Dental X-ray Classification

Adra Data Science Evaluation Challenge

Divyansh Chahar

https://www.linkedin.com/in/divyanshchahar/

https://github.com/divyanshchahar

Sunday 29th November, 2020

1 Introduction

Often in dental X-rays it is difficult to identify the number of roots i.e. the structure of inflamed, infected or injured pulp while performing root canal treatment as the roots could be overlapping or might have other anomalies. If any infection is left untreated because of an unknown root or missed anatomy, the infection can fester and might result in even more severe consequences.

Why is a root canal treatment needed ?

To understand the need of a root canal treatment, we first need to understand the anatomy of a healthy tooth. As can be seen in figure 1-1, tooth's anatomy can be divided into two main structure named root and crown. The root comprises of pulp and various other parts. If due to any reason the pulp becomes inflamed, infected of injured, it needs to be removed. The procedure is performed when the patient is under local anesthesia.

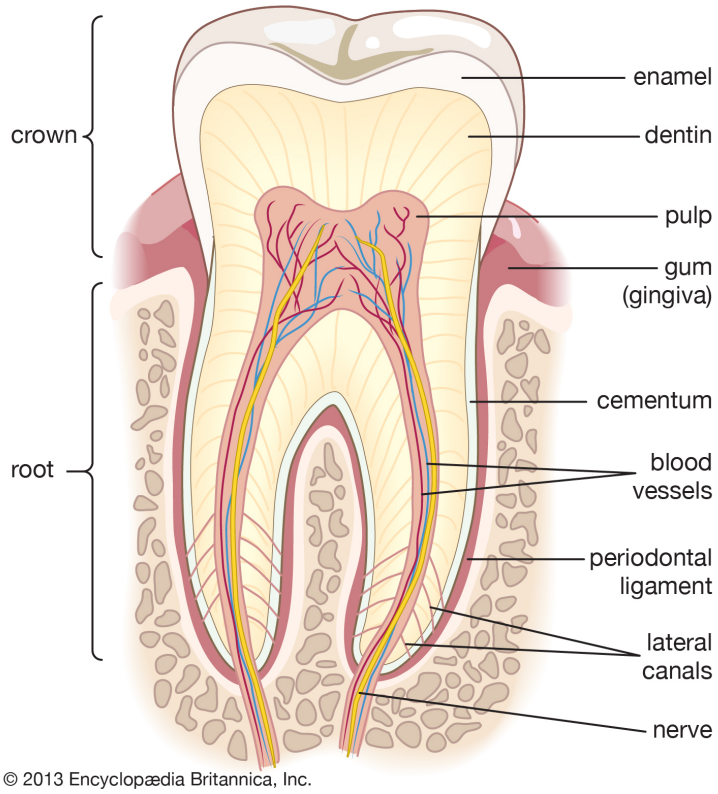


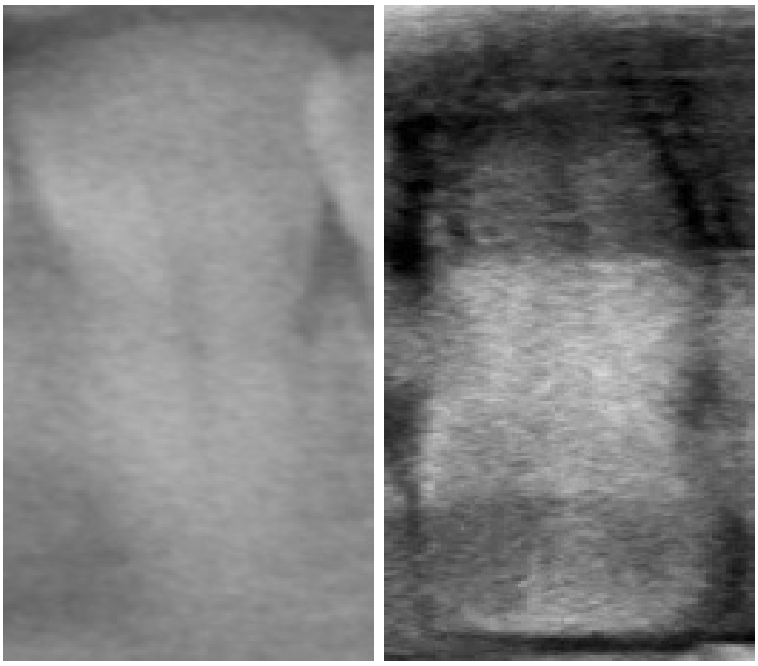
Figure 1-1: Anatomy of a tooth. Source:Internet

