Automated Detection of Pathologies in Knee MRI Scans

Miles Wang S.B. Electrical Engineering
Fall 2019 - Spring 2020
Flavio Calmon, Benjamin Odry (Covera Health)

The Project

Background and Motivation

In 2010, more than 10 million visits to the doctor's office were due to knee pain or injury [8]. When injuries are more severe, oftentimes radiologists will be asked to read MRI (Magnetic Resonance Imaging) scans or CT (Computed Tomography) scans to locate and diagnose abnormalities. Common abnormalities found in the knee using MRI scans include ACL (Anterior Cruciate Ligament) and/or meniscal tears. The ACL is one of the major ligaments which cross the middle of the knee, connecting the femur to the tibia and helps stabilize the knee joint. A meniscal tear refers to damage in the cartilage that is positioned between the tibia and the femur that allows for smooth joint movement.

However, the standard of care for diagnosing knee injuries using MRI scans still comprises a large error rate across the healthcare industry due to its sheer complexity. Images are not standardized: they often have varying spatial resolutions, contain motion artifacts due to patient movements, different care providers may have different policies for their imaging protocols, among other factors. According to a recent study involving experienced abdominal imaging radiologists from Massachusetts General Hospital, who were asked to re-interpret CT scans that were previously interpreted by themselves or their colleagues, disagreements between colleagues occurred more than 30% of the time, and disagreements with themselves occurred 25% of the time [9]. There is therefore a tangible potential for diagnostic improvement using deep-learning.

Deep-learning is a subset of machine learning. In recent years machine learning has been a trendy field of study: the idea is that with technological advancements in processors, we can create algorithms that self-corrects to produce desired outputs by feeding itself with structured data inputs. Deep learning is similar, but has a more advanced, layered architecture: by having layers of algorithms feeding their outputs as inputs into the next algorithm, with each algorithm finding different features of significance from the data it sees, it can use a error-measuring function (called loss function) to figure out which algorithms and which features are most important in creating accurate predictions.

However, deep learning methodologies often require large amounts of training data in order to

avoid a very common problem called “overfitting”. This is the scenario where a model adapts too well to the training data - taking very specific information related to this particular dataset - and uses that as a rule of thumb to predict outcomes. This can fail spectacularly: a rather extreme example would be if a model was trying to distinguish between photos that contain cats and dogs and was trying to distinguish between cats and dogs, but every cat photo happened to be indoors and every dog photo happened to be outdoors, the model might train on the color of the background rather than features relevant to the animals. Ideally one would be using a large dataset that would cover multiple backgrounds to remove this as a factor.

There is a big concern for analyzing medical images using deep-learning architectures: the lack of large public databases. Deep-learning models can often have over a million features, which means overfitting is a concern if the database contains 1000s of images (very common for medical databases due to strict patient confidentiality policies), compared to training over a dataset such as ImageNet which has more than 14 million images [10]. This lack of large datasets problem isn’t specific to the medical domain, but is one that is hard to solve going forward - compared to automated cars which can acquire more data with more test drives, personal medical information remains private under strict confidentiality policies.

Currently, Stanford has released a paper describing a deep learning model called ‘MRNet’ which achieves 0.911 Area Under Curve (AUC) on a test set of 183 images, training over 1000 images and additional augmented images (rotation, shifting, flipping) [1]. If the objective is to have automated pathology detection be applicable to real life situations, a more generalizable and accurate model is needed.

The Problem, and Metrics of Success

The specific problem that I aim to solve in this thesis project is:

“How can we improve the reliability for automated knee pathology detection?”

The critical metrics of success would be as follows:

1. Area Under Curve (AUC)
    a. A metric that shows how well a model is at distinguishing between classes (abnormal knee, normal knee, ACL tear knee, etc). The True Positive Rate (TPR) is plotted against the False Positive Rate (FPR), and a high area under the curve means that the model is better at distinguishing which image is which class.
    b. Goals are:
        i. Baseline: .90 (worst performing model on leaderboard)
        ii. Good: .917 (current MRNet SOTA)
        iii. Stretch: .93 (1.3% is a pretty big improvement given .90 to .917 range)
2. Accuracy

