Research on the Robust and Low-cost Computer Method Based on Deep Residual Learning

Chung Ka Po

The Webb Schools, Claremont, California
G13what.never@gmail.com

Abstract. *At this exact moment on our planet, there's are more than 65.2 million people suffering from visual illnesses caused by cataract. It is the cause of a third of all blindness in the world, and 99% of these patients live in developing countries. However the recovery surgery is one of the most cost-effective ones in the field. Lack of medical care is the real cause of such skyrocketing number of patients in developing country. Out of this reason, I developed an integrated diagnosis system based on deep learning methods, which could diagnose cataract with the accuracy of 91.7%. The high accuracy and the low-cost features of this diagnosis method make it an excellent auxiliary tool of preliminary diagnosis in developing countries. The best performance is held by the 50-layer deep residual neural network trained with Adam optimizer, which could adjust learning rate according to the training status and specific weights. The further experiments with the quantity of data as a variable indicated that the best performance of deeper model is limited by the insufficient data.*

Keywords: *Low-cost, Computer, Deep Residual Learning*

1. Introduction

Hearing this problem, I starting contemplating the ways to resolve this problem, to make medical resources more accurate, efficient, and available to these developing countries. Then it hit me, if cataract rates are so high in these countries, the most fundamental step to take would be first identifying whether a person had cataract. Judging by the scanty medical resources in the countries, asking to be diagnosed for cataract would often be a tedious process, with doesn’t necessarily yield accurate results. Which is why I decided to develop a program utilizing TensorFlow to recognize and report different stages of cataract in potential patients.

The development of computer vision in recent years had been skyrocketing, improving the functionality of classification, object detection, and such technologies in ways surpassing the traditional algorithms. The support-vector machine for instance, is a traditional supervised learning model that utilizes kernel functions to fit data and perform classification tasks.

Benefitting from massive data and the GPU's massive parallel computing power at the same time, the randomized gradient descent to optimize the training of the convolutional neural network in the iterative update, and reached an unprecedented height in the picture-related tasks.

2. Materials and methods

2.1 Data and data cleaning method

The whole diagnosis model takes the images of the patients' eyeball as its inputs. All these raw image-based data come from an open-source dataset of cataract diagnosis. Most of these raw data utilize JPEG as its primary encoding method and present image in RGB color space, while a small part of them use PNG as their primary form [3], which has an extra channel.

These data all firstly are coarsely divided by two classes, cataract patients and healthy people. Then a more fine-grained classification method is used, and the data of cataract patients are assigned to different levels of severeness.

The basic statistics can be seen in the following chart:

Table 1 Distribution of different levels

Class	Level1	Level2	Level3	Level4	Level5
	19.3%	17.4%	21.9%	16.85%	24.55%

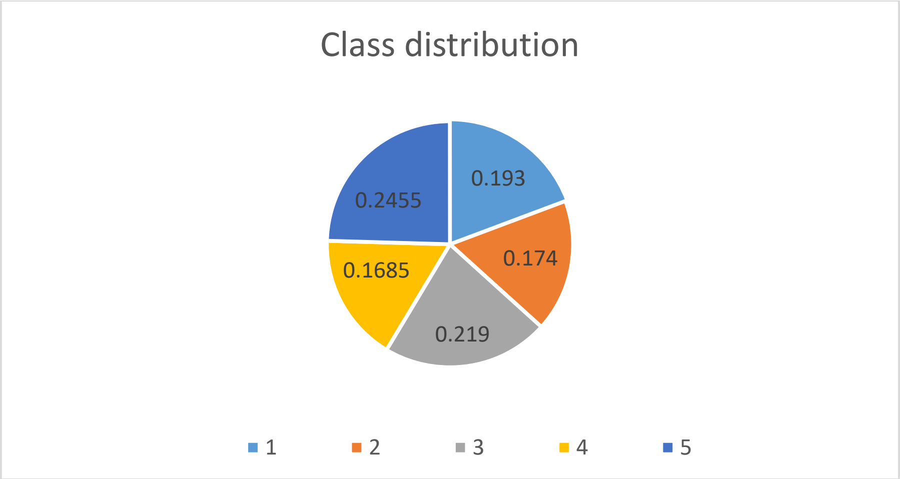

Figure. 1 Class Distribution

Some examples of different levels of cataract can be seen from below:

