Adversarial generative models applied to diagnosing Osteoarthritis

Evaluating different techniques for fine-tuning discriminator models to classify osteoarthritis

Teun den Boer[1]

Supervisors: Jesse Krijthe[1]**, Gijs van Tulder**[1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 22, 2025

Name of the student: Teun den Boer
Final project course: CSE3000 Research Project
Thesis committee: Jesse Krijthe, Gijs van Tulder, Michael Weinmann

Abstract

Osteoarthritis is a chronic joint disease in which the protective cartilage between bones deteriorates over time, leading to pain, stiffness, and reduced mobility. Diagnosis is a time-consuming and somewhat subjective process. To address this challenge, machine learning techniques can be applied. However, training supervised models on medical images is often challenging because of the limited availability of labeled training data. Self-supervised methods, which pretrain models to learn useful features without labels, offer a potential solution to this issue. In this paper, we explore the use of Generative Adversarial Networks (GANs) as a pretraining step for osteoarthritis diagnosis. The first step is the training of a GAN on a semi-public dataset of x-ray images. In the second stage, we explore different strategies for fine-tuning the discriminator model to diagnose osteoarthritis. Our experiments suggest that while GAN-based pretraining offers slight improvements over purely supervised approaches, the performance gains remain modest. [1]

1 Introduction

Osteoarthritis is a progressive joint disease that impacts many individuals, leading to symptoms such as pain, stiffness, and discomfort. The diagnosis of osteoarthritis is clinically defined mainly based on the patient's symptoms, but imaging methods are widely used [9]. Healthcare professionals rely on X-ray images to identify damage to joints. However, the process of analyzing the scans can be both time-intensive and subjective, particularly in the early stages of the condition when signs of damage are subtle and harder to detect. If an automated system could produce reliable predictions and diagnoses, it could enable patients to get quicker access to appropriate treatment and free up valuable time of highly trained clinicians to focus on more critical cases. If the automated system proved robust, it could even help minimize the late-stage effects by giving early warnings.

One promising approach to building such automated systems is through the use of machine learning, specifically neural networks. Convolutional neural networks have been shown to be able to achieve remarkably low error rates when trained on massive datasets of labeled data (such as ImageNet [10]). These training methods however require the data to be labeled, which is an expensive and time consuming task, especially in the medical field.

To reduce the need for massive labeled datasets, self-supervised learning frameworks have been developed. These methods train a model on unlabeled data using a pretext task for which labels are not needed. The goal of this pretext task is to help the model learn useful representations from the data without relying on labels. Afterwards, the model can be fine-tuned on the actual problem using a much smaller labeled training set. [8] [13]

There are many possible pretext tasks. The model could for example be trained as an autoencoder, or it could be tasked to find representations that are invariant to predetermined transformations, it could even predict obscured parts of an image. In this paper we will explore the use of pre-training the classifier as part of a Generative Adversarial Network (GANs). In GANs two networks a generator and a discriminator are trained in competition with each other. Typically, the trained generator is the goal of this process. Its latent space can encode high-level features and map them to images [18]. However, in this work we focus on an alternative use case. Using the discriminator as the basis for the downstream classifier. Specifically, we experimented with two strategies: (1) directly fine-tuning the discriminator and (2) replacing the final classification layer with a new classification head of varying types.

This is an interesting pre-training task because the model is implicitly learning visual features in order to tell real images apart from fakes. In doing so, it learns to understand the underlying structure and distribution of the input images, or at least the parts the generator is getting wrong. This could be especially useful in the context of medical images because the pretraining is teaching the model to look for properties that are out of the ordinary, which could be correlated with a disease, though this would have to be confirmed.

Working with GANs also comes with unique challenges in the form of training instability. The training has a couple failure modes that other machine learning techniques do not. For example the discriminator can become too strong or too weak relative to the generator leading to a vanishing gradient. Additionally, models can experience mode collapse, where the generative models start to produce only a small subset of the training data. The models may also oscillate between strategies, never stabilizing into an equilibrium. [1] [6] [18]

Despite these challenges, the potential of this approach makes it a compelling direction to explore. In this paper, we investigate whether using a GAN discriminator as the foundation for a classifier can improve osteoarthritis classification performance compared to standard supervised learning. We also examine whether replacing the classification head with additional layers can further enhance the model's performance. Our hypothesis is that pre-training a model using the GAN-based method allows it to learn useful features from unlabeled data. This pre-trained model should provide a stronger starting point than random weight initialization for supervised learning, improving its ability to diagnose osteoarthritis.