    a. Out of all the validation cases, how many did my model correctly classify?
    b. Goals are:
        i. Baseline: 90%
        ii. Good: 95%
        iii. Stretch: 99%
3. Sensitivity
    a. Also known as the TPR. “How many sick people are correctly identified as being sick?”
    b. Goals are:
        i. Baseline: 90%
        ii. Good: 95%
        iii. Stretch: 99%
4. Specificity
    a. Also known as True Negative Rate (TNR). “How many healthy people are correctly identified as not being sick?”
    b. Goals are:
        i. Baseline: 90%
        ii. Good: 95%
        iii. Stretch: 99%
5. Precision
    a. Also known as Positive Predictive Value (PPV). “How many people identified as being sick are actually sick?”
    b. Goals are:
        i. Baseline: 90%
        ii. Good: 95%
        iii. Stretch: 99%
6. Training and Testing Speeds
    a. How quickly does the model learn, and how quickly can it predict on new data? Could be interesting metric for determining viability for on-the-go diagnosis.
    b. Goals are (On Google Colab K80 GPU):
        i. Baseline: train 12 hours, predict 30 minutes
        ii. Good: train 6 hours, predict 10 minutes
        iii. Stretch: train 2 hours, predict 1 minute
7. Generalizability
    a. How effective is my model at predicting new and unseen data? I will be entering my best model(s) to the Stanford MRNet competition, to be compared to other teams’ models and test my model’s predictive accuracy on new data. This can only be done once to avoid competitors from overfitting on data.
    b. Goals are (metric here is AUC):
        i. Baseline: .90 (worst performing model on leaderboard)
        ii. Good: .917 (current MRNet SOTA)
        iii. Stretch: .93 (1.3% is a pretty big improvement given .90 to .917 range)

Project Summary Statement

Magnetic Resonance Imaging (MRI) is a popular method for diagnosing knee injuries. Due to variations in image acquisition and human error, knee injuries are often misdiagnosed and lead to further health complications and increased health care spending. There is potential for deep learning - a subfield of machine learning suited for such complicated tasks due to their complex feature architectures - to help automate the process of interpreting MRI scans.

However, deep learning models often require vast amounts of data in order to produce algorithms that are generalizable and can accurately predict on new unforeseen data. In the medical world, such access to data is extremely limited due to the nature of personal medical information and the privacy they require. There is a need to improve the reliability of automated knee pathology detection without the requirement of acquiring more data.

In this project, I will try to improve upon two aspects of the deep learning pipeline: by changing the model architecture to be more suited for medical images, and by changing the framework in which the model trains in. To measure the effectiveness of such methods, some metrics include finding the Area Under Curve (AUC), sensitivity, specificity, and precision of the resulting model, as well as how much of the provided training data was needed in order to develop a robust model.

[220 words]

Client, Use, and Impact

The overall problem faced by end users which this project attempts to solve is that interpreting MRI scans is time-intensive and easy to misdiagnose. The ability to distinguish between knee injuries using an automated, fast process would be beneficial to any radiologist, doctor, or healthcare provider that interfaces with patients that experience knee pain. Indirectly, a higher level of care would benefit those patients, and a more efficient healthcare system would help healthcare insurers save money due to reduced need for repeat hospital visits from misdiagnoses.

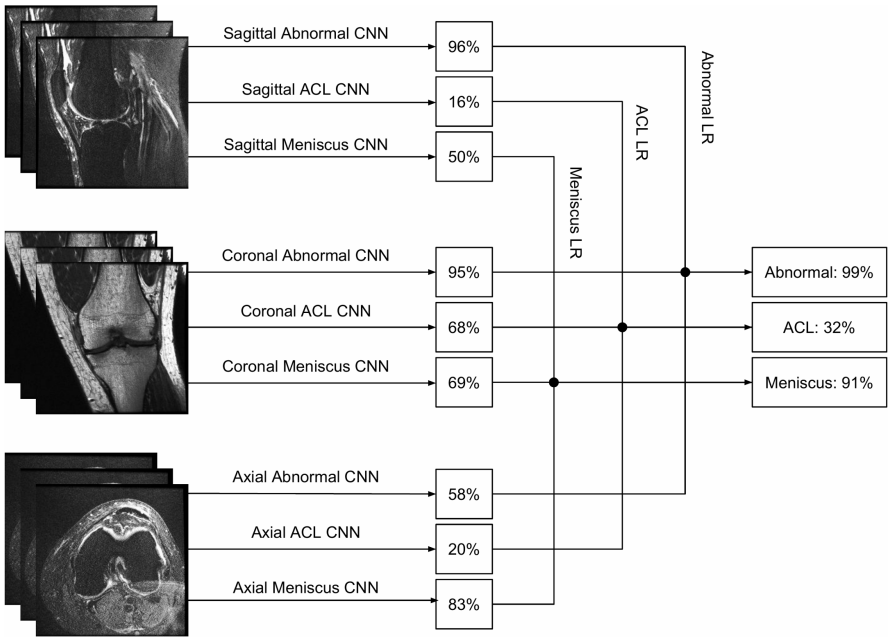
The impact of having such a tool would go beyond the immediate case of a doctor treating a patient. With the ability to distinguish between different injuries, patients can be given a priority for doctors and surgeons to view on a severity-of-injury basis, enabling a more effective workflow in the hospital. Also, with a computer-aided system, practicing radiologists and doctors would be able to devise training schemes in which they can review and learn cases where they have mis-interpreted the scans. Helping medical professionals learn would also serve to improve the healthcare system as a whole.

