Shoulder Implant X-Ray Manufacturer Multiclass Image Classification

Name: Rajat Lingwal

1. Introduction

Total Shoulder Replacement (TSR) is a common Orthopedic surgery where a dysfunctional shoulder joint is replaced with an artificial implant. Different manufacturers produce these implants with unique structures. Identifying the implant's manufacturer is crucial for addressing issues that may arise.

This paper aims to explore diverse convolutional neural network (CNN) methodologies for identifying the manufacturer of a shoulder implant. We applied five pretrained models: VGG16, ResNet-150, X-Net, Inception ResNet V2, DenseNet201, and a Vision Transformer. Out of the all models we tried, DenseNet201 achieved the highest accuracy, reaching 73%.

2. Dataset

Maya Stark collected the images at BIDAL Lab at SFSU for her MS thesis project. These images were sourced from the UW shoulder site, manufacturer websites, and Feeley Lab at UCSF. Initially, the collection comprised 605 X-ray images. However, eight images suspected to be duplicates from the same patients were eliminated, leaving a final count of 597 images. These images represent products from various manufacturers, including 83 from Cofield, 294 from Depuy, 71 from Tornier, and 149 from Zimmer.

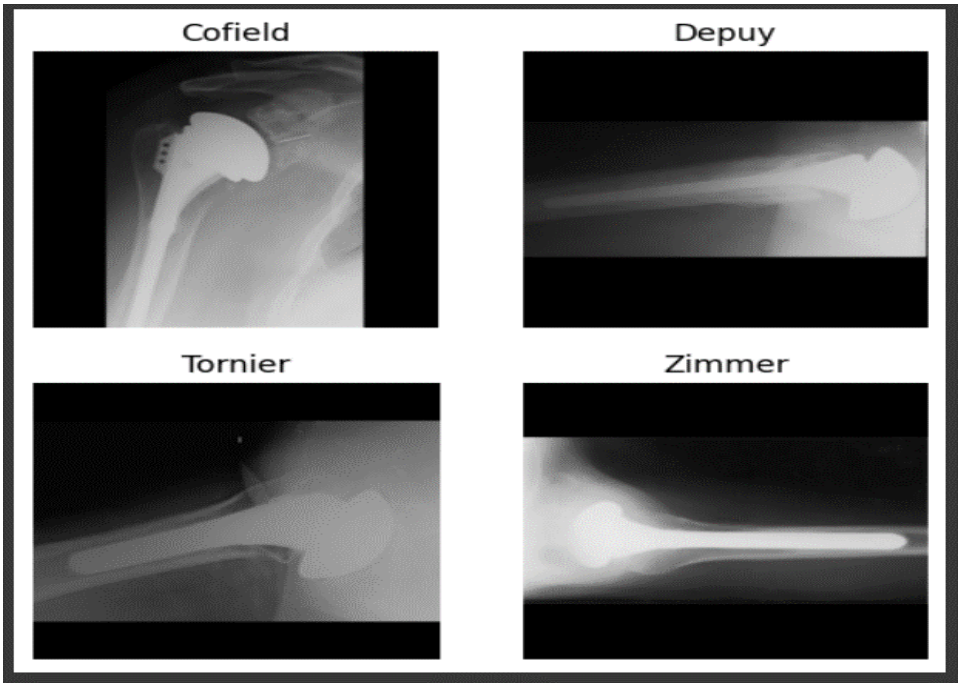

Figure 1: Images of different manufactures in the data set

3. Data Preprocessing

4.1. Image Processing and Label Mapping

First, we converted images into NumPy arrays, and stores them along with their corresponding labels. We resized the images to a standard dimension of 256x256 pixels and stores them in a NumPy array. The labels are assigned based on the filenames, and a mapping from label indices to label names is created using a dictionary. Label 0 is 'Cofield', 1 is 'Depuy', 2 is 'Tornier' and 3 is 'Zimmer'. Finally, the images and labels are stored in NumPy arrays, and a mapping from label indices to label names.

