CNN in Histopathologic Cancer Detection

Qiang Guo, Shuo Tian, and Chunyu Mao

***Abstract*—Histopathologic cancer detection is a general problem belongs to the category of medical image processing. Due to the development of deep learning, the problem can be solved by computer instead of experts involvement. Convolutional Neural Network (CNN) is one of the most popular deep learning architecture in image processing since it can not only reduce the number of neurons, but also extract texture features of the images. Since 2012, multiple CNN based network architectures are being proposed for image classification. Our project will focus on several techniques in CNN design with evaluation of the effects through multiple experiments. And then design a shallow CNN for histopathologic cancer detection in Kaggle competition. The data set is a large set of colored medical images provided by Kaggle. The results show that our design is suitable for cancer detection. The most promising CNN architecture is NasNet since it can search the best CNN architecture in model training.**

***Index Terms*—Cancer Detection, Kaggle Competition, CNN.**

I. Introduction

OVER the last several years, machine learning techniques, particularly neural networks, have played an increasingly important role in the design of pattern recognition systems. In fact, it has been agreed that the availability of learning techniques has been a crucial factor in the recent success of medical image processing [1]. Moreover, deep learning algorithms such as deep neural networks, deep belief networks and recurrent neural networks have various application areas including computer vision, speech recognition, natural language processing, audio recognition etc., where they have produced results comparable to and in some cases superior to human experts.

As before, the identification of metastatic cancer has high clinical relevance and would normally need extensive microscopic evaluation by pathologists. Therefore, an automated solution would hold great promise to reduce the workload of pathologists while at the same time reduce the subjectivity in diagnosis. The goal of the project is to investigate and evaluate deep learning algorithms for automated identification of metastatic cancer in small image patches taken from larger digital pathology scans.

Deep learning uses a cascade of multiple layers of nonlinear processing units for feature extraction and transformation. Each successive layer uses the output from the previous layer as input. It has the ability to learn multiple levels of representations that correspond to different levels of abstraction; the levels form a hierarchy of concepts[2].

Application of deep learning algorithms to whole-slide pathology images can potentially improve diagnostic accuracy and efficiency. Deep learning is a rich family of methods, encompassing neural networks, hierarchical probabilistic models, and a variety of unsupervised and supervised feature learning algorithms. The three of the most important types of deep learning models with respect to their applicability in visual understanding are Convolutional Neural Networks (CNNs), the Boltzmann family including Deep Belief Networks (DBNs) and Deep Boltzmann Machines (DBMs) and Stacked (Denoising) Autoencoders.

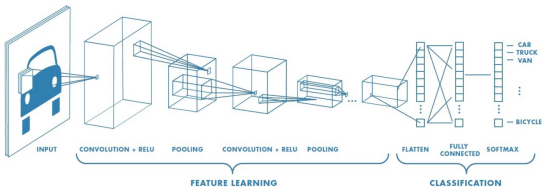


Fig. 1. Neural network with many convolutional layers

Convolutional neural network (CNN) is one of the main categories to do image recognition and classification. Objects detection, recognition faces etc., are some of the areas where CNNs are widely used. CNN for image classification takes an input image, process it and classify it under certain categories. System can treat the input image as a two dimensional array of pixels with fixed resolution. From figure 1 we can see that when implements CNN to train and test models, each input image will pass it through a series of convolution filters (Kernals), pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1 [3]. As a result, CNN is categorized as a novel two-dimensional taxonomy for image processing. One dimension specifies the type of task performed by the algorithm: preprocessing, data reduction/feature extraction, segmentation, object recognition, image understanding and optimization. The other dimension captures the abstraction level of the input data processed by the algorithm: pixel-level, local feature-level, structure-level, object-level, object-set level and scene characterization.

The paper is organized as follows. A brief background introduction about the issue needs to be addressed and the method would be implemented, including, deep learning, neural networks and CNNs has been given in this section. Then the detailed information about the data set and problems to be solved will be described in section II. Section III gives a survey of the development and the cut-edge technology of CNNs. Then the method and approaches applied in this paper will be stated in section IV. Section V presents all the experiments details and results analysis. Finally, a brief conclusion will be given in section VI.