2 Related Work

Fine-tuning a discriminator is not a novel idea. This approach was first introduced by [18] in their work on deep convolutional generative adversarial networks (DCGANs). In their experiments, they demonstrated that a discriminator, trained to distinguish between real and generated images, could learn high-level, abstract features useful for downstream tasks.

Since then, this approach has been successfully adapted to

[1] Code is available at https://gitlab.tudelft.nl/osteoarthritis-2025-bsc/gan_classification_model_teun

various domains, including remote sensing. For example [11] and [20] employed this method to develop classification systems for satellite imagery. These approaches were effective in leveraging the vast amounts of unlabeled image data available.

Further theoretical insights were provided by [14], in their analysis into why fine-tuning discriminators can yield strong feature extractors. Their study suggests that discriminators can capture robust and semantically meaningful representations. However, they also identified potential limitations, such as little feature separation and sensitivity to mode collapse.

3 Methodology

3.1 General Approach

To investigate whether pretraining a discriminator within a GAN can improve downstream classification performance on osteoarthritis, we followed a two-step approach. First, the discriminator is trained as part of a GAN. Afterwards the discriminator model was repurposed to become the classifier. The classifier was then trained directly on the final task diagnosing osteoarthritis. The pre-trained classifier was compared with the same architecture but initialized with random weights, this served as the control. The GAN is trained on a set of unlabeled data, while the fine-tuning is on a smaller labeled set. The labeled data is split into a training, testing and validation set. The labeled training set is also used in the GAN training without the labels, as having more images helps with the pretraining. Testing and validation are not.

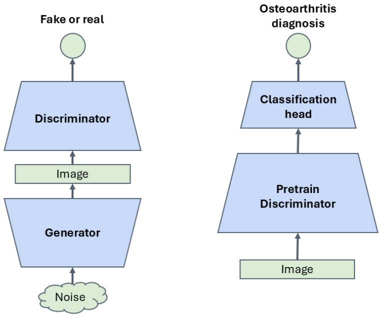


Figure 1: General approach, step one is GAN training. Step two is fine-tuning

3.2 GAN Framework

To pre-train the model the GAN framework was used. This was first introduced in [5]. In this paper the authors propose a way to train generative models via an adversarial process. Two models are optimized simultaneously, a generative model that produces images, and a discriminative model that classifies the images as coming from the dataset or produced by the generator. This framework is a two-player minimax game, in which the two models are in direct competition with each other. The result of this training process is that the generator is incentivized to produce images from the same distribution as the training set (even though it has never seen the training data).

This approach offers various benefits, namely the discriminator is learning visual features in order to tell real images from the generated images. To properly distinguish images from a sufficiently advanced generator the discriminator needs to have a model of what the structure of the joints should be. This suggests that the model will also need to be looking at features like the distance between the bones and texture of the joint. The theory to test is that this will also help the model learn the different types of images there are i.e. healthy joint vs unhealthy.

GANs are well known for their training instability. Several methods have been proposed to solve this issue such as Wasserstein GAN [1], Wasserstein GAN with gradient penalty [6], R1 Regularization [15], spectral normalization method [16]. Along with several guidelines for architectures [18].

In our testing the Wasserstein GAN, R1 Regularization and spectral normalization method did not work, for the dataset used in this paper. This may be the result of suboptimal hyperparameter tuning or bugs. However one possible explanation is that it is a result of the dataset itself. The dataset images are all of the same joint taken in a standardized way. This makes them more similar to each other than natural images. The discriminator can therefore figure out fake from real with a high confidence very early on in the training process, which could lead to the experienced mode collapse. What did work was the Wasserstein GANs with gradient penalty, which is to be expected as this method is relatively good at preventing mode collapse.