2 Challenges in the Data

Data forms the basis of any deep learning algorithm. Thus if the data is bad or dirty the algorithms will not give proper results. The dataset provided for this task was no exception.

The challenge in this data set came in the shape of unidentifiable roots and dental filling. These challenges were present in both classes of data.

Two images of particular concern from the class of images with one root were *photo35* and *photo50*



(a) tooth35 **(b) tooth57**

Figure 2-2: Images with difficult to identify roots from class of figures with one root

Similar images were present in class of images with 2 or more roots.

(a) tooth12 **(b) tooth20**

Figure 2-3: Images with difficult to identify roots from class of figures with multiple root

As can be seen in the figure above *tooth12* has a dental implant and *tooth20* has almost no visible root, this makes it difficult for the algorithm to learn the pattern of multiple roots.

Important note about data

There were several images which in which root was barely visible, however these images were not removed because the obective of this code was to identify weather there is single root or there are multiple roots.

It must also be noted that the daatset is not perfectly balanced.

3 Model Architecture and Coding Strategy

To tackle this challenge convulation neural networks were used. Several architectures were used but 3 rather simple architectures were selected they will be referred to as - cnn1, cnn2 and cnn3.

Various different strategies were used to test these 3 architectures including image data augmentation.

These strategies and architectures are discussed below.

cnn1-Architecteture

This was the simplest architecture used in the model. It consisted of following layers

- **Layer-1**: Conv2D layer with 32 neurons, 3x3 filter, relu activation function and padding enabled
- **Layer-2:** Fully Connected Dense Layer with 32 neurons and relu activation function.
- **Layer-3**: MaxPooling layer with 3x3 pooling size
- **Layer-4**: Flatten Layer
- **Layer-5**: Dense layer with 2 neurons and sigmoid activation function.

cnn2-Architecteture

This was the the second neural network used in classification, it had the following layers

- **Layer-1**: Conv2D layer with 32 neurons, 3x3 filter, relu activation function and padding enabled
- **Layer-2:** Fully Connected Dense Layer with 32 neurons and relu activation function.
- **Layer-3**: Conv2D layer with 64 neurons, 3x3 filter, relu activation function and padding enabled
- **Layer-4:** Fully Connected Dense Layer with 64 neurons and relu activation function.
- **Layer-5**: MaxPooling layer with 3x3 pooling size

- **Layer-6**: Flatten Layer
- **Layer-7**: Dense layer with 2 neurons and sigmoid activation function.

cnn3-Architecteture

The third neural network used in the model had the following architecture:

- **Layer-1**: Conv2D layer with 32 neurons, 3x3 filter, relu activation function and padding enabled
- **Layer-2:** Fully Connected Dense Layer with 32 neurons and relu activation function.
- **Layer-3**: Conv2D layer with 64 neurons, 3x3 filter, relu activation function and padding enabled
- **Layer-4:** Fully Connected Dense Layer with 64 neurons and relu activation function.
- **Layer-5**: Conv2D layer with 128 neurons, 3x3 filter, relu activation function and padding enabled
- **Layer-6:** Fully Connected Dense Layer with 128 neurons and relu activation function.
- **Layer-7**: MaxPooling layer with 3x3 pooling size
- **Layer-8**: Flatten Layer
- **Layer-9**: Dense layer with 2 neurons and sigmoid activation function.

First Strategy

Standard run practices for binary classification were employed during this approach. All the 3 architectures were run using a batch size of 32, adam optimizer and binary cross entropy loss function.