II. Problem Definition

The data for this project is a slightly modified version of the PatchCamelyon (PCam) benchmark dataset from the kaggle competition, which is a new and challenging image

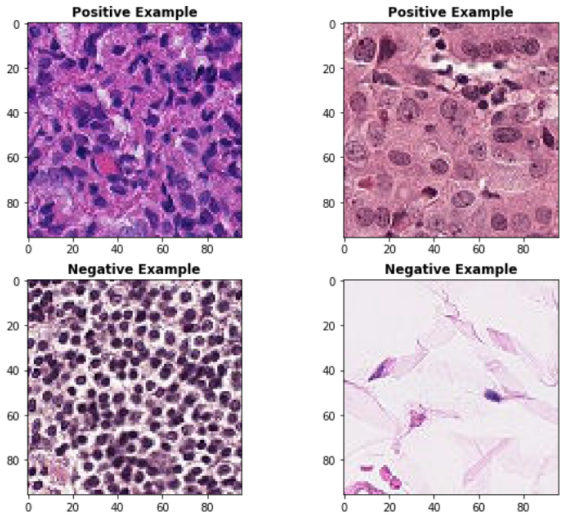


Fig. 2. Histopathologic scans of lymph node sections

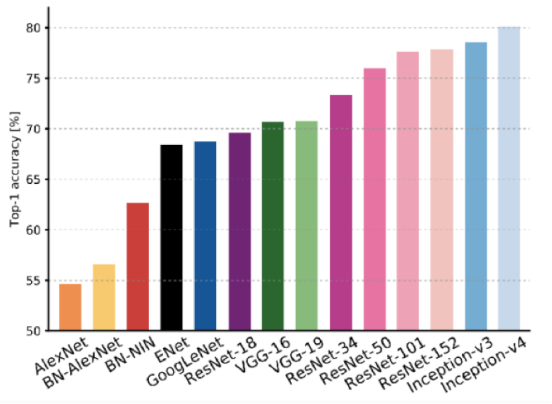


Fig. 3. Deep Neural Network Models for Practical Applications

classification dataset. The dataset consists of 277,500 color images extracted from histopathologic scans of lymph node sections. The figure 2 displays some positive and negative image samples from the dataset. Each image is a 96x96x3 color image and annoted with a binary label indicating presence of metastatic tissue. PCam provides a new benchmark for machine learning models: bigger than CIFAR10, smaller than imagenet, trainable on a single GPU. The dataset is divided into a training set of 220,000 examples, and a test set of 57,500 examples.

2

Models can easily be trained on a single GPU in a couple hours, and achieve competitive scores in the Camelyon16 tasks of tumor detection and whole-slide image diagnosis. Furthermore, the balance between task-difficulty and tractability makes it a prime suspect for fundamental machine learning research on topics as active learning, model uncertainty, and explainability.

The main message of this paper is that better image recognition systems can be build by relying on CNNs. Using metastatic cancer images as a case study, we evaluate various techniques in neural network design first. Then, we implement a shallow CNN in Tensorflow according to experiment results and train the model for image classification.

III. Literature Review

Convolutional neural networks (CNNs) have been applied to visual tasks since the late 1980s. However, despite a few scattered applications, they were dormant until the mid-2000s when developments in computing power and the advent of large amounts of labeled data, supplemented by improved algorithms, contributed to their advancement and brought them to the forefront of a neural network renaissance that has seen rapid progression since 2012.

CNN architectures come in several variations; however, in general, they consist of convolutional and pooling layers, which are grouped into modules. Either one or more fully connected layers, as in a standard feedforward neural network, follow these modules. Modules are often stacked on top of each other to form a deep model. The accuracy improved steadily in image recognition as shown in figure 3 [5].

Through these network architectures shown in figure 3, the trends demonstrated as follow: (1) Normalization methods are an important ingredient for achieving state-of-the-art performance; (2) Deeper and larger networks lead to better predictive performance; (3) Multi-scale architectures provide great predictive performance while minimizing computational demand. In this paper, we will review the modern developments in CNN, particularly focusing on the topics of normalization, dropout, architecture and transfer learning.

3.1 Normalization.

Almost all vision models employ some form of normalization throughout a network representation. The benefits are: (a) Accelerate training efficiency up to 20-fold; (b) Train models previously untrainable; (c) Boost cross-validated performance. [6][7][8][9]

The training of deep CNNs is convoluted by a phenomenon known as internal covariate shift, which is caused by changes to the distribution of each layers inputs because of parameter changes in the previous layer. This phenomenon has severe consequences, which include slower training due to lower learning rates, the need for careful parameter initializations, and complexities when training CNNs with saturating nonlinear activations. To reduce the consequences of internal covariate shift, Ioffe and Szegedy [4] proposed a technique known as batch normalization (BN).