To design the models several guidelines from the literature are used. The first is the use of LeakyRelu activation functions in the discriminator. This type of activation function is especially good at allowing the gradient to flow to earlier layers which in this case includes the generator. This helps the generator to always have a clear direction to improve. Secondly is the use of strided convolution blocks instead of pooling layers, this also helps with stability [18]. Lastly no dropout and batch normalization was used in the discriminator as this doesn't work well with the optimization method [6].

3.3 Fine-tuning

After pre-training, the discriminator model is repurposed under the assumption that it has learned task-relevant features. We test two fine-tuning strategies. The first involves minimal modification: the pre-trained discriminator is fine-tuned as is. The second strategy replaces the classification head with additional layers, and the entire model is fine-tuned. In both cases, the fine-tuned models are compared to controls with the same architecture and training procedure, the only difference being the weight initialization.

No layers are frozen during training, as this would not provide a realistic comparison with the control models. Since the models are sufficiently large to overfit the data, early stopping is used to prevent overfitting. The models are trained until the validation loss stops decreasing, after which performance is evaluated on the test set.

3.4 Evaluation

Measuring performance improvements in GAN training is more challenging than in supervised learning, as the generator and discriminator losses do not steadily decrease over time. To assess the performance of the GANs during pretraining a combination of metrics are used.

To evaluate the generator, the Fréchet Inception Distance (FID) is employed [7]. This metric estimates the distance between the distribution of real images and the distribution of generated images. In this research, the FID implementation from the TorchMetrics library was used. It relies on Inception v3, a model trained on natural RGB images of a different resolution. While this may not be the best metric for x-ray images, FID remains a standardized benchmark, which makes it useful for comparing models. The FID is also calculated against the test set. A significant difference in FID between the training and test sets could indicate overfitting.

To further test for overfitting, the discriminator's confidence on the training set is compared to its confidence on the test set. A significantly higher confidence on the training data could suggest that the discriminator has memorized the training samples rather than learning generalizable features.

The effectiveness of the pretraining strategy is evaluated using area under the ROC curve (AUC) on the testing set. The AUC is more important than metrics like accuracy in this case as there is a class imbalance in the data. The ROC curve is the plot of the true positive rate over the false positive rate at each possible decision threshold [4]. The area under the ROC curve represents the probability that the model, if given a randomly chosen positive and negative example, will rank the positive higher than the negative. The AUC is therefore independent of the specific threshold used for prediction and gives a more clear idea of how much the model has learned.

4 Experiment

4.1 Data used

To train the models, the CHECK [19] and OAI [17] datasets were used. These datasets consist of multiple scans of 969 and 4755 individuals, respectively. Each individual has no more than 5 scans in the dataset. The datasets have slightly different distributions and properties. The OAI contains significantly more healthy patients. The images are also on average less sharp and have different brightness histograms [2]. To test self-supervised methods, it is useful to have a larger unlabeled dataset than the fine-tuning dataset, as this is a more realistic scenario. To also avoid the models learning to distinguish between the datasets, thereby unfairly achieving higher accuracy, the OAI dataset was used exclusively as an unlabeled dataset to assist during pre-training.

The CHECK dataset was split 80-10-10 into a train, test and validation sets. The split was done on individuals and not on scans in order to make sure the testing set is independent. For the purposes of training the images were converted to a standard resolution of 224x224 and normalized to [-1, 1]. The right hip joint is also flipped so that the GAN doesn't need to learn the mirror image of all the features. Osteoarthritis is typically classified into five levels, which are grouped into two categories for training purposes: the first two levels indicate no or doubtful presence of the condition, while the last three indicate definite presence.

4.2 Architecture

The generator is designed to map a noise vector to an image. It starts off by using a fully connected layer that projects the latent vector into a 7×7×512 feature map. This is followed by five upsampling blocks. Each block consists of upsampling, a 3×3 convolution, batch normalization, and a ReLU activation function. The final output is scaled using the tanh activation function. For specific details such as the amount of channels in each layer reference the codebase.