Second Strategy

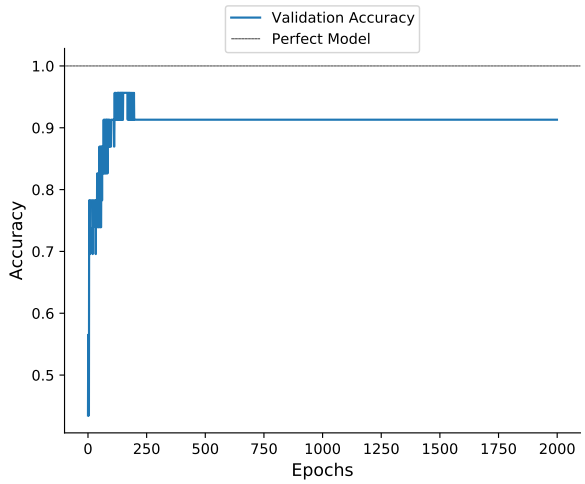
It was noticed that during some epochs the validation accuracy reached 100 percent, thus there were indications of over fitting. To tackle this batch size was increased from 32 to 64, this approach was employed instead of using Dropout layer because of low availability of data.

Third Approach

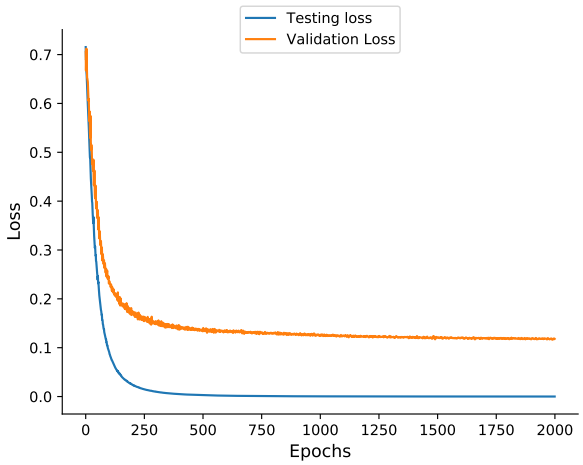
Due to low availability of data, some image augmentation techniques were used to generalize the model and develop a more robust model.

4 Model Performance

4.1 cnn1



(a) variation of validation accuracy with epochs

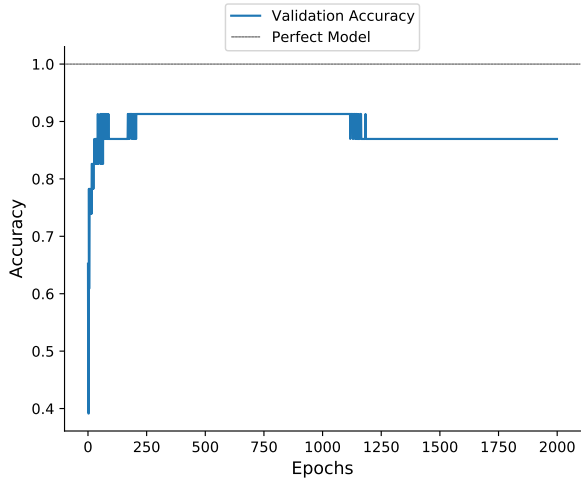


(b) variation of loss with epochs

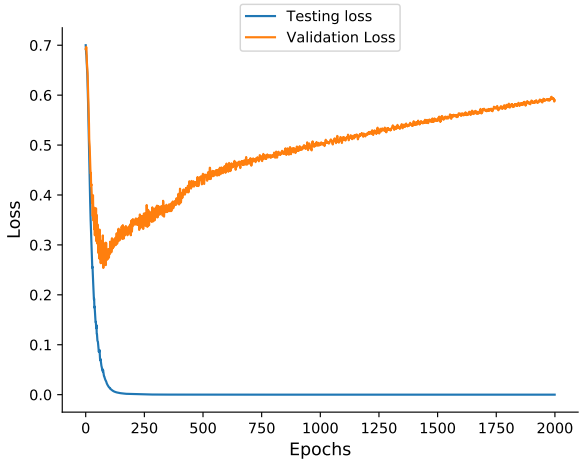
Figure 4-4: Performance Curves for cnn1

As can be seen from figure 4-4, cnn1 architecture does not suffers from overfitting, it can also be observed from testing and validation loss curves that the model is still learninng as validation loss has not flattened but Validation Accuracy has become stagnant, hence there is no significant gain in validation accuracy even as the model continues to learn.