This technique introduces a normalization step, which is simply a nonlinear transform applied to each activation, that fixes the means and variances of layer inputs. To allow integration with SGD, which also uses mini-batches during training, BN computes the mean and variance estimates after mini-batches rather than over the entire training set. Batch Normalization (BN) is a milestone technique in the development of deep learning, enabling various networks to train. However, normalizing along the batch dimension introduces problems BNs error increases rapidly when the batch size becomes smaller, caused by inaccurate batch statistics estimation. Group Normalization (GN) as a simple alternative to BN. GN divides

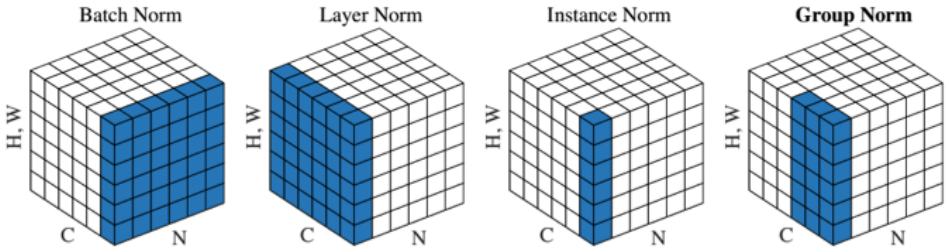


Fig. 4. Normalization methods. Each subplot shows a feature map tensor, with N as the batch axis, C as the channel axis, and (H, W) as the spatial axes. The pixels in blue are normalized by the same mean and variance, computed by aggregating the values of these pixels.

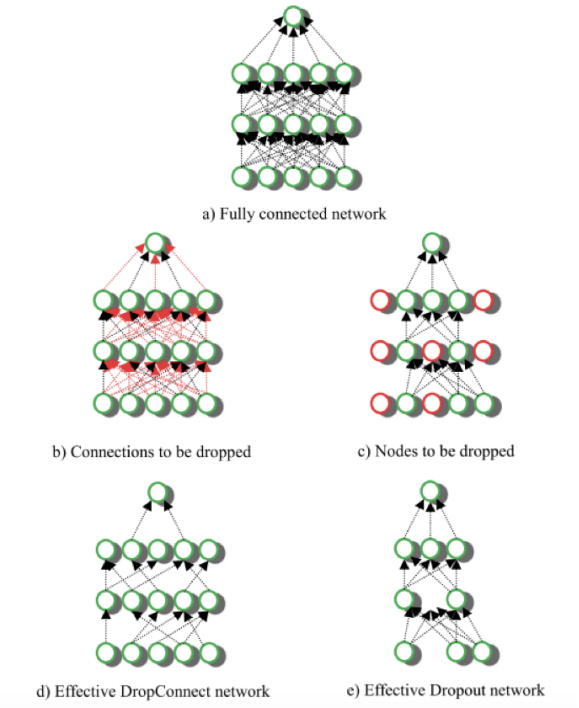


Fig. 5. The effect of Dropout on a standard feedforward network, with two hidden layers

the channels into groups and computes within each group the mean and variance for normalization. GNs computation is independent of batch sizes, and its accuracy is stable in a wide range of batch sizes. Up to now, many normalization methods have been developed as figure 4.

3.2 Dropout

To overcome overfitting, a regularization technique known as Dropout has been employed [14]. Specifically, when each training case was presented to the network during the training phase, each hidden neuron was randomly omitted from the network with a probability of 0.5. Thus, hidden neurons could not rely on other hidden neurons being present, and this prevented complex co-adaptations of features on the training data. At test time, all of the hidden neurons were used, but their outputs were multiplied by 0.5 to compensate for the fact that double the number of neurons were now active. The result of this was a strong regularization effect that significantly reduced overfitting [15]. figure5 shows the effect of Dropout on a standard feedforward network, with two hidden layers

3.3 Network Architecture

This section introduces the improvements made to the convolutional layers of deep CNNs, including the latest advancements in this regard. The convolutional layers learn the feature representations of their input images, and this makes them the main building block of deep CNNs. Thus, it is natural to try to improve this aspect of CNN architecture. The follow table summarize the various architectures of CNN performance on the ImageNet dataset.