My client is a company named Covera Health. They seek to create a quantitative scoring system for healthcare providers - hospitals, clinics, organizations - and direct patients towards the best healthcare providers in order to reduce health complications and reduce monetary waste in the system. To do this, one part of their scoring metric is to judge how well doctors - specifically radiologists - diagnose injuries. They compare the problems reported in the institution's provided doctor reports with their own medical experts' diagnosis. In the future, they seek to use machine learning to automate both ends of the process, which would require computer vision and natural language processing.

Previous and Related Work

Previous Work

Perhaps the most relevant previous work done is showcased in the paper by Stanford research group (Bien, Rajpurkar, et al.), since I will be using their knee MRI dataset to train my models on [1]. Their method involved training three separate Convolutional Neural Networks on the same knee MRI dataset, using rotation, horizontal flipping, and shifting to augment their training dataset. Furthermore, to address the problem of having a small dataset, they initialized the weights of their Convolutional Neural Network (CNN) based on a model named AlexNet, which was optimized on the ImageNet database (1.2 million images, 1000 classes, irrelevant to MRI scans), then fine-tuned the weights by training over the MRI dataset (transfer learning) [10].



I will be using their model as a baseline to compare the rest of my work with. After implementing their MRNet algorithm, I will modify either the training dataset the algorithm trains on, or various features in the MRNet algorithm.

Related Work

Similar studies have been performed on brain MRI images, as discussed in “Deep Learning for Brain MRI Segmentations: State of the Art and Future Directions” [2]. As this is more of a survey paper, it discusses an overview of other recent explorations in the field. Various challenges and datasets are showcased, ranging from brain tumor segmentation to lesion detections to segmentation of gray matter, white matter, and cerebrospinal fluid. Also discussed are image preprocessing techniques such as brightness intensity normalization and skull stripping. I believe that Stanford MRNet research group has already done most of the image preprocessing when creating the dataset, but I may draw inspiration from some of these techniques when thinking of data augmentation methods.

In the paper “Semantic Segmentation using Adversarial Networks”, segmentation has been performed using GANs but the application wasn’t used for medical images [5]. Segmentation is the practice of separating an image into distinct objects that we would recognize (for example, a blurry ‘X’ in a static screen, or a car and a street sign from the rest of traffic). The goal of the study was to predict a category label at every pixel in the image using a GAN network instead of the conventional State of the Art CNN.

“AutoAugment: Learning Augmentation Policies from Data” is a very recent paper published in 2019 that discusses the potential for a “supervisor model” to learn the optimal data augmentation strategies from a pool of defined policies for a specific dataset [6]. In previous data augmentation architectures, often times the researcher would define the augmentation strategies for a dataset: for example, in Stanford’s MRNet, the research group defined 3 augmentation strategies (rotating, shifting, flipping) that would be applied on a random basis at certain steps in the training pipeline [1]. With more research into the nature of MRI images, perhaps AutoAugment won’t be necessary, but AutoAugment would be an interesting method to explore, to see how and why computers may think differently from human understanding.