4.2 cnn2



(a) variation of validation accuracy with epochs

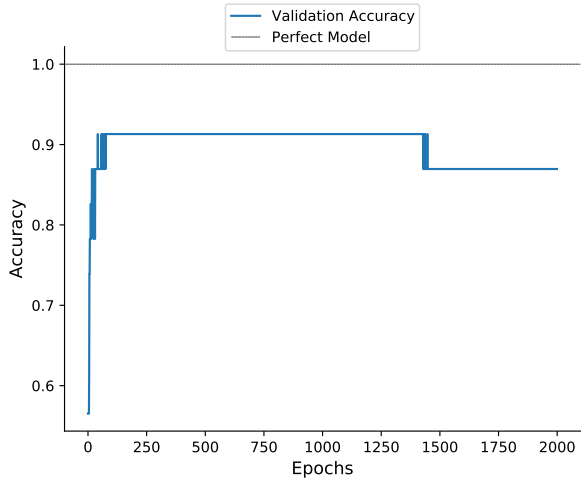


(b) variation of loss with epochs

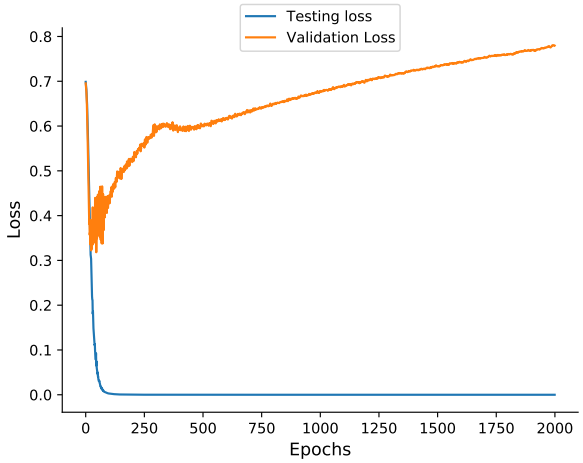
Figure 4-5: Performance Curves for cnn2

It could be seen from figure 4-5, that cnn2 architecture results in overfitting, also after 1250 epochs the accuracy drops suddenly and the saturates.

4.3 cnn3



(a) variation of validation accuracy with epochs

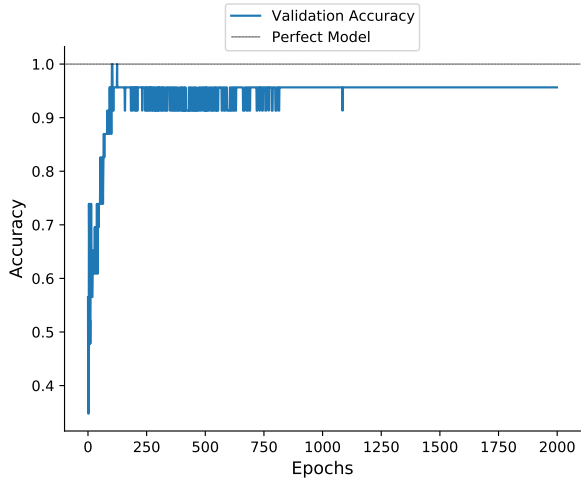


(b) variation of loss with epochs

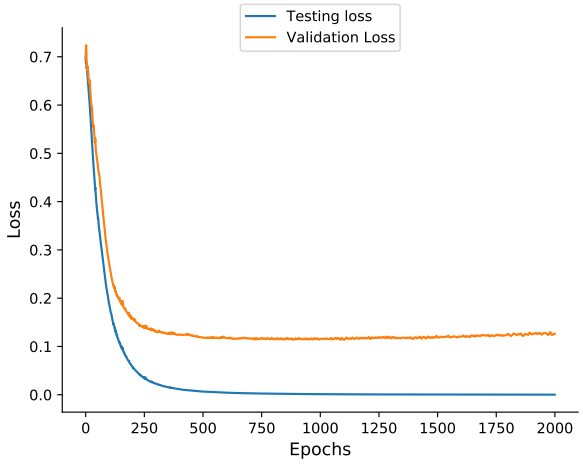
Figure 4-6: Performance Curves for cnn3

Overfitting can be observed even for cnn3 architecture. The drop in validation accuracy occurs at nearly 1500 epochs. This architecture delays the accuracy saturation point but results in more over fitting than the cnn2.

