Using Deep Learning for Diabetic Retinopathy Detection

Project Group 28 – Vivian Zhong (MSCA)

Table of Contents

Presentation Video Link (either one should work):
https://youtu.be/mSpgEJEbYz8
https://drive.google.com/file/d/12_gKze3nl3UxjQGb3_NwJ3VsZj7p9672/view?usp=sharing

Abstract

Diabetic retinopathy is a common complication of diabetes and a leading cause of blindness. Early detection and treatment can prevent vision loss and improve patient outcomes. In this project, we propose to use deep learning techniques, specifically convolutional neural networks (CNNs), to detect diabetic retinopathy in eye images. We will also utilize a pre-trained model, EfficientNetB0, to improve the performance of our CNN. The project has achieved an accuracy of 0.74 in classifying diabetic retinopathy. Our approach will involve training the CNN on a large dataset of eye images, both with and without diabetic retinopathy, and using the trained model to classify new images. We will also perform a thorough evaluation of the model's performance and identify any potential areas for improvement. Overall, our project aims to contribute to the development of effective and efficient tools for the early detection and treatment of diabetic retinopathy.

Introduction

What is Diabetic Retinopathy

Diabetic retinopathy is a common complication of diabetes that affects the retina, the light-sensitive tissue at the back of the eye. If left untreated, it can cause vision loss and blindness.

Business Value of Diabetic Retinopathy Detection

According to VersantHealth.com[i], diabetes costs the U.S. an estimated $327 billion annually, with $237 billion coming from direct medical costs and $90 (htt)billion coming from decreased productivity. Nearly 30 percent of diabetics suffer from diabetic retinopathy. Diabetes-related blindness costs can total more than $500 million per year. Patients with diabetic retinopathy usually have higher treatment costs than those with normal diabetes.

Early detection and treatment of diabetic retinopathy can help prevent these complications and preserve vision. However, current identification involves high resources and requires high expertise and professional equipment. If invested in diabetic retinopathy detection, we will have a chance to detect this disease at an early stage using just retina images, which would require less expertise and would help patients who live in a place where resources for DR are scarce. If the model can identify DR correctly at an early stage, it would greatly reduce the treatment costs, as well as improve the quality of life for those patients. This tool can be useful for both insurance companies and healthcare providers, as it will greatly reduce medical costs. It's an asset for diabetic patients as well.

Solution

Therefore, the solution that we come up with is to use the retina image and detect early presence of DR. We hope to achieve a higher accuracy rate and put the model into actual use.

Contribution

Overall, the business value of diabetic retinopathy detection lies in its potential to improve the health and quality of life for people with diabetes and to help prevent the financial and social costs associated with vision loss. The goal of this project is to use high-resolution retina images to identify the presence of diabetic retinopathy, especially for those at early stages.

Literacy Reviews

1. *Real-time diabetic retinopathy screening by deep learning*[ii]

The researchers take real-time individual eye images for diabetic retinopathy detection and achieve an accuracy of 94.7%, sensitivity of 91.4%, and specificity of 95.4%. This research shows that the actual implementation of DR detection using deep learning methods is viable in the real-world. One challenge that this research has is most referred patients are referred for referrable diabetic retinopathy or diabetic macular oedema, compared with low visual acuity, and ungradable images. Therefore, it might not be representative of all DR patients, especially for early detection.

2. *General Deep Learning Model for Detection of Diabetic Retinopathy*[iii]

The researchers use a general pre-trained CNN model for the detection of DR in this research study. They have tested the result with several pre-trained models, including NASNET_Large to identify the DR and no-DR. They also apply SOMTE synthetic samples to identify the severance of DR presence. The idea is great since indeed for early detection, the most important step is to identify the presence of DR instead of classifying the severance. They are able to achieve an accuracy of 0.85 for DR/no-DR detection and an accuracy of 0.83 in classifying DR into five categories. They have trained relatively big datasets, with more than 70,000 images, and the accuracy rate is not high enough, especially for DR/no-DR detection, which I believe is the most important part of diabetic retinopathy detection.