4.2. Data Augmentation

Deep learning models typically demand substantial volumes of training data to achieve optimal performance. However, with only 597 images available for this task, it falls short of the quantity necessary to train a highly accurate model capable of classifying each class effectively. To address this limitation, the data augmentation technique is employed to augment the training dataset. Data augmentation involves applying various transformations

to each image in the dataset, thereby expanding the dataset and enhancing the model's ability to generalize across different variations in the data.

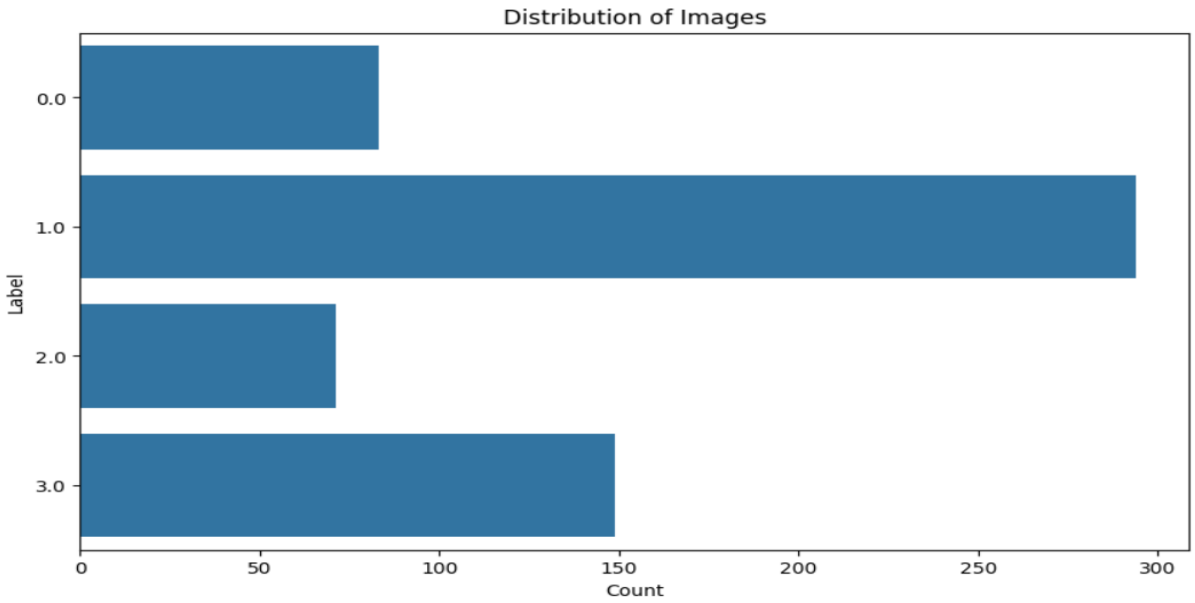

Figure 2: Images of different manufactures in the data set

As shown in Figure 2, our image data is imbalanced. To address this, we used data augmentation technique. However, due to limitation of high computation resources, we only applied data augmentation to balance the dataset. As a result, the size of our image dataset has increased to 1639.