4.4 cnn4



(a) variation of validation accuracy with epochs

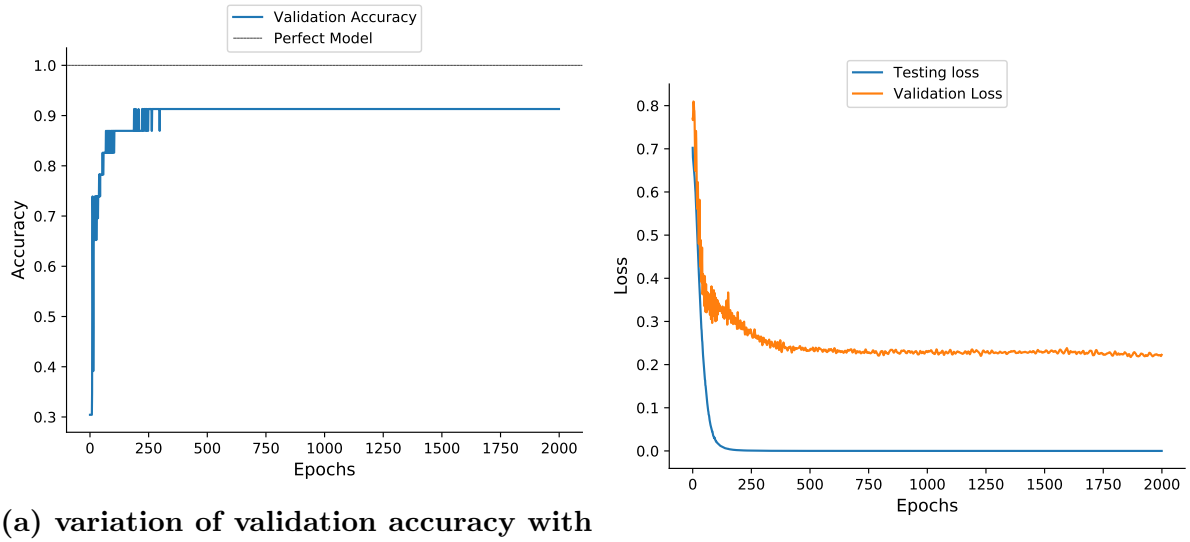


(b) variation of loss with epochs

Figure 4-7: Performance Curves for cnn4

cnn4 borrows its architecture from cnn1 but with an increased batch size. Amongst the three architectures used for this project, the architecture of cnn1 is the simplest and gives the highest accuracy. cnn4 is the only architecture which has achieved 100 precent accuracy, even though this level accuracy is not permanent, the accuracy value drops down and saturates to a lower value. As compared to previous models accuracy saturation for cnn4 results in the highest validation accuracy. Although the accuracy is very promising this model is also overfitting, there is slight upward trend in the validation loss curve.

4.5 cnn5



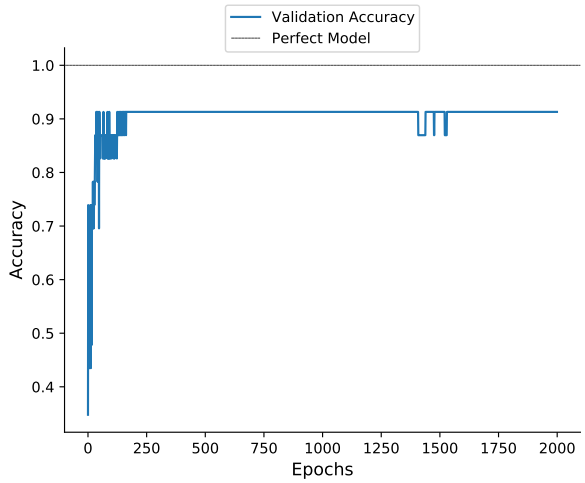
(a) variation of validation accuracy with epochs