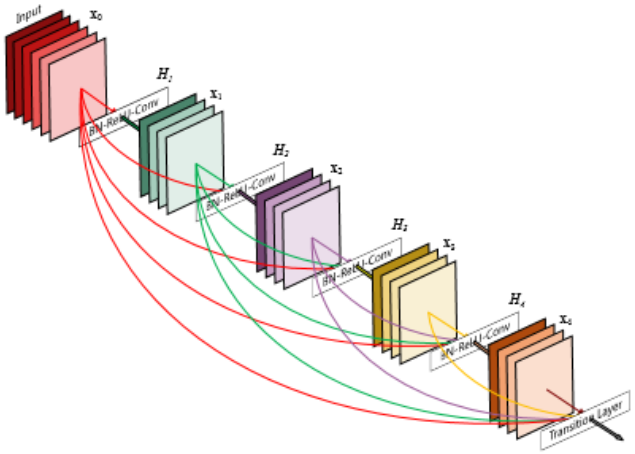
In Goodfellow’s paper on Generative Adversarial Networks (GANs), a new framework for estimating generative models is introduced through game theory [7]. A GAN consists of two models: a Generator and a Discriminator. Like a Cop and Robber game, each tries to perform better than the other - the Generator attempts to generate new “fake” pictures from some latent distributions it learns after training over the “real” data, and tries to convince the Discriminator that these “fake” images are real. The Discriminator looks at both “fake” and “real” data, and tries to distinguish between which are real and which are fake. After several iterations, if the Generator and Discriminator were initiated with good weights, they would ideally approach the condition where:

- The Generator only needs to make small changes to its model after iterations
- The Discriminator can not distinguish the fake images from the real images (50%

accuracy)

Once this equilibrium is reached, the fake images are good representations of the real image population.

“Densely Connected Convolutional Networks”, or otherwise known as “DenseNets”, is a paper that describes a new architecture of CNNs [3]. They modify CNN architecture based on recent observations that shorter connections between layers close to the input and layers close to the output result in deeper, more accurate, and more efficient networks. They propose connecting each layer with each layer afterwards, meaning that if there were L layers, they would have L(L+1)/2 connections in the model. They claim that significant improvements have been observed compared to State of the Art models on four highly competitive object recognition datasets, one of which is the ImageNet used in the AlexNet paper. The graphic below is a representation of DenseNet.



In “Deep Residual Learning for Image Recognition”, layers are reformulated to learn residual functions in reference to layer inputs instead of learning unreferenced functions [4]. They provide evidence that these networks are easier to optimize and perform better with more layers, which deep neural networks tend to have a lot of. With this model, they won first place in ILSVRC and COCO 2015 competitions on ImageNet and COCO datasets in detection, localization, and segmentation.

The only previous work that I have conducted in relation to this project was my summer internship at Covera Health, where my task was to modify a pipeline that extracted metadata from images and organize large amounts of data by categories such as body part or image orientations. There is very little similarities between this previous work and my project; other than the common theme of “data interpretation”, there was no machine learning or modeling involved in my previous work, which is the main focus of this project.

New Approach

I plan to build and test the deep learning model described in Stanford's paper [1] that will serve as a point of comparison for my further modifications. The results of this deep learning model will help measure for the specific objectives described above, such as AUC, sensitivity, etc, by using the model to predict on testing data (that I have created as described earlier). Further modifications to the model will be designed by me. As an overview, I have identified two major routes I can take when it comes to increasing the reliability of the model.

1. I can change the training framework in which the model operates in
2. I can change the model itself to make it more suited towards the specific medical imaging problem

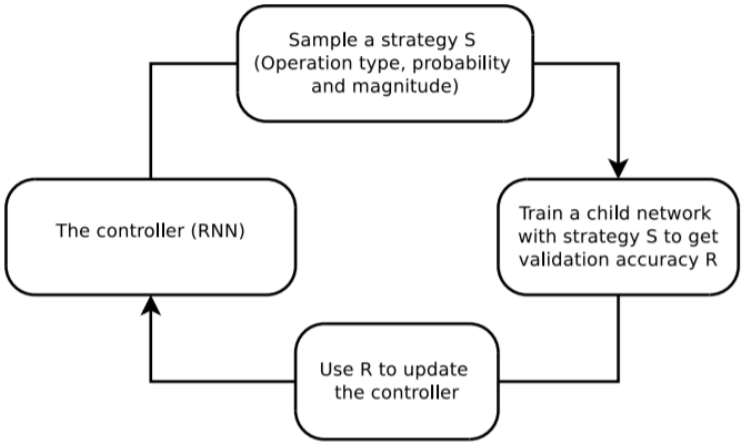
Both parts will be discussed below.