3.3.1 ResNet

The major issue with extending many layers is the vanishing gradient problem. The ResNet introduced residual blocks, it stacks these residual blocks together where we use an identity function to preserve the gradient. These residual connections act as a gradient highway since the gradient distributes evenly at sums in a computation graph. This allows us to preserve the gradient as we go backwards. [13]

ResNets revolutionized deep architectures and now researchers are starting to use skip connections to make their architectures deeper.

3.3.2 DenseNet

It is from Facebook AI Research (FAIR), DenseNet introduced a new block called a Dense Block and stacked these blocks on top of each other, with some layers in between, to build a deep network. These dense blocks take the concept of residual networks a step further and connect every layer to every other layer. The benefit of doing this is that it encourages feature reuse, resolve the vanishing gradient problem, and (counter-intuitively) have fewer parameters overall.[12]

3.3.3 Inception

The real novelty of this network is the Inception Modules. The network itself simply stacked these inception modules together. The motivation behind these inception modules was around the issue of selecting the right filter or kernel size. We always prefer to use smaller filters, like 3*3 or 5*5 or 7*7, but which ones of these works the best? The inception modules just accept that fact that choosing this is difficult: Instead of choosing just a single filter size, choose all of them and concatenate the results. Now it was applied to GoogLeNet, they simply stack many of these inception modules on top of each other to create GoogLeNet.[10]