3. *K-means algorithm for diabetic retinopathy detection*[iv]

This research study uses k-means and fuzzy c-means clustering for DR detection. The benefits of this technique are that there are a lot of retina image data that are unlabeled. If we want to do supervised learning, then we are given less data since we need to have labeled dataset. However, k-means clustering and fuzzy c-means clustering are part of unsupervised learning techniques, and thus can use unlabeled data as well. The clustering technique is also simple enough, so the training and computation time is shorter, which improves efficiency. The model can identify dark spots in the retina with less computation time. However, this model cannot separate macular and thus is not practical for general use.

Problem Description

The problem here is to classify retina images into 5 groups:
0 - No DR
1 - Mild

2 - Moderate
3 - Severe
4 - Proliferative DR

Because this is an image classification problem, we plan to use CNN or pre-trained models such as EfficientNet to solve this problem. This approach is similar to the approaches used in the second paper we discussed, but with a different pretrained model. We think our approach is better because EfficientNet usually have better accuracy than other CNN models, with optimized FLOPS.

For the problem, we plan to first use CNN and EfficientNet to fit the data, and fine-tune the hyperparameters. We will try to classify the images into 5 classes and check the model's performance.

Model Description

We plan to use basic CNN model as well as the pretrained model with EffcientNetB0. The reason we use this model is because it's an image classification problem and usually CNN works best for this type of problem. We choose to use EfficientNet because it's deeper than the simple pretrained model such as ResNet or VGG16. We hope it performs better because the dataset that we're using is very complex and while we're trying to fit it on a CNN model, the model underfits the data, meaning we need more complex structure.

For CNN, we have 2 convolutional & maxpooling layers. Then we use several dense layers with relu activation to fit the data and finally we use dense layers with 5 neurons to output the final classification.

First CNN model:

conv2d_10 (Conv2D)	(None, 254, 254, 258)	7224
max_pooling2d_10 (MaxPooling2D)	(None, 127, 127, 258)	0
conv2d_11 (Conv2D)	(None, 125, 125, 64)	148672
max_pooling2d_11 (MaxPooling2D)	(None, 62, 62, 64)	0
flatten_4 (Flatten)	(None, 246016)	0
dropout_4 (Dropout)	(None, 246016)	0
dense_12 (Dense)	(None, 64)	15745088
dense_13 (Dense)	(None, 32)	2080
dense_14 (Dense)	(None, 5)	165

CNN with fine tuning:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 399, 399, 256)     3328

 max_pooling2d (MaxPooling2D  (None, 199, 199, 256)    0
 )

 conv2d_1 (Conv2D)           (None, 198, 198, 256)     262400

 max_pooling2d_1 (MaxPooling  (None, 99, 99, 256)      0
 2D)

 conv2d_2 (Conv2D)           (None, 98, 98, 128)       131200

 flatten (Flatten)           (None, 1229312)           0

 dense (Dense)               (None, 1024)              1258816512

 dense_1 (Dense)             (None, 1024)              1049600

 dense_2 (Dense)             (None, 512)               524800

 dense_3 (Dense)             (None, 512)               262656

 dense_4 (Dense)             (None, 256)               131328

 dense_5 (Dense)             (None, 256)               65792

 dense_6 (Dense)             (None, 64)                16448

 dense_7 (Dense)             (None, 16)                1040

 dense_8 (Dense)             (None, 5)                 85

=================================================================
Total params: 1,261,265,189
Trainable params: 1,261,265,189
Non-trainable params: 0
_________________________________________________________________
```

For pretrained model, we set all layers in the model to be non-trainable except the final convolutional layers, which is "top-conv" and add several dense layers after the pretrained model to further fit the data.

The model structure is as below:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 efficientnetb0 (Functional)  (None, 13, 13, 1280)     4049571

 flatten (Flatten)           (None, 216320)            0

 dense (Dense)               (None, 512)               110756352

 dense_1 (Dense)             (None, 256)               131328

 dense_2 (Dense)             (None, 256)               65792

 dense_3 (Dense)             (None, 128)               32896

 dense_4 (Dense)             (None, 64)                8256

 dense_5 (Dense)             (None, 5)                 325

=================================================================
Total params: 115,044,520
Trainable params: 110,994,949
Non-trainable params: 4,049,571
_________________________________________________________________
```