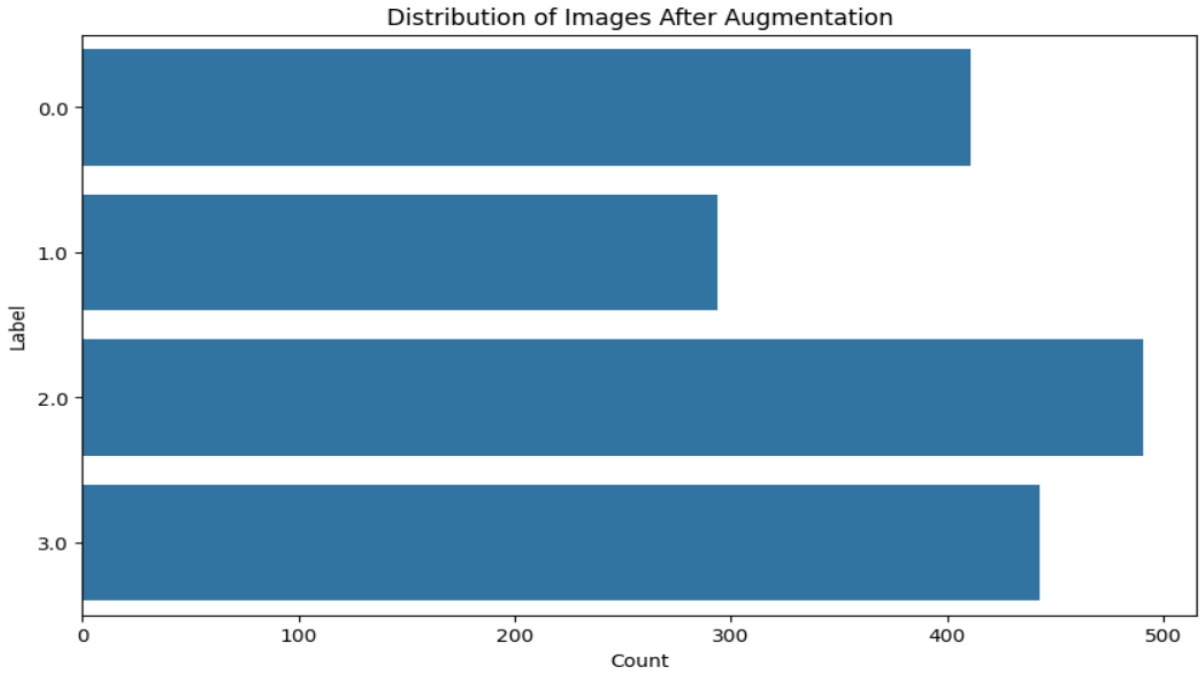

Figure 3: Image distribution after augmentation

4.2. Normalization and Split Data

In machine learning tasks, particularly when handling numerical or image data, normalizing the data is frequently essential. Accordingly, we adjusted the pixel values of the images to fall within a range of 0 to 1 by converting their data type to 'float32' and dividing by 255.0. Subsequently, we divided the data into training and testing sets using the train_test_split function from the scikit-learn library. The training set encompasses 80% of the data, while the remaining 20% constitutes the testing set.

4. Experiments

We utilized 6 CNN pretrained models and Vision Transformers on Google Colab, harnessing its GPU for efficient computation. Initially, we standardized parameters across 6 CNN models which are pretrained on ImageNet and excluding their top classification layers. We then froze the base model layers to retain their pretrained weights and constructed a custom classification head comprising a GlobalAveragePooling2D layer, followed by two dense layers with 256 units and 'relu' activation, capped off by a dense layer with 4 units and 'softmax' activation. Model architecture was summarized using the Adam optimizer, 'sparse_categorical_crossentropy' loss, and a learning rate of 0.001, with training conducted over 20 epochs and a batch size of 32. Among these models, DenseNet201 showed the most promising results, prompting further fine-tuning efforts. To enhance performance, we expanded the scope of trainable layers in the DenseNet201 model, incorporated dropout

layers with rates of 0.5 and 0.3 to mitigate overfitting, adjusted the learning rate to 0.0001 for more precise updates, and introduced a learning rate scheduler through the ReduceLROnPlateau callback to ensure stable convergence.

5.1. VGG-16

VGG-16 was chosen as the primary deep learning model for its efficacy in image classification tasks, VGG-16 consistently delivers high accuracy rates. However, it's worth noting that this architecture demands approximately 144 million parameters, rendering it computationally intensive and time-consuming to train, especially on standard laptop setups. The architecture itself is characterized by numerous 3x3 convolutional filters with a stride of 1, complemented by max-pooling layers featuring a 2x2 window and a stride of 2. After convolution, it features multiple fully connected layers with 2048 units each, culminating in a 1000-dimensional output for ImageNet's 1000 classes.