3.4 Transfer learning

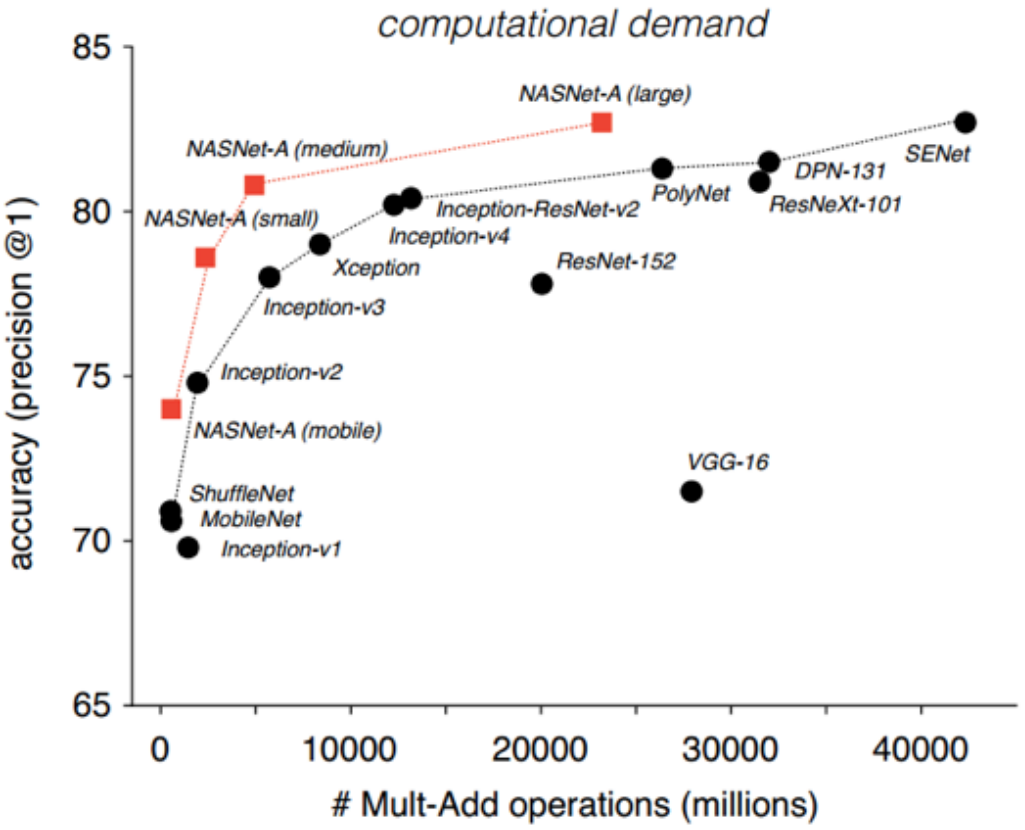


Fig. 6. The performance of NASNet

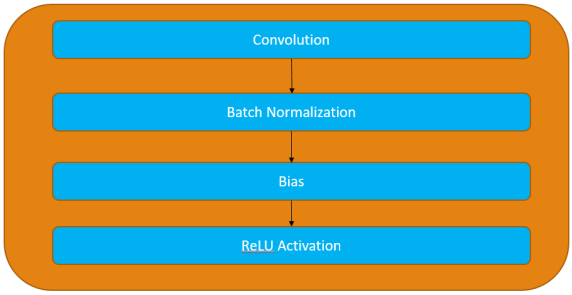


Fig. 7. Convolution Layer

After many architectures have been created, we start to consider if we can systematize the process of discovery? Perhaps computers are better than humans? Many researches have been focused on this field. A method to learn the model architectures directly on the dataset of interest has been done. As this approach is expensive when the dataset is large, they propose to search for an architectural building block on a small dataset and then transfer the block to a larger dataset. Their work is the design of a new search space (NASNet) which enables transferability. The result shows learned architectures surpass human-invented. Figure 6 shows the performance of NASNet.[11]

IV. Methods and Approach

According to the previous section, there already exists lots of complex CNN in literature for image processing and classification, such as VGG, ResNet, NasNet. Some of the networks need to take weeks for training with the help of multi-processor computer. By balancing the practical implementation and available resources, this project is designed to implement a self-constructed shallow CNN with several techniques covered in the lecture. In order to achieving high performance, the network structure and parameters are selected from properly designed experiment results. This section will give a brief description of CNN first, and then discuss considered techniques and factors.

CNN is one of the most popular class of deep neural network, especially in image processing. The main difference between CNN and regular neural network is that the input is mapped by multiple filters through convolution instead of full connected neurons. Thus, the network can be trained by adjusting the weights of the filters. As a result, CNN can not only reduce the number of neurons, but also extract the texture features of images. However, like every other neural networks, the network structure is the key bridge.

In order to design a successful CNN, several techniques covered in the lecture are considered. First of all, drop out layer is a technique to overcome overfitting. But the drop out rate needs to be examined properly. In addition, batch normalization is a newly proposed technique in network training in

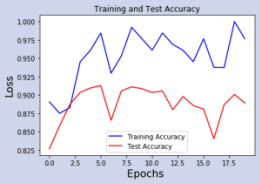
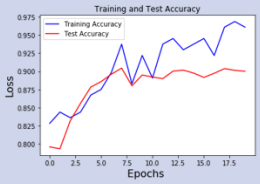
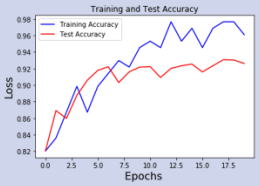
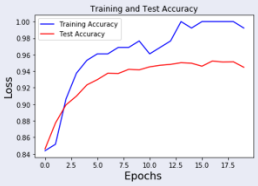
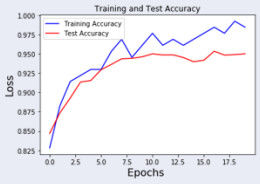
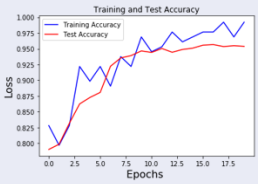
6 Conv Layers Batch Size=128 Epochs=20	No Dropout Layer	Dropout Rate=0.5	Dropout Rate=0.75
No Batch Norm	auc=0.88908 	auc=0.90039 	auc=0.92271 
Batch Norm	auc=0.94981 	auc=0.94629 	auc=0.95033 