One drawback of the model is that it only used supervised learning, because in this project we have a large, labeled dataset. However, there are also datasets for diabetic retinopathy which are not labeled. We should experiment with self-supervised learning such models such as VAEs to classify the images and see how it performs compared to the supervised learning method. Another drawback is that because we use relatively complex models in this project, if we apply this model into real-world for real-time classification (similar to the approaches taken in the first paper), then it might need extra computation time to classify the results. Therefore, a simpler approach should be tested to be more practical in real-life.

Data and Experiment Results

Dataset Description

The dataset is taken from Kaggle: https://www.kaggle.com/competitions/diabetic-retinopathy-detection/data. The dataset is very large, with a total of more than 80G. However, for training the data I only used the training data, which is about 35G. The training data consists of a total of more than 30,000 images and the images are classified into 5 groups.

Data Preprocessing

The dataset includes RGB image file and a csv file listing all file names and the groups they belong to. For the data preprocessing, I resize all images into (400, 400) size and add the full file path to the dataframe that we use in the project. I also split all data into 85% training data set and 15% validation dataset. The training dataset has about 30,000 files, while the validation dataset has about 5000 image files.

Results

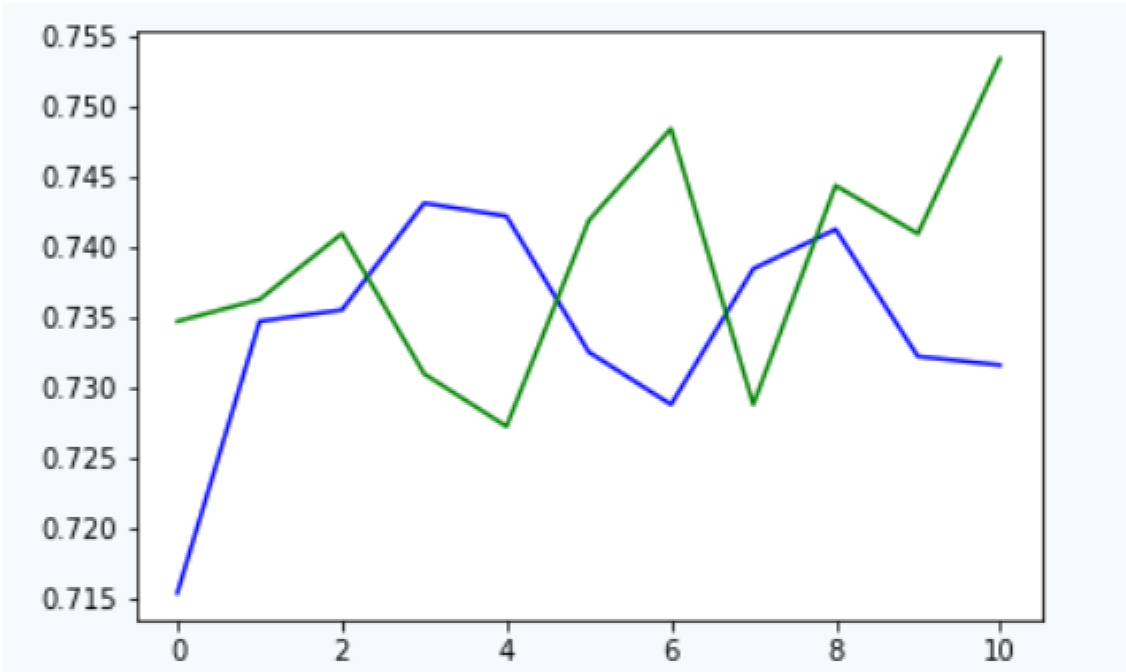
The best result I got is around 0.75 accuracy for the validation dataset. The test data on the Kaggle website is too large so I decided to only use validation accuracy for the general guidance.