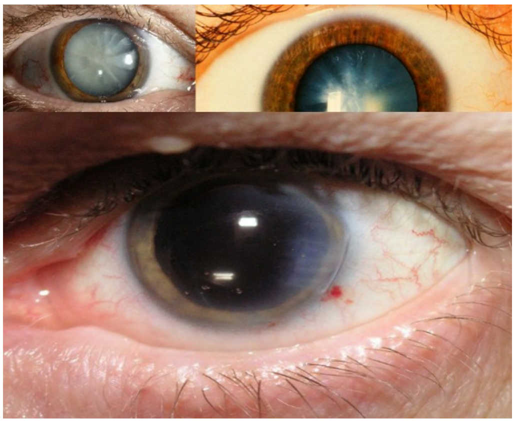

Figure. 2 Data example

The data cleaning part mainly takes three jobs: deleting images with PNG format, resizing them all in fixed size, and realign the presenting data shape. Because the deep learning model used is set to take only 3-channel input, the 4-channel PNG format is abandoned in the process. Due to the constraints posed by the last fully connected layer of the deep residual model, all the data must be altered into a fixed size. The realignment of the data comes from the specification of the Pytorch framework, which provides the basic functionality of implementing a deep neural network.

2.2 Abstraction of data and dataset

In the actual engineering implementation, the dataset should be reorganized to fit the parallel features provided by modern computing hardware, like GPU(Graphics Processing Unit). The data input into the diagnosis model is wrapped in a batch form, which is to say that multiple images are processed at the same time. In the meantime, to provide a flexible training process, the data input order should be shuffled to avoid the dependence of the order of the model, which could compromise the generalization ability of the diagnosis model.

The official paradigm suggested by the Pytorch framework is to implement both the subclass of DataSet and DataLoader. The former one provides an abstraction form of the dataset, which could index specific data items and provides the information on the quantity of the dataset. The later one wraps the former one and provides built-in methods of shuffling.

Hardware inevitably influences the training process and even the performance of the results. With the technology of batch normalization, greater batch size usually means a more robust training process. However, the batch size is constrained by the physical memory of the GPU. All of the finished experiments in this paper are based on GTX 1080TI, produced by Nvidia Inc. It has a physical memory of 12 GB.

2.3 Software dependency

The main libraries and frameworks used are PIL and Pytorch. PIL is an image library for Python, which takes the responsibility of image decoding and encoding. Pytorch is one of the popular deep learning framework sustained by Facebook. Pytorch supports dynamic graph structure, which is more flexible than the static graph structure implemented by Tensorflow. Pytorch provides basic and robust implementation of convolution operations and some commonly used loss functions and optimizing algorithms.

2.4 Diagnosis model

A deep residual neural network is proposed to be used in the diagnosis model. In the common image classification task, various models are proposed in various circumstances. The deep residual learning model is chosen because it is based on its core residual block, which is specifically designed to improve the depth of the neural network. Many experiments indicated that a deeper neural network could outperform shallow ones if sufficient data are given. However, a deeper model could also overfit the data when less data are used. According to this, I utilize resblocks to build up a neural network with different layers.

2.5 Basic operations

The neural network mainly takes several usually used operations: convolution, pooling, batch normalization [4], and activation.

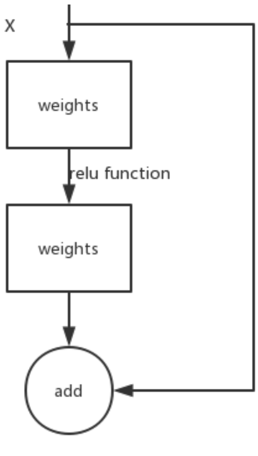

Figure. 3 Basic Block

The basic structure of a residual block comprises mainly two parts: the basic block, the bottleneck block. The following figure visualizes this structure.