Fig. 8. Comparison between drop rate and batch normalization

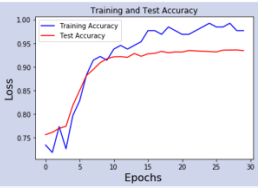
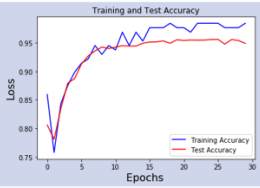
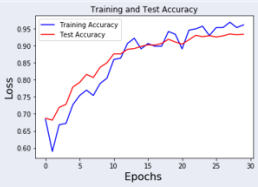
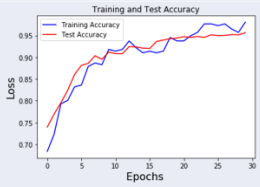
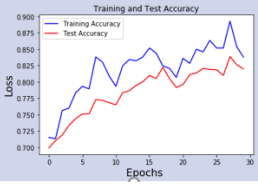
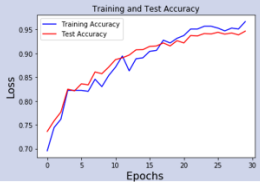
Dropout=0.75 Batch Norm Epochs=30	Conv Layers=2	Conv Layers=4		Conv Layers=6	
Batch Size=128	FAIL	auc=0.93406		auc=0.95371	
Batch Size=256	FAIL	auc=0.93018		auc=0.95469	
Batch Size=512	FAIL	auc=0.83242		auc=0.94229	

Fig. 9. Comparison between number of layers and batch size

2015 which can increase training rate. Batch size and epochs in training should also be considered since it can affect the model accuracy. Last but the most important is the number of convolution layers. By considering the mentioned techniques, each convolution layer is designed as figure 7. Experiments design and results will be given in next section.

V. Results and Analysis

From above discussion, the experiments setup and results are given in this section. The experiments are running on Kaggle platform with GPU enabled. The data set is provided by the Kaggle competition as presented in the section of introduction. The data set is divided into a training set of 220k (70%) examples, and a validate set of 57.5k (30%) examples. Each sample is a 96*96*3 color image. A positive label indicates that the center 32x32px region of a patch contains at least one pixel of tumor tissue. The experiments are implemented in Python with Tensorflow tool set.

Before finalizing the network model, multiple CNNs are being evaluated by varying the techniques stated in last section. The results are shown in figure 8 and 9. Figure 8 compares the

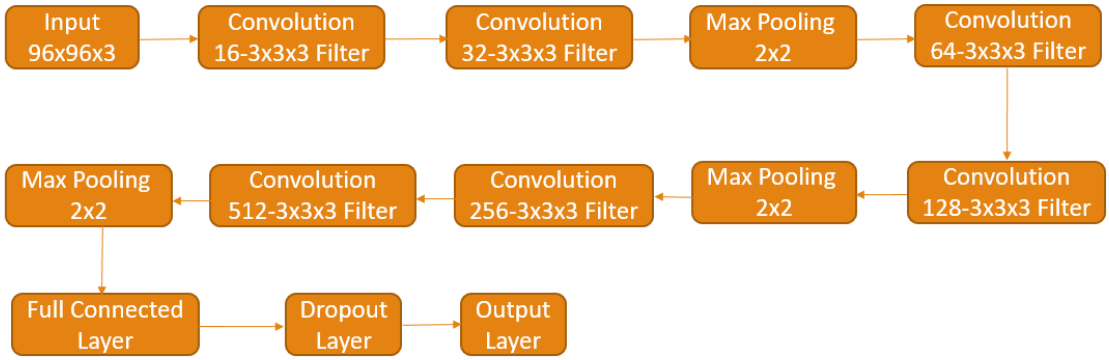


Fig. 10. CNN Architecture

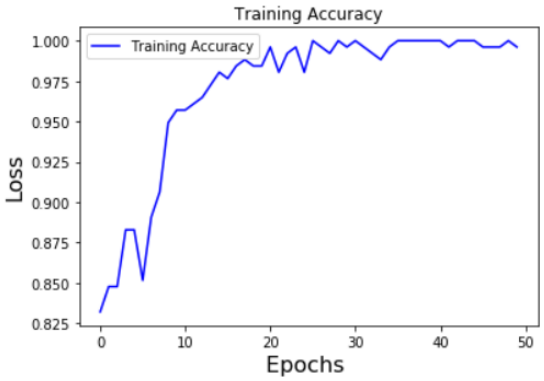


Fig. 11. Training Accuracy

training and testing accuracy by varying the drop out rate and inserting batch normalization. The drop out layer is added right after the full connected layer. The batch normalization layer is inserted before nonlinear layer as instructed in [4]. The model is trained with fixed 6 convolution layers, the batch size is set to 128 with 20 training epochs. In contrast, figure 9 compares the same accuracy by varying the batch size and number of convolution layers. The dropout rate is fixed to 0.75 with batch normalization and 30 training epochs.