One way to increase a deep learning model's accuracy and robustness is to generate new training data for the model to learn from. "New" in this sense isn't necessarily new information like scanning more patients, but different variations of the same image can be used to help prevent the model from overfitting - for example, thinking that every time a group of pixels near the upper left corner is white, the patient has a problem (whereas it's the pixels' position relative to the rest of the knee that matters).

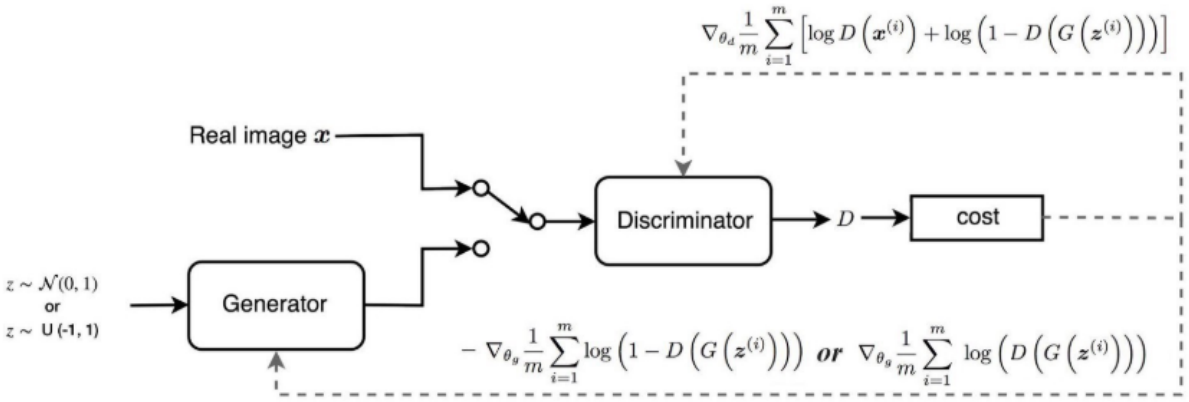
In the Stanford paper, the research group used 3 image modification methods to augment their training data: rotation, shifting, and flipping (horizontally) [1]. In other similar papers, like "Deep Learning for Brain MRI Segmentations", researchers have used rotation, shifting, flipping, and deformations [2]. I want to improve upon this augmentation technique in two ways:

1. Introduce new modification methods and implement AutoAugment from "AutoAugment: Learning Augmentation Policies from Data" [6], which learns to emphasize the modifications that have more significant impacts.
2. Use a Generative Adversarial Network (GAN) to learn the meaningful features of a Knee MRI image and generate fake images that capture these patterns.

The first way is a direct improvement on the traditional method of data augmentation for images. I chose AutoAugment as a potential solution because it's a new method with interesting application to studies where the relative effectiveness of each image modification technique is not immediately obvious. With the AutoAugment method, I could supplement my training dataset with more images that were created from variation methods that had a higher impact on the accuracy of the model, as shown in a visual from the paper.



The second way is to use a GAN model to learn to generate life-like fake knee MRIs by playing a zero-sum optimization game [7]. Like discussed above in previous works, a well-trained GAN can re-create images that capture the generative model's understanding of the dataset, and theoretically these fake generated images can be used to supplement the training dataset here for the automated pathology detection model. Below is a naive diagram of how such a GAN would be trained using a cost function:



Source: https://medium.com/@jonathan_hui/gan-why-it-is-so-hard-to-train-generative-advisory-networks-819a86b3750b

I have not seen studies that use GANs to generate training data for another model to optimize on, so this is a new exploration in the field to my knowledge. Both of these methods would address issues with overfitting that plague many deep learning models. When optimizing over the training and validation datasets, the goal is to improve metrics such as AUC, accuracy, etc. on the test set using augmentation.

Another potential route is to investigate and change the model. Fine-tuning the model's

architecture to be optimal for characteristics of the dataset population may give better predictive results without loss of generality for other similar datasets.

In Stanford's MRNet design, they chose to implement a CNN architecture using weights optimized with AlexNet: a Deep CNN trained to classify images in ImageNet [10]. This style of learning is called transfer learning; by using a model that's already trained to distinguish between various classes of images, hopefully they won't need to learn as much and can ignore slight variations when training on the much smaller knee MRI images.