Both the basic block and the bottleneck are combination of basic operations of Convolution, Pooling and batch normalization. This can also be described using the following formula:

$$F(x) = H(x) + G(x)$$
$$H(x) = BasicBlock(x)$$
$$G(x) = Bottleneck(x)$$

The $G(x)$ part can be regarded as the residual part of $F(x)$, which could resolve the bias between the expected output and the actual output by the front end of the network. This technology significantly increases the depth of the neural network and makes it possible to break through the former depth limitation.

Deep residual block is the core part of the network and could be stacked layer by layer to make up the backbone of the whole diagnosis model. In this project, I built up four versions: the 34 layers, 50 layers, 101 layers version, and 152 layers version. Except for the backbone of the diagnosis model, there are still two main layers of the neural network.

The first one is the fully connected one, which requires the output of the former layer to be reshaped into a vector. The operation fully connected layer take is actually a matrix product, which outputs a fixed-length vector. The number of classes restricts the length of the vector. Parameters in this layer will increase rapidly along with the number of classes, sometimes even holds the most quota of parameters of the model.

The second part is the sigmoid function, which tries to provide a meaningful explanation in probability.

Sigmoid function can be described in the following form:

$$Sigmoid(x) = \frac{1}{1 + \exp(-x)}$$

The corresponding figure of this function could be seen in figure 4.

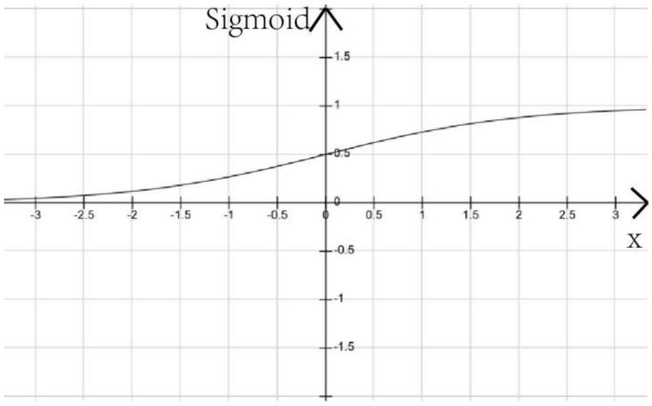

Figure. 4 Sigmoid Function

The single element of the output of the sigmoid function is the probability assigned to the according class.

2.6 The training process

Initial parameters in the neural network are all initiated by some random generators, which means that the neural network without training is useless. In this project, supervised learning is used, which could result in a more exemplary performance than unsupervised one.

The core process of training is backpropagation, which uses massive data to optimize the inner parameters. Loss function and optimizers, the wrapper of optimization algorithms, take the responsibility of implementing this process.

Loss function is the criteria to judge the bias between labels and actual outputs. In this project, the CrossEntropyLoss [5] is used.

CrossEntropyLoss:

$$loss(x, class) = -\log(\frac{\exp(x[class])}{\sum_j \exp(x[j])}) = -x[class] + \log(\sum_j \exp(x[j]))$$

Optimizer

Optimizer uses the losses calculated by loss function to update the parameters in the neural network. Usually, there are Stochastic gradient descent, Adam optimization, and AdaGrad.

Stochastic gradient descent and other machine learning optimization method provide ways to minimize the objective loss function, like CrossEntropyLoss, which has the form of a sum:

$$P(w) = \frac{1}{n}\sum_{i=1}^{n} p_i(w)$$

Stochastic gradient descent would use the following method to minimize the objective function:

$$\mathrm{w} := \mathrm{w} \text{ -}\eta\nabla P(w) = w - \eta\sum_{i=1}^{n}\nabla p_i(w)/n$$

η is the learning rate in training.

To prevent Stochastic gradient descent from being stuck in the local optimal area, it is usually combined with momentum.

Different from Stochastic gradient descent, the Adam optimization method maintains a specific learning rate for each parameter. This makes Adam optimization method more effective and robust than original SGD algorithms.