From figure 8, we can see that drop our rate equals to 0.75 gives the best performance. The reason for this is that not all features are useful. The image can be accurately classified by some of essential features. Also, the drop rate affects the model fitting rate which because of the noise features. But the drop out rate should be picked case by case with careful examination. In addition, batch normalization is an extremely effective technique which increase the accuracy at least 4 percent and result in a smooth training curve. It is because that batch normalization can reduce the internal covariate shift which means maintaining the distribution of the layer input. As a result, drop out rate equals to 0.75 and batch normalization will be included in the final CNN architecture.

From figure 9, we can see that more layers give higher performance since more layers can learn more features which is the core of deep learning. Several existing architectures have even higher layers, such as VGG-16 which has 16 layers but for sure takes much longer time for training. Fail means the accuracy is lower than 80%. Moreover, batch size in model training affects the fitting rate. The figure shows that with 30 epochs, the network with batch size equals to 128 is already overfitting while the other two cases are underfitting. And batch size equals to 256 gives the best performance among all these 3 cases. Thus 256 batch size and 6 convolution layers will be considered in the final CNN model with more training epochs.

As a result, the final network is designed as figure 10. There are totally 6 convolution layers, 3 max pooling layers, 1 dense layer and 1 dropout layer. Based on this model, the network is being trained with 40 epochs and the training accuracy is presented as figure 11. The final result is being submitted to Kaggle for evaluation. The final test accuracy is 85.95% that is evaluated by Kaggle while the highest accuracy of the competition is 100%.

VI. Conclusion

In summary, we briefly review the machine learning in medical image processing first. And then do a survey on the update to date CNNs architectures. Based on the learned techniques, we implement several experiments to evaluate their effects and design a shallow CNN according to the results. From this project, we can see that the architecture of the network makes a big difference, and the parameters and architecture need to be considered carefully. As a result, ResNet in recent years becomes a potential algorithm in constructing CNN since it is an algorithm that searches for the best neural network architecture.

References

[1] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, *Gradient-based learning applied to document recognition*, Proc. IEEE, vol. 86, no. 11, pp. 2278-2324, 1998.

[2] H.S. Yoon, J. Soh, Y.J. Bae, H.S. Yang, *Hand gesture recognition using combined features of location angle and velocity*, Pattern Recogn., vol. 34, no. 7, pp. 1491-1501, 2001.

[3] A. Krizhevsky, I. Sutskever, G. Hinton, *Imagenet classification with deep convolutional neural networks*, Adv. Neural Inf. Process. Syst., pp. 1097-1105, 2012.

[4] S.Loffe and C. Szegedy, *Batch Normalization: Accelerating Deep Network Training by Reducing INternal Covariate Shift*, arXiv:1502.03167, 2015.
[5] A Canziani, A Paszke, E Culurciello, *An Analysis of Deep Neural Network Models for Practical Applications*. arXiv:1605.07678.
[6] J Ba, J Kiros, GHinton, *Layer Normalization*. arXiv:1607.06450
[7] D Ulyanov, A Vedaldi, V Lempitsky, 2016 *Instance Normalization: The Missing Ingredient for Fast Stylization*. arXiv:1607.08022
[8] M Ren, R Liao, R Urtasun, F Sinz, R Zemel,2016 *Normalizing the Normalizers: Comparing and Extending Network Normalization Schemes*. arXiv:1611.04520
[9] Y Wu, K He, 2018, *Group Normalization*. arXiv:1803.08494
[10] B Zoph, V Vasudevan, J Shlens, Q Le, 2018 *Learning Transferable Architectures for Scalable Image Recognition*. arXiv:1707.07012v4
[11] C Szegedy, W Liu, Y Jia, P Sermanet, S Reed, D Anguelov, D Erhan, V Vanhoucke, A Rabinovich, 2014, *Going Deeper with Convolutions* . arXiv:1409.4842
[12] G Huang, Z Liu, L Maaten, K Weinberger, *Densely Connected Convolutional Networks* . arXiv:1608.06993
[13] K He, X Zhang, S Ren, J Sun *Deep Residual Learning for Image Recognition*. arXiv:1512.03385
[14] N Srivastava, G Hinton, A Krizhevsky, ISutskever, R Salakhutdinov, 2014 *Dropout: a simple way to prevent neural networks from overfitting* . Journal of Machine Learning Research 15 (2014) 1929-1958
[15] A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov, *Improving neural networks by preventing co-adaptation of feature detectors* . arXiv:1207.0580