One potential approach I want to consider is to replace AlexNet with a different CNN model. Other than training over other images and creating a naive understanding of how images work, AlexNet theoretically does not add much insight into the medical imaging realm. DenseNets, on the other hand, claim to have several advantages, one of which is a substantial reduction in the number of parameters [3]: given that knee MRIs all follow a general body structure with only minor differences in various forms of injuries, it makes sense to me that reducing parameters that influence the prediction would allow the model to have higher accuracies.

The biggest consideration that would be used to differentiate between which solution I prefer would be the improvements seen in the metrics of success as defined above. Apart from this, an important difference between the two approaches is that the training framework method can be applied to many other machine learning problems, whereas the model modification method is more tailored towards medical imaging (or maybe just knee MRIs). If the two solutions provided similar benefits in terms of AUC, accuracy, etc., I would likely choose to emphasize the former due to its broader application.

I anticipate coding in Jupyter notebooks using the python coding language, and training my models on Google Colab, a limited free service where users have access to a Tesla K80 GPU. Various python libraries will be needed: either PyTorch, Tensorflow, or SciKit Learn for machine learning specific tasks, matplotlib for data visualizations, and Numpy, Pandas, and Json for data management.

The dataset contains 2 partitions of data: the training dataset, and a validation dataset. A third dataset, the test dataset, is kept on Stanford's github and unavailable to me (to prevent competitors from overfitting on the data), so I plan to partition a portion of my training dataset (~100 cases) and create my own testing set that I won't use until calculating the metrics for success as described earlier.

In order to calculate the AUC metric, I can first plot an ROC curve by calculating the true positive and false positive rates along a threshold axis. To acquire the other metrics (accuracy, sensitivity, specificity, precision), I will take the threshold value that yields the highest accuracy and calculate the other 3 using their respective formulas. Training and testing speeds can be recorded using a python time-marking library (os.time), and generalizability will be measured by

how well my model performs on completely new data (Stanford's testing dataset).

Related courses, useful skills and background knowledge

I am pursuing an S.B. in Electrical Engineering, and this project is a good application of the signals and systems portion of EE. Many of my classes in S.B. EE such as - ES150: Probability and Statistics for Engineers, ES156: Signals and Systems, ES157: Biological Signal Processing, ES96: Engineering Problem Solving & Design Project, and CS181: Machine Learning - have taught me skills that this project will continue to build upon. This project requires understanding of feedback systems, machine learning, statistics, data processing, signal distributions and manipulations within images.

I may need to acquire new skills in programming, specifically with the Google Colab Interface and with python libraries like Pytorch and Tensorflow. I may need to ask Harvard for permission to use their Odyssey modules and acquire training in order to use it. Being able to ask local experts in MRI imaging techniques, like the ones situated in Massachusetts General Hospital, would also be beneficial to this project.

My motivation for choosing this project is that I am very interested in the applicability of Machine Learning to solve complex, humanitarian problems. The world has evolved so quickly within the last few decades that we're now facing a problem of trying to analyze and process all the information we have generated and collected. This project will be a good stepping point in my career, to prove that I can learn about and apply algorithms to new fields.

Preliminary Project Milestones

Milestone 1: Defining

1. Literature Review (9/9 - 9/23)
    i. More State of the Art methodologies for detection algorithms
    ii. Literature on MR Knee images and their protocols / processes
    iii. Rework project proposal and direction if necessary
2. Coding Review + Permissions (9/9 - 9/23)
    i. Understanding PyTorch/Tensorflow/other ML libraries, learn how to use
    ii. Learn how to integrate data and algorithms with Google Colab
    iii. Ask Harvard for access to Odyssey GPU

Milestone 2: Designing

3. Exploring the Data (9/23 - 10/08)
    i. Plot distributions of relevant image information, such as:
        1. Class distributions
        2. Image data like contrast distributions
        3. Histogram for each case: see differences between cases, maybe

we can cluster into different scanners were used …? Acquisition could be different. See if I can separate the cases into different scanners or some other categories.

4. Implementing Baseline Models (10/08 - 10/22)
    i. Implement Stanford MRNet setup
    ii. Analyze cases where model fails
    iii. Measure metrics used for specification (sensitivity, AUC, etc.)
5. Prioritize which solutions to explore (10/22 - 10/29)
    i. Based on the failure analysis from MRNet model, which models are more likely to make improvements?

