problems. When the value is 0 it will map to 0.5.

Unlike the sigmoid function, the tanh function will map any real number to the range (-1, 1). The mean output for tanh function is 0 which makes it can achieve something similar to normalization.

ReLu function will return max(0, input), which makes it speeds up learning with easier calculation. However, ReLu has a problem which is dead or inactivated neurons. Because with input less than 0, the derivative is 0 and the previous layer will be passed with no errors when back-propagated. Therefore, some other version of ReLu appeared. With Leaky ReLu, when the input x is less than 0, the output of leaky ReLU is x/α, instead of zero, where *alpha* is a fixed parameter in the range of (1, $+\infty$). And a more improved function is PReLu with the α based on the data and not a predefined one like leaky ReLu.

How to choose the correct activation function?

Some rules that are helpful and act as guiding points:

- Last layer can use the sigmoid function for binary classification problems, and the softmax function for multiclassification problems.
- In hidden layers, ReLu or leaky ReLu is a good choice.
- ReLu or leaky ReLu are the good default choice for activation functions.
- If many neurons are not activated in the training process, try leaky ReLu and PReLu.
- Choose a negative slope of 0.02 in ReLu to speed up training.

2.2.2 Loss Function

Loss functions show the difference between the predicted result and the actual results. The goal for the regression problem and classification problem is to minimize the loss function. Some most commonly used loss functions:

- Mean absolute error (MAE)
- Mean square error (MSE)
- Cross entropy

2.2.2.1 Regression Problems

Mean absolute error and mean square error are for regression problems. MAE is good when data has many outliers, but the MSE result is derivable so the rate of update can be adjusted. So when the outliers in the data negatively affect the result, MAE should be used. Otherwise, use MSE for loss function.

2.2.2.2 Classification Problems

Cross entropy is preferred for classification problems. The function compared the predicted result with the actual output and calculate the difference. There are also other versions of cross entropy that focus on the distance between categories (or in a category) and not only on individual classes.

How to choose the correct loss function?

Some guides to choosing loss functions:

- Choose MAE or MSE for regression problems
- Choose either one of Cross entropy, contrastive loss, triplet loss, center loss, or large-margin softmax loss for classification problems
- Cross entropy is a popular choice with a softmax layer in the end.

2.2.3 Optimizers

To optimize or reach good and optimal network parameters, optimization algorithms are used. Most of the methods are based on gradient descent. There are three kinds of gradient descent methods:

- Batch gradient descent (BGD)
- Stochastic gradient descent (SGD)
- Mini-BGD (MBGD)

BGD calculates the whole batch of data for each update, which ensure convergence to the global optimum and local optimum, but with some drawbacks like slow computing time, data may not be suitable for in-memory calculation. SGD uses one example for each update, this makes it faster to compute but also with high variance and can cause severe oscillation. In some cases, SGD never converges due to the high oscillation. MBGD combines the advantages of both BGD and SGD, with the batch size depending on the GPU or CPU ability.

On the basis of MBGD, many effective algorithms are introduced. First is the momentum algorithm, which simulates the physical momentum. Nesterov accelerated gradient (NAG) even gives the predictability to slow down at positive slopes. Adagrad algorithm adapts the learning rate to parameters, making smaller updates for frequent feature-related parameters and large updates for less frequent ones. Adagrad algorithm is suitable for sparse data.

How to choose the suitable optimizers?

- MBGD should be used to balance the computing cost and accuracy of each update.
- Performance of the optimizer depends on the data and should be chosen considering their strengths and weaknesses.
- If oscillation appears too frequently, try to reduce the learning rate.

2.2.4 Hyperparameters

Other hyperparameters besides activation functions, and loss functions can greatly affect the model performance. The priority orders of hyperparameters are as follows:

1. Learning rate
2. Acceleration value, convolution kernels, and mini-batch size.
3. Number of layers, learning rate decay
4. Other hyperparameters of the optimizer

Tuning hyperparameters can be very costly without a good search strategy. First, try to take random values from an appropriate scale. If the number of layers can be from 4 to 6, try 4, 5, and 6. Then narrow down the range of possible choices. For example, when tuning the learning rate that can be from 0.0001 to 1, try to think of it as 10^{-4} to 10^{0}, and we can choose a value from -4 to 0.

There are also some algorithms to automatically optimize the hyperparameters such as derivative-free optimization (DFO), which iteratively leverages the information about the optimal solution to approximate the optimal solution. Another algorithm is Simplified swarm optimization (SSO) which is efficient and does not change the CNN structure to optimize hyperparameters [3].

Chapter 3

Application and Energy-related situation

CNN is a core concept in deep learning that can give a good solution to problems like time series prediction, signal identification, image classification, object detection, image segmentation, and many more. Despite having many benefits, CNN still faces many problems when working with diverse data. When adding random noise to the image the accuracy can reduce significantly (or even predict a wrong class) but to the human eye, there are barely any differences in the image. Another drawback is computing resources. CNN requires a large number of computing resources for large and complex neural networks. This can consume a lot of energy and are not suitable for mobile devices such as smartphones and wearable devices [2]. However, there are proposes of energy-efficient CNN [2] that may change how CNN is trained and used to be more environmentally friendly and consume less energy but still with good accuracy.

Bibliography

[1] Li Z;Liu F;Yang W;Peng S;Zhou J;. A survey of convolutional neural networks: Analysis, applications, and prospects.

[2] Tien-Ju Yang, Yu-Hsin Chen, and Vivienne Sze. Designing energy-efficient convolutional neural networks using energy-aware pruning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[3] Wei-Chang Yeh, Yi-Ping Lin, Yun-Chia Liang, Chyh-Ming Lai, and Chia-Ling Huang. Simplified swarm optimization for hyperparameters of convolutional neural networks. *Computers & Industrial Engineering*, 177:109076, 2023.