Models	Accuracy
CNN	0.7534
CNN with fine tuning	0.7350
EfficientNetB0	0.7287

Although it seems that the original CNN has the best performance, it is because the data size is too small. If we have a larger training data, CNN with fine tuning and EfficientNetB0 should outperform the original CNN network.

Fine Tune Parameters

For the first model that we build using CNN model. The training accuracy as well as the validation accuracy is low, at around .75 accuracy rate for both training and validation dataset.



Because this clearly shows the sign of underfitting, I tried to use a more complex architecture by:

- Increase filter layers in convolutional layers - trying to preserve more information
- Add more Dense layers
- Get rid of dropout layers, regularizers, and early stopping
- Decrease batch size (increase step per epoch) to increase computation speed
- Increase epoch - more training

However, with less maxpooling layers and 9 Dense layers and 100 epochs for training the data, the result is not improving.

This is probably because the dataset is not large enough. We currently have 30,000 image files in the dataset while the model might require more datasets for training.

I also use EfficientNetB0 as the pretrained model to fit the data. Due to the time limit (the model is taking a long time to train, even on a better GPU), the result is slightly better than the CNN sequential model, of an accuracy of 0.74. However, it is still not ideal enough. Again, the problem might be that I do not have large enough data for training. For future works, it might need data augmentation to increase the training data size.

Business Usage

I save the neuron network's parameters and weights into an h5 file. The parameters can be used to fit future images. For business usage, the patient can do a retina scan and then send the image file to the model. The model will then classify the image into one of the 5 groups and return the result to the patient. The patient or healthcare provider can use the classification from the model as general guidance on identification. This is important because the manual identification of diabetes retinopathy requires the expertise of the doctors, and using this model, the patient can have general diagnosis on whether they should seek professional help and prevention at an early stage. In this way, they can avoid eyesight loss or decreases in life quality. After deploying this model, it also needs regular monitoring and updates in order to make sure it performs well in a clinical setting. This may involve retraining the model on new data or incorporating additional features to improve its performance.

Conclusions and Recommendations

We have achieved 0.74 accuracy on the validation data, meaning only 74% of the time we can accurately classify the DR detected in the image file. While this accuracy is not sufficiently high, we can still gain some insights from this project. First, the model should be trained on a larger dataset, since diabetic retinopathy detection is a rather complex problem, it will need a larger training data, and the image quality should be preserved (meaning we should not resize it to a too small size). Also, we will need greater computation resources as well as more training time. After achieving higher accuracy, the model can be put into practice, and if the patient inputs an eye image, the model should be able to tell the patient whether they should worry about the problem of diabetic retinopathy or should they seek professional help for early prevention. This would greatly increase the probability that DR is detected at an early stage, so patients would have less treatment costs and preserve their life quality.

---

[i] https://versanthealth.com/blog/the-financial-impact-of-diabetic-retinopathy/

[ii] Ruamviboonsuk, Paisan, et al. “Real-time Diabetic Retinopathy Screening by Deep Learning in a Multisite National Screening Programme: A Prospective Interventional Cohort Study.” *The Lancet Digital Health*, 1 Apr. 2022, www.thelancet.com/journals/landig/article/PIIS2589-7500(22)00017-6/abstract.

[iii] Chen, PN., Lee, CC., Liang, CM. et al. General deep learning model for detecting diabetic retinopathy. BMC Bioinformatics 22 (Suppl 5), 84 (2021). https://doi.org/10.1186/s12859-021-04005-x

[iv] Gogula, S. V., Divakar, C.h, Satyanarayana, C.h, & Rao, A. A. (2014). A diabetic retinopathy detection method using an improved pillar K-means algorithm. Bioinformation, 10(1), 28–32. https://doi.org/10.6026/97320630010028