Milestone 3: Build, Measuring + Verifying

6. Build models / training modifications (10/29 - 12/10)
    a. For each model and training modification:
        i. Implement (10 days)
        ii. Measure metrics of success (2 days)
        iii. Propose hypothesis on reason for results, make edits and reiterate (2 days)
7. Prepare for Oral Design Review and Presentation (12/10 - 12/17)

Milestone 4: Analysis + Re-iteration

8. Analyze Model Results (12/20 - 1/08)
    i. Brainstorm why some models worked better than others. What can I take away from this information? Come up with a way to take the next step and combine the best parts.
9. More Model Iterations (1/08 - 2/29)
    a. For each model and training modification:
        i. Implement (10 days)
        ii. Measure metrics of success (2 days)
        iii. Propose hypothesis on reason for results, make edits and reiterate (2 days)
    b. Compare resulting models to those generated in earlier part of project
    c. Prepare model for submission to Stanford's Competition

Milestone 5: Preparing for Presentations

10. Finalize Project (2/29 - 3/05)
    i. Finish any missing documentation for Final Progress Report
    ii. Generate nice, detailed graphics
11. Rehearse Presentation (3/05 - 3/25)
    a. Create detailed summary of the project
        i. What I hoped to accomplish
        ii. What I accomplished
        iii. Future Steps
    b. Practice, Rehearse, and fix presentation

    c. Ask advisors for hard questions that I might be expected to answer
12. Post Project Logistics (3/25 +)
    i. Incorporate feedback for report and turn in Final Report
    ii. Rehearse and present poster presentation for Spring Design Fair

Budget

Due to the software nature of this project, I do not plan to use any significant money for the budget. Access to the Knee MRI dataset is free, and access to a limited version of Google Colab (GPU) is also free. Depending on how complex my model becomes, I may want to request access to Harvard's Odyssey cluster. Also, as I iterate through my models I may want to use more online storage to secure my work and ensure it doesn't get lost (but I believe Harvard emails provide infinite storage).

Resource	Cost
Access to Stanford MRNet Dataset	Free
Google Colab (limited)	Free
Access to Harvard's Odyssey	Free?
Google Drive Storage	Free with Harvard Gmail

References

[1] Bien, N., Rajpurkar, P. et al. “Deep-learning-assisted diagnosis for knee magnetic resonance imaging: Development and retrospective validation of MRNet”. PLOS Medicine Journal (2018). https://doi.org/10.1371/journal.pmed.1002699.

[2] Akkus, Z., Galimzianova, A., Hoogi, A. et al. “Deep Learning for Brain MRI Segmentations: State of the Art and Future Directions”. D J Digit Imaging (2017) 30: 449. https://doi.org/10.1007/s10278-017-9983-4

[3] Huang, G., Liu, Z., Maaten, L., Weinberger, K. “Densely Connected Convolutional Networks”. arXiv (2018). arXiv:1608.06993v5

[4] He, K., Zhang, X., Ren, S., Sun, J. “Deep Residual Learning for Image Recognition”. arXiv (2015). arXiv:1512.03385v1

[5] Luc, P., Couprie, C., Chintala, S., Verbeek, J. “Semantic Segmentation using Adversarial Networks”. arXiv (2016). arXiv:1611.08408v1

[6] Cubuk, E., Zoph, B., Mane, D., Vasudevan, V., Le, Q. “AutoAugment: Learning Augmentation Policies from Data”. arXiv (2019). arXiv:1805.09501v3

[7] Goodfellow, I., Pouget-Abadie, J., et al. “Generative Adversarial Networks”. arXiv(2014). arXiv:1406.2661v1

[8] Fletcher, Jenna. “Ten common knee injuries and treatment”. Web Article. Medical News Today (2017). https://www.medicalnewstoday.com/articles/319324.php.

[9] Abujudeh, H., Boland, G., Kaewlai, R., Rabiner, P., Halpern, E., Gazelle, G., Thrall, J. “Abdominal and pelvic computed tomography (CT) interpretation: discrepancy rates among experienced radiologists”. Eur Radiol. 2010;20(8):1952–1957. doi: 10.1007/s00330-010-1763-1

[10] Krizhevsky, Alex, Sutskever, I., Hinton, Geoffrey. “ImageNet Classification with Deep Convolutional Neural Networks”.

→ THE NEXT PAGE MUST BE INCLUDED IN YOUR PROPOSAL WITH THE PROPER SIGNATURES