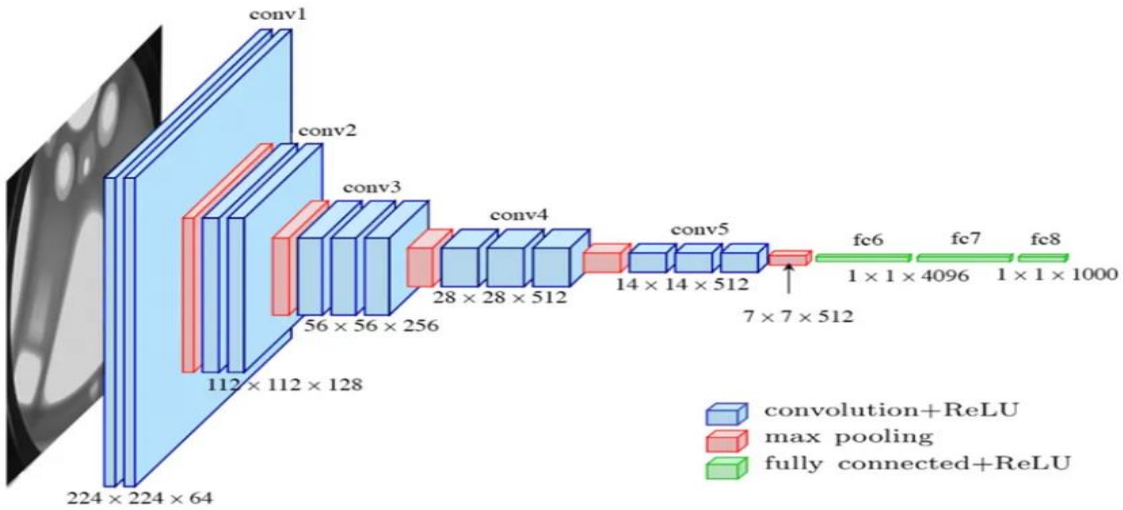

Figure 4: VGG-16 Architecture
Source: https://medium.com/@mygreatlearning/

Classification Report:	precision	recall	f1-score	support
Cofield	0.00	0.00	0.00	15
Depuy	0.54	1.00	0.70	65
Tornier	0.00	0.00	0.00	11
Zimmer	0.00	0.00	0.00	29
accuracy			0.54	120
macro avg	0.14	0.25	0.18	120
weighted avg	0.29	0.54	0.38	120

Figure 5: VGG-16 Classification Report

5.2. RESNET-150

ResNet-150 is an extension of the ResNet architecture, notable for its remarkable depth with 150 layers. It addresses the challenge of training very deep networks by introducing residual blocks, which enable the direct flow of gradients during training. These blocks consist of convolutional layers followed by batch normalization and ReLU activation functions, along with skip connections to facilitate gradient flow. Despite its depth, ResNet-150 maintains computational efficiency and is highly effective for tasks like image classification and object detection.