The discriminator model maps images to one value that represents the model's confidence that the image is from the real dataset. It begins with 5 down sampling blocks, each consisting of 4x4 convolution with stride 2 and an activation function in this case LeakyRelu ($\alpha = 0.2$). The convolutional blocks are followed with one fully connected layer to the output. No activation function is used on the output to allow the WGAN-gp training method to properly optimize the networks.

4.3 Training WGAN-GP

To optimize the networks the Wasserstein GAN GP method was used [3]. The models were trained for 200 epochs, with a batch size of 64 images. The Adam optimizer was used with a learning rate of 0.0001, $\beta_1 = 0$ and $\beta_2 = 0.99$. The latent space (random input) was 100 dimensional. The discriminator was trained 5 times as often as the generator per epoch. The penalty constant $\lambda_{gp} = 10$. As the goal of this research is not to find new training methods for GANs an existing implication of the Wasserstein loss with gradient penalty was used, credits to [12].

4.4 Base model

The first test conducted was to test if the discriminator model has indeed learned useful features. The pretrained and control models were trained again using the Adam optimizer on the labeled dataset, until the performance on the validation set decreased. The pretrained model performed best with 7-8 epochs. After which performance on the testing set decreased as the model began to overfit. This procedure was repeated for the discriminator models saved at every 10-epoch intervals during pretraining.

4.5 Classification head

The second experiment was to examine whether replacing the classification head with additional layers can further enhance the model's performance. First we define the classes of possible architectures to search through. Each method will leave the base of the model unchanged and replace the classification head. The features in the base are again not frozen and can also be optimized. In Table 1 we define the possible classification heads that could be used. From this set 50 models were sampled and trained on the dataset with labels. Each model was trained until validation loss began to increase. The models are again compared with the same architecture but random weights.

Table 1: Set of architectures to append to the base model

Nonlinearity	[ReLU, LeakyReLU, Tanh, Sigmoid]
Depth	[1, 2, 3, 4]
Batch norm	[True, False]
Dropout	[0, 0.2, 0.5]
Layer size	[16, 32, 64, 128]

5 Results

5.1 Generator performance

The generator produces reasonable results but suffers from a bit of blurriness and random white blobs, though the blobs are also present in the training set. 25 samples can be seen in Figure 2. The FID score can be seen in Figure 4. In the Figure it can be seen that the model is somewhat steadily decreasing, and has stabilized around 200 epochs. Secondly it can also be seen that the test and training curves diverge a little, implying that the generator is slightly overfitting to the data.

5.2 Discriminator performance

To test how much the discriminator is overfitting, average confidence of the final model on the training and testing set is compared. On the training set it is 47.855 and on the testing set it is 47.030. The train-test gap is 0.824. For the purpose of the next tests this was deemed good enough. One somewhat interesting test that can still be applied is to find the input image which maximizes the discriminator's realness score. Starting with a random input image, gradient descent was used to optimize the image to get the highest possible confidence from the model. The resulting image that does that can be seen in Figure 5. The image does not look anything like a hip joint, this suggests that the generator is sufficiently good at producing the contours of bones that the discriminator cannot use it to classify it as fake. This may have implications for turning the discriminator into a classifier.

5.3 Fine tuning

The results from the experiment show that the pretrained (base) model gets AUC of 0.72 vs the control which got an AUC of 0.68, The roc curve for this model can be found in Figure 6. The best addition to the model are the layers described in Table 2. The AUC of this model is 0.734. The AUC of the control model was also computed, this was 0.727. (remember this was the same architecture only with a different starting point). This implies that the discriminator model did learn some useful features for classification. Discriminator models saved at 10-epoch intervals were also compared, and found to perform similarly or worse.