With the above criteria and optimization method, the whole training process can be depicted below:

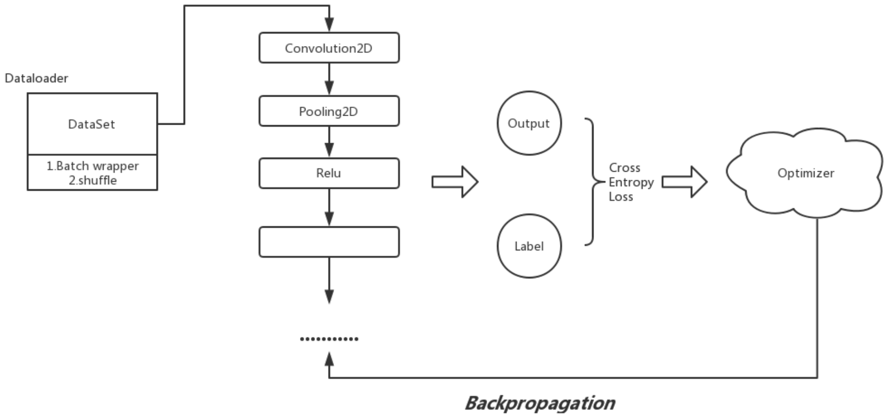

Figure. 1 Training process of the neural network

2.7 The experiment strategy

To train and evaluate the model separately, 80% of the data is used in training, while the other 20% is used in evaluation. The learning rate is set to 0.001 to all models at the begin. Several optimizers are tested against each to find the most efficient one.

To scale the limitation lead by the number of data, I also checked the relation between the best performance and the amount of data.

3. Result

3.1 Model depth against performance

Model depth is usually considered a significant standard for model performance. The performance typically increase along with model depth. However, it also requires sufficient data as its precondition. So, taking a moderate size is more meaningful to the diagnosis task.

It can be seen from the chart that the diagnosis performance increase with increasing the depth of the model until the depth approach 101 layers. The 152 layer version failed to converge and show terrible generalization ability on the validation data. It may be caused by the limited amount of data collected.

FLOPS and number of parameters are two standards to evaluate the operands. It could be seen that a massive increase in FLOPS may lead to subtle improvement.

Table 2 Parameters and FLOPS

Model Name	Accuracy	FLOPS	Number of Parameters
ResNet34	83.2%	3678M	21.3M
Res50	89.3%	4131M	23.5M
ResNet101	91.7%	7864M	42.5M
ResNet152	67.8%	11601M	58.2M

3.2 Training with different optimizers

The optimization method mentioned above is tested on performance and efficiency.

It can be seen that Adam is performing better both in performance and efficiency, which takes less time in converging to fine results.

Table 3 Accuracy and Optimizers

Model Name	SGD algorithm accuracy	Adam algorithm accuracy	SGD training duration	Adam training duration
ResNet34	80.5%	83.2%	9.3h	7.0h
ResNet50	84.7%	89.3%	9.5h	7.1h

Because the batch size could influence the training duration and performance, the batch size of both models is set to 64. Further experiment on deeper version is limited by limited physical memory of GPU.

4. Conclusion

The best performance of the diagnosis model shows 91.7% accuracy on predicting different levels of cataract, which could be a great help for doctors. The experiment considering the limitations of data amount shows more various data could lead to a more exceptional performance. This whole system could easily be built on a computing cloud and be utilized everywhere. Developing countries that lack professional doctors could greatly benefit from this diagnosis model. Patients can be categorized into those who need immediate medical attention, and those who do not need it at all. People in remote areas could then be diagnosed by this intelligent system first, and categorized into those who need immediate medical attention, and those who do not need it at all. This would result in a more efficient usage of medical care, and my research will become the remedy of a lifetime of suffering to some people, and a stepping stone for future researchers to extend this noble pursuit of the solution to one of the biggest health care issue around the world.

References

[1] Steinkuller, P. G. (1983). Cataract: The leading cause of blindness and vision loss in Africa. Social Science & Medicine, 17 (22), 1693-1702.
[2] Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. arXiv: Learning,.
[3] Roelofs, G. H. (2005). History of the Portable Network Graphics (PNG) Format. Linux Journal.
[4] Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv: Learning,.
[5] De Boer, P., Kroese, D. P., Mannor, S., & Rubinstein, R. Y. (2005). A Tutorial on the Cross-Entropy Method. Annals of Operations Research, 134 (1), 19-67.