AI Knee Osteoarthritis Diagnosis

Problem Definition:
As an athlete for the majority of my life, I understand how much force, power and energy our knees deal with on a consistent basis and how much recovery is required to prolong their health. However, knee osteoarthritis is still extremely prevalent in athletes and has been affecting athletes at higher rates whilst beginning at earlier points in their lives (almost 2x as likely in athletes relative to non-athlete control groups). This degenerative condition occurs when the cartilage in one's knee degrades causing pain and swelling whilst continuing to worsen as time goes on. Whilst this may not be a condition that is life-threatening, it is a chronic pain that makes all aspects of life more difficult and cumbersome whilst constantly worsening if appropriate treatment isn't provided.

Alongside this, our Canadian healthcare system has been overwhelmed for a while with large backlogs and long wait times. This means that conditions that aren't absolutely critical or life-altering (e.g. knee osteoarthritis) are not prioritized and are often given the longest waits. This allows these conditions to worsen and diagnosis to be delayed leading to worse outcomes for patients of this condition. Therefore, any tool which can speed up this process and reduce the load on the doctors responsible for treating and diagnosing this condition would have a sizable impact on those with knee osteoarthritis and help improve their lives through earlier diagnosis and faster treatment.

This is why I wanted to create an AI-powered tool which can view knee X-rays and determine whether or not one has knee osteoarthritis and the magnitude of the condition. Whilst this tool shouldn't be used as a diagnosis on its own, it can be a valuable tool to quickly provide medical professionals with a predicted diagnosis to help prioritize patients and direct them to the correct specialist/treatment facility.

AI Methodology:
When designing this model, I first began by determining what data I would have access to to train this model. I found a dataset on Kaggle which had two medical experts classify knee X-rays as either normal, doubtful, mild, moderate or severe which led me to decide upon creating a model that would classify knee X-rays in the same manner.

Since this model would be dealing with image classification, the best model type to use for this task was determined to be a convolutional neural network. These models are well suited for image classification as they are very good at determining and recognizing complex patterns from input images.

This model should take an input image and return a classification/diagnosis (normal, doubtful, mild, moderate or severe) and was designed to do so.

Implementation Details:

When first beginning to implement the model described above, the training dataset and testing dataset had to first be determined. Since there was the classifications given by two medical experts in the Kaggle dataset chosen, the X-rays and classifications of medical expert #1 was used for training and those of medical expert #2 were used for testing and validation. This would give two separate, unrelated sets of data which can be used to validate the accuracy of the model and ensure overfitting does not occur.

Once the datasets were decided, a method of implementation had to be chosen. I chose to use a custom convoluted neural network made in PyTorch trained in a Google Collab notebook as it would be easy to design and fast to train (using google GPUs).

The network design used the following structure as my testing determined that this would be the most effective.

convolution layer -> pooling layer -> convolution layer -> pooling layer -> linear layer (RELU) -> linear layer (RELU) -> linear layer (LOG SOFTMAX)

This model was then trained for 10 epochs (determined to be the ideal amount by my testing) and then exported as a .pt file to be used in a web-based interface.

This web-based interface was made in Flask for its simplicity (and scalability) and allows medical professionals to input an image and receive an immediate, automatic diagnosis. This works by running the image through the pre-trained model (for efficient and quick responses) and returning the result to the user.

Learning Journey:
Throughout this process I learnt how to better use and optimize convolutional neural networks for the most accurate possible result (as usage in the medical industry would require high accuracy). I learnt through the testing process what types of layers and structures could be used to increase accuracy whilst also preventing overfitting.

I also got more comfortable with the PyTorch library due to the many additional hours I have gained in experience throughout this project. I have learnt better ways to do things with the PyTorch library and be more efficient.

I also learnt how medical classification from scans/images is a great use case for modern day machine learning and AI as it is useful and actually feasible in the vast majority of cases.

Challenges:
The two biggest challenges when designing this model was training speed and accuracy.

To fix training speed, I swapped to using Google Collab and the GPUs provided by it to speed up the training process.

To improve model accuracy, I had to spend a large amount of time rewriting and redesigning the layers (including modifying the input data) of the neural network to find one that works ideally for this use case.

Model Evaluation Metrics:
To evaluate the model throughout the development process, I measured the accuracy of the model on the testing dataset. I kept on improving the model until it reached an accuracy suitable for use in the medical industry. The final accuracy of the model on the testing dataset was 95% which shows that the model is very effective.