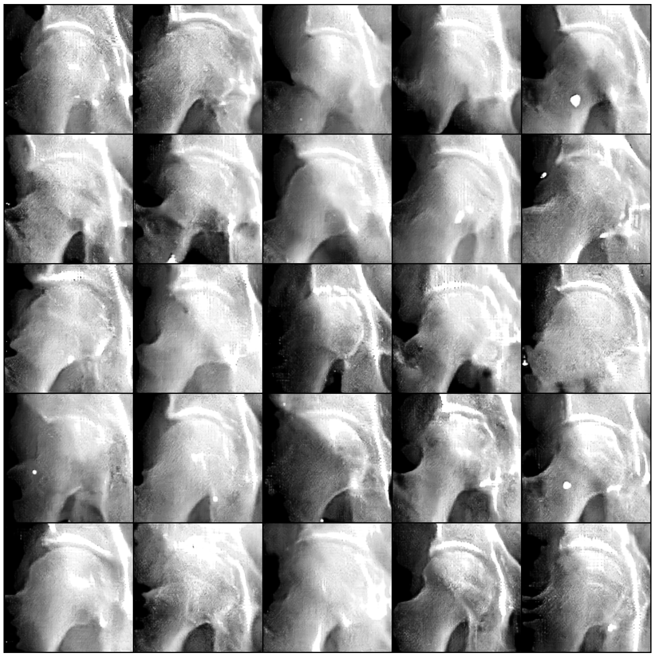


Figure 2: 25 random images produced by the generator. The images are reasonable fakes but suffer from a bit blurriness

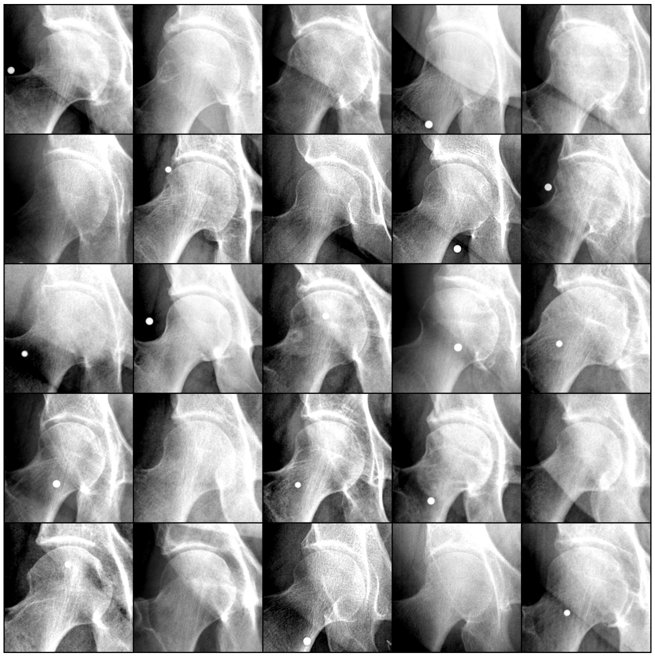


Figure 3: Reference images from the dataset

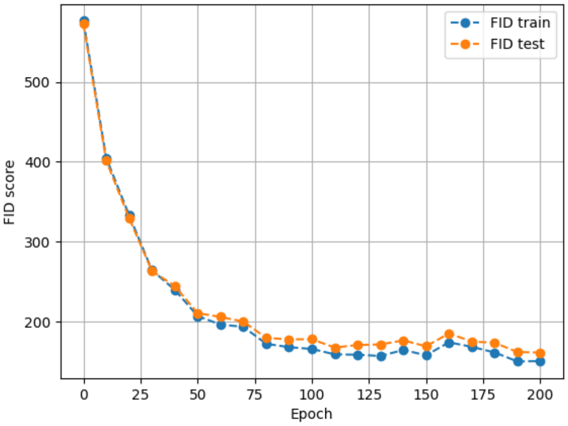


Figure 4: FID score plotted over the training epochs. The FID score was calculated over 2024 fake images and 2024 real images per 10 epochs. In blue the generated images are compared with the train set and in orange with the test set. The curves diverge a little, suggesting that the generator is slightly overfitting to the data.