(b) variation of loss with epochs

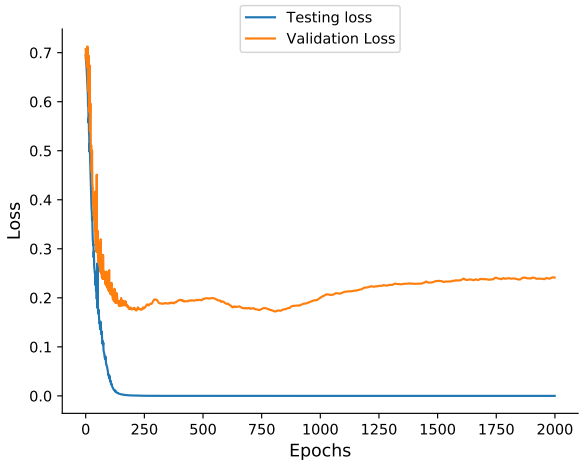
Figure 4-8: Performance Curves for cnn5

Just like cnn4, cnn5 also borrows its architecture from a previous model. cnn5 has the same architecture as cnn2 but with a increased batch size of 64 images. Although no signs of overfitting can be observed, the validation loss is the highest among all the the models.

4.6 cnn6



(a) variation of validation accuracy with epochs

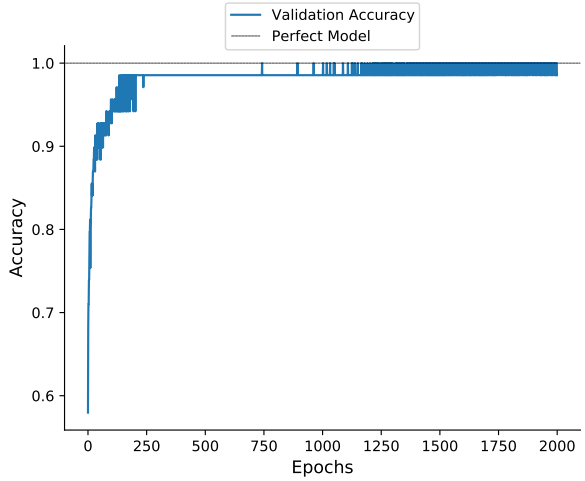


(b) variation of loss with epochs

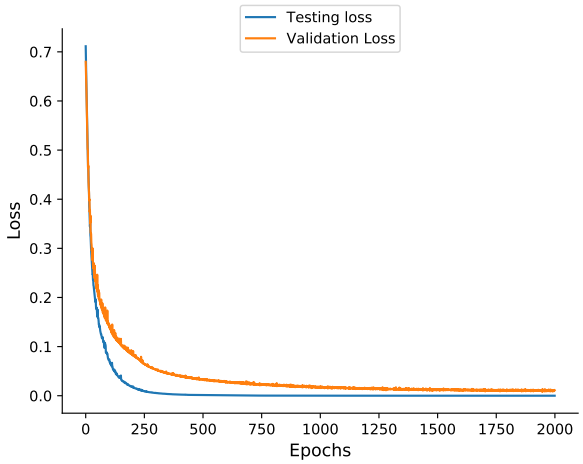
Figure 4-9: Performance Curves for cnn6

cnn6 shares its architecture with cnn3 but with an increased batch size of 64. Clear signs of overfitting can be seen from after 750 epochs.

4.7 cnn7



(a) variation of validation accuracy with epochs



(b) variation of loss with epochs

Figure 4-10: Performance Curves for cnn7

cnn7 has the same architecture as that of cnn1 and is executed with a batch size of 32, however image data augmentation was used with this model to over come the hurdle of less data. It can be seen that validation

accuracy is very high and fluctuates between 100 percent and 98 percent. As can be clearly seen from the loss curves that this is the best fitting model that has used on this data set.