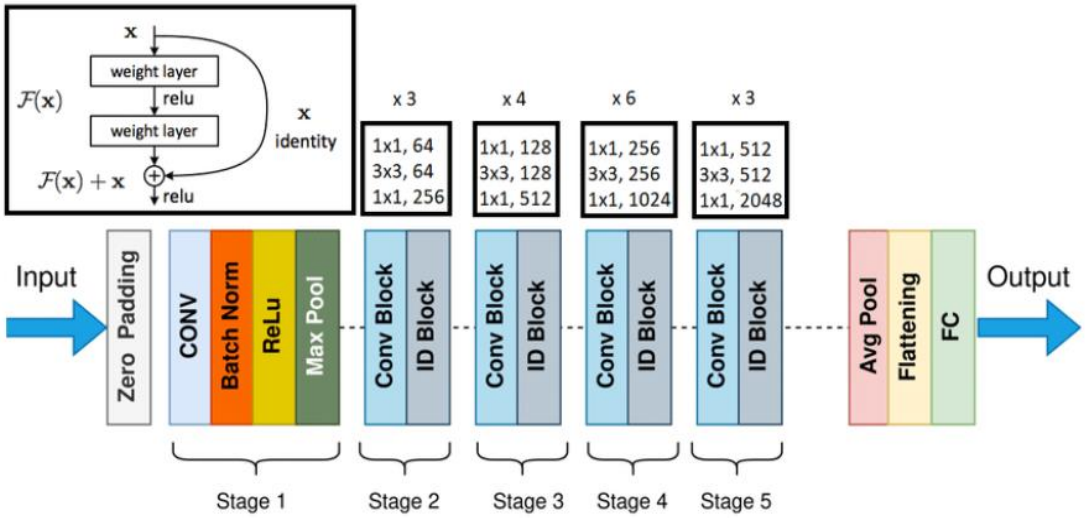

Figure 6: ResNet Architecture
Source: [The-structure-of-ResNet]

Classification Report:

	precision	recall	f1-score	support
Cofield	0.00	0.00	0.00	15
Depuy	0.54	1.00	0.70	65
Tornier	0.00	0.00	0.00	11
Zimmer	0.00	0.00	0.00	29
accuracy			0.54	120
macro avg	0.14	0.25	0.18	120
weighted avg	0.29	0.54	0.38	120

Figure 7: ResNet-150 Classification Report

5.3. X-NET

X-Net utilizes an encoder-decoder architecture for image segmentation. The encoder extracts feature through convolutional layers and downsamples the image, while the decoder upsamples to generate a segmented mask. To balance downsampling and maintain boundary detail, X-Net incorporates two encoder-decoder modules in succession and stores encoder feature maps for dense feature map creation during decoding. Regularization techniques, including L2 norm regularization, are applied to prevent overfitting.

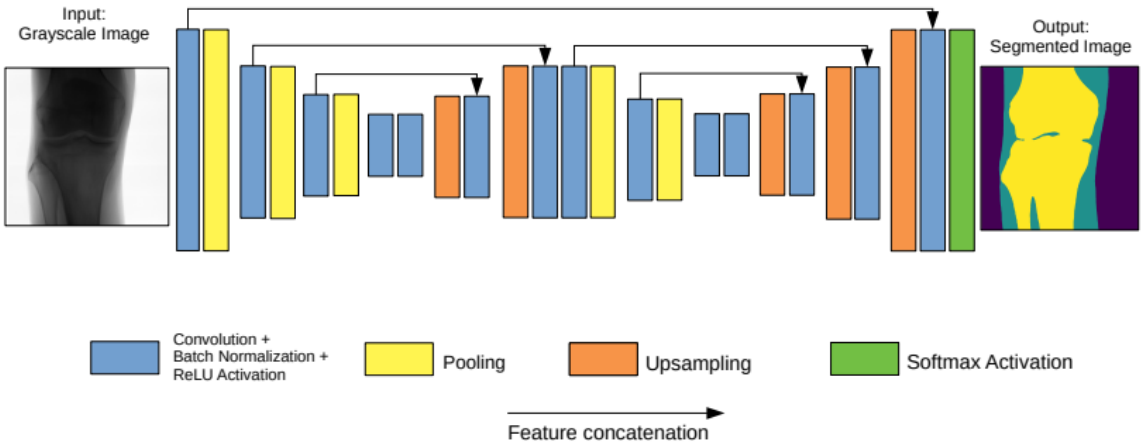

Figure 8: X-Net Architecture
Source: [XNet: A convolutional neural network (CNN)]

Classification Report:	precision	recall	f1-score	support
Cofield	0.86	0.59	0.70	86
Depuy	0.51	1.00	0.68	70
Tornier	0.81	0.81	0.81	88
Zimmer	0.73	0.38	0.50	84
accuracy			0.68	328
macro avg	0.73	0.70	0.67	328
weighted avg	0.74	0.68	0.67	328