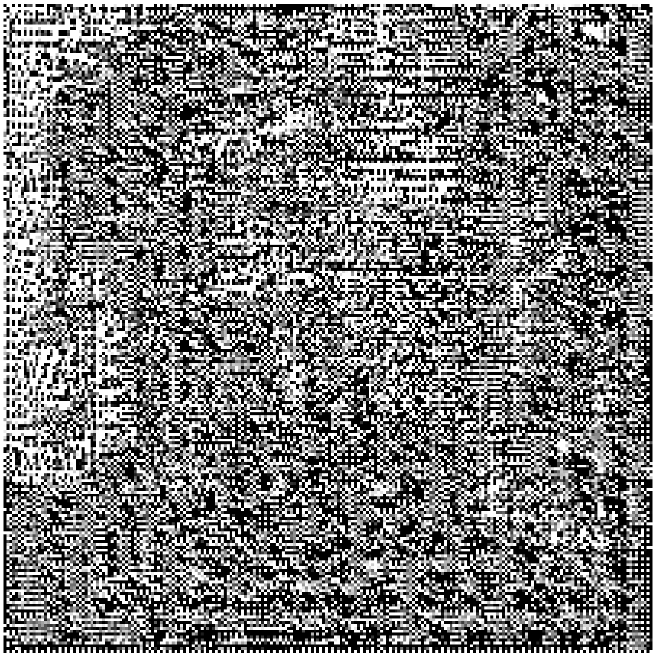


Figure 5: This image that maximises the confidence of the discriminator that it is a real sample

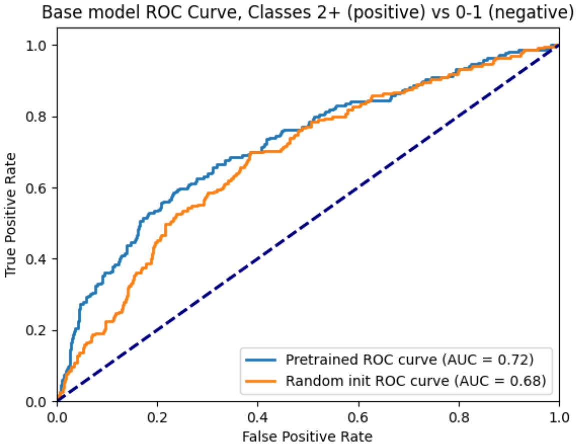


Figure 6: The ROC curve of the base discriminator when finetuned vs the control

Layers	**Details**
`Linear`	25088 → 64
`BatchNorm1d`	64
`Tanh`	-
`Linear`	64 → 128
`BatchNorm1d`	128
`ReLu`	-
`Dropout`	p = 0.2
`Linear`	128 → 16
`Dropout`	p = 0.5
`Linear`	16 → 64
`ReLu`	-
`Dropout`	p = 0.5
`Linear`	64 → 1

Table 2: Best fine-tuning addition

6 Responsible Research

Several considerations are necessary to ensure this research is conducted responsibly. First, the models are trained on biomedical data derived from real individuals, which raises important ethical and privacy concerns. As the data of these individuals cannot be released by both the rules associated with the datasets and by ethical guidelines, the results of this research are not reproducible by other institutions without releasing the original data set. Should this need arise, it would be advisable to contact the TU Delft directly. Secondly the generator could have learned some of the specifics of individuals. It appears that the model is not overfitting, but nevertheless this model will not be shared, without a lot more tests to prove it doesn't violate any privacy concerns.

One limitation for reproducibility comes from the fact this study does not use a large sample set of different architectures and hyper-parameters. This is because training was prohibitively time consuming, which made it impractical to set up a proper T-test. While the code base used a defined random seed to make the data reproducible, any small change to the method will mean different outcomes.

7 Discussion

The results of our study show that while GAN-based pretraining provides some benefit to downstream osteoarthritis classification tasks, the gains are modest. The AUC improvements (0.72 from 0.68 for the base discriminator and 0.734 vs 0.727 for the best architecture) indicate that the discriminator does learn features useful for classification, but these are not major improvements. There are several possible reasons for this.

One major factor could be the structure of the dataset. The X-ray images used are highly standardized, which may limit the features the GAN learns to distinguish real from fake. The discriminator may be relying on texture cues to stay ahead of the generator. We see some hints of this in Figure 5. What is clear is that the features in the model are not well lined with diagnosis. A quick test to see Pearson correlation between the neurons activation and the labels shows that in the last 3 layers of the network no neurons are more than 0.2 correlated with having osteoarthritis. This means that the optimizer still has to do a lot of work.

Of course the GAN training only represents one sample, the network architecture and hyper-parameters are not tested over a range of values. This is a problem for making more generalized claims about how well the method works or doesn't. All we can say is this combination performed the way that it did. The reason not multiple GANs were trained was the massive amount of compute needed. Training this GAN is approximately a 12 hours task on the Delftblue supercomputer.

This study also does not explore the effects of additional data on performance. In the experiments the OAI database was used as unlabeled data to help train the GAN. It was not tested what the effect was of differences between the OAI and CHECK. It could be that the difference between the datasets negatively impacted performance. Also it was not tested how adding more data to the training helps with improve performance or not.

8 Conclusion

In this study, we demonstrated that GAN-based pretraining can improve model performance in diagnosing osteoarthritis. Our method achieved an increase in AUC compared to the baseline models without GAN pretraining, suggesting that discriminators can learn useful features even in medical imaging.

However, while the improvements are measurable, they remain modest. The performance gains may not justify the considerable computational cost and time needed to work with the instability typically associated with training GANs. These factors limit the practical value of the approach in its current form, especially when more accessible and less resource intensive alternatives, like data augmentation and regularization techniques may offer comparable or greater improvements.

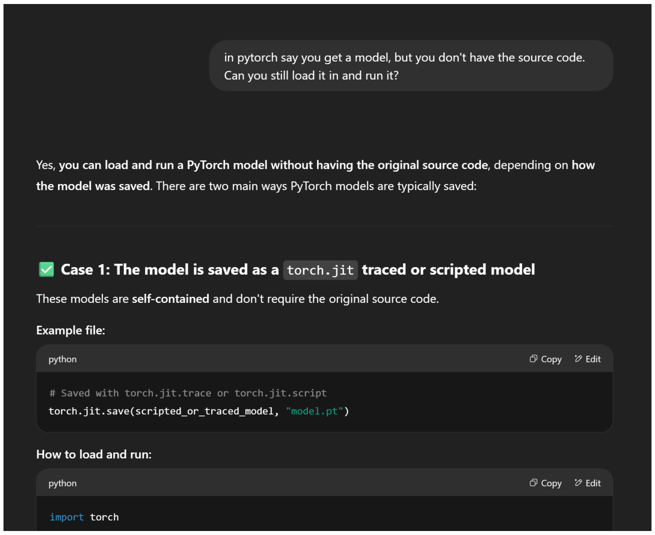
A Acknowledgements

I would like to thank my supervisor, Dr. Grijs van Tulder, for his guidance and advice during the course of this project. I also wish to thank Prof. Jesse Krijthe for his time and constructive feedback.

Computations were performed using the DelftBlue supercomputer [3], without which this study would have been much more time consuming.

B Use of AI

Chat-GPT was used during the course of the project, mainly with helping with coding and as a replacement to stack overflow. Chat-gpt was also asked how to make code better and cleaner. In addition, it was asked to help fix grammar and improve sentence structure, though the sentences produced were often very general and overly formal. Grammarly was also used to help improve writing.



in pytorch say you get a model, but you don't have the source code. Can you still load it in and run it?

Yes, **you can load and run a PyTorch model without having the original source code**, depending on **how the model was saved**. There are two main ways PyTorch models are typically saved:

✅ **Case 1: The model is saved as a `torch.jit` traced or scripted model**

These models are **self-contained** and don't require the original source code.

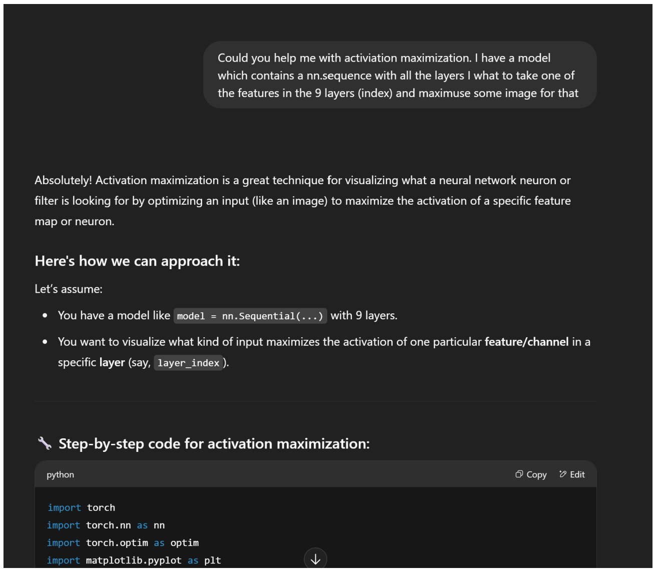
Example file:

```python
# Saved with torch.jit.trace or torch.jit.script
torch.jit.save(scripted_or_traced_model, "model.pt")
```

How to load and run:

```python
import torch
```

Figure 7: Prompt to GPT and response



Could you help me with activiation maximization. I have a model which contains a nn.sequence with all the layers I what to take one of the features in the 9 layers (index) and maximuse some image for that

Absolutely! Activation maximization is a great technique for visualizing what a neural network neuron or filter is looking for by optimizing an input (like an image) to maximize the activation of a specific feature map or neuron.

Here's how we can approach it:

Let's assume:

- You have a model like `model = nn.Sequential(...)` with 9 layers.
- You want to visualize what kind of input maximizes the activation of one particular **feature/channel** in a specific **layer** (say, `layer_index`).

🔧 Step-by-step code for activation maximization:

```python
import torch
import torch.nn as nn
import torch.optim as optim
import matplotlib.pyplot as plt
```

Figure 8: Prompt to GPT and response

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.

[2] Roland Bockholt. Improving generalizability in x-ray segmentation of the femur, 2024. Bachelor's thesis.

[3] Delft High Performance Computing Centre (DHPC). DelftBlue Supercomputer (Phase 2). https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase2, 2025.

[4] Tom Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. ROC Analysis in Pattern Recognition.

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[6] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans, 2017.

[7] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2018.

[8] Shih-Cheng Huang, Anuj Pareek, Malte Jensen, Matthew P. Lungren, Serena Yeung, and Akshay S. Chaudhari. Self-supervised learning for medical image classification: a systematic review and implementation guidelines. *npj Digital Medicine*, 6(1):74, 2023.

[9] David J Hunter and Sita Bierma-Zeinstra. Osteoarthritis. *The Lancet*, 393(10182):1745–1759, 2019.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, 01 2012.

[11] Daoyu Lin, Kun Fu, Yang Wang, Guangluan Xu, and Xian Sun. Marta gans: Unsupervised representation learning for remote sensing image classification. *IEEE Geoscience and Remote Sensing Letters*, 14(11):2092–2096, 2017.

[12] Erik Linder-Norén. Pytorch-gan: A collection of pytorch implementations of gans. https://github.com/eriklindernoren/PyTorch-GAN, 2019. Accessed: 2025-06-01.

[13] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, page 1–1, 2021.

[14] Xin Mao, Zhaoyu Su, Pin Siang Tan, Jun Kang Chow, and Yu-Hsing Wang. Is discriminator a good feature extractor?, 2020.

[15] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge?, 2018.

[16] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks, 2018.

[17] National Institutes of Health. National institutes of health - osteoarthritis initiative (oai) dataset. https://nda.nih.gov/oai.

[18] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.

[19] J Wesseling, M Boers, M. A. Viergever, W. K. Hilberdink, F. P. Lafeber, J Dekker, and J. W. Bijlsma. Cohort profile: Cohort hip and cohort knee (check) study. *International Journal of Epidemiology*, 45(1):36–44, 2016. Epub 2014 Aug 29.

[20] Mingyang Zhang, Maoguo Gong, Yishun Mao, Jun Li, and Yue Wu. Unsupervised feature extraction in hyperspectral images based on wasserstein generative adversarial network. *IEEE Transactions on Geoscience and Remote Sensing*, 57(5):2669–2688, 2019.