Figure 9: X-Net Classification Report

5.4. Inception RESNET V2

Inception-ResNet-v2 stands as a deep convolutional neural network design that merges features of Inception and ResNet architectures. It integrates ResNet's residual connections to ease the training of deep networks and incorporates Inception's modules for effective multi-scale feature extraction. Typically, this architecture comprises stacked modules, alternating between Inception and residual

connections. This setup fosters hierarchical feature extraction while mitigating issues like the vanishing gradient problem.

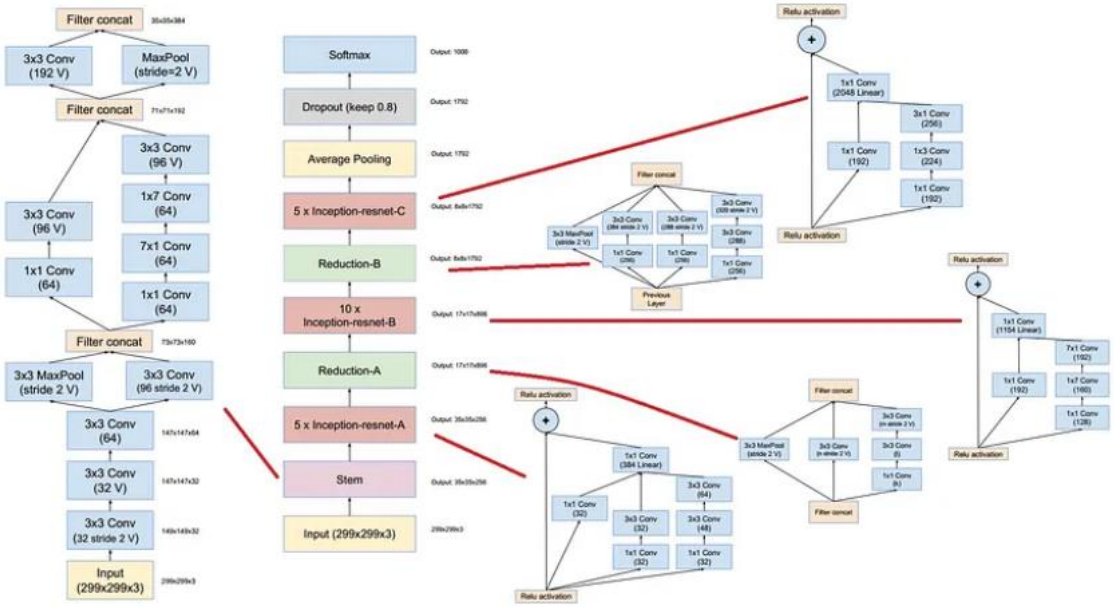

Figure 10: Inception ResNet V2 Architecture
Source: medium.com/the-owl/building-inception-resnet-v2

Classification Report:	precision	recall	f1-score	support
Cofield	0.83	0.45	0.59	86
Depuy	0.51	1.00	0.68	70
Tornier	0.76	0.84	0.80	88
Zimmer	0.62	0.35	0.44	84
accuracy			0.65	328
macro avg	0.68	0.66	0.63	328
weighted avg	0.69	0.65	0.63	328

Figure 11: Inception ResNet V2 Classification Report

5.5. DENSENET201

DenseNet201 architecture is distinguished by its dense blocks, which consist of multiple convolutional layers. These blocks foster dense connectivity patterns, allowing for efficient information flow across the network by receiving input from the preceding block as well as all prior blocks. The fundamental concept of DenseNet involves amalgamating feature maps from each layer to serve as input for subsequent layers, promoting dense interlayer connections. Transition layers interspersed between dense blocks aid in reducing spatial dimensionality and the number of feature maps through batch normalization, 1x1 convolutional layers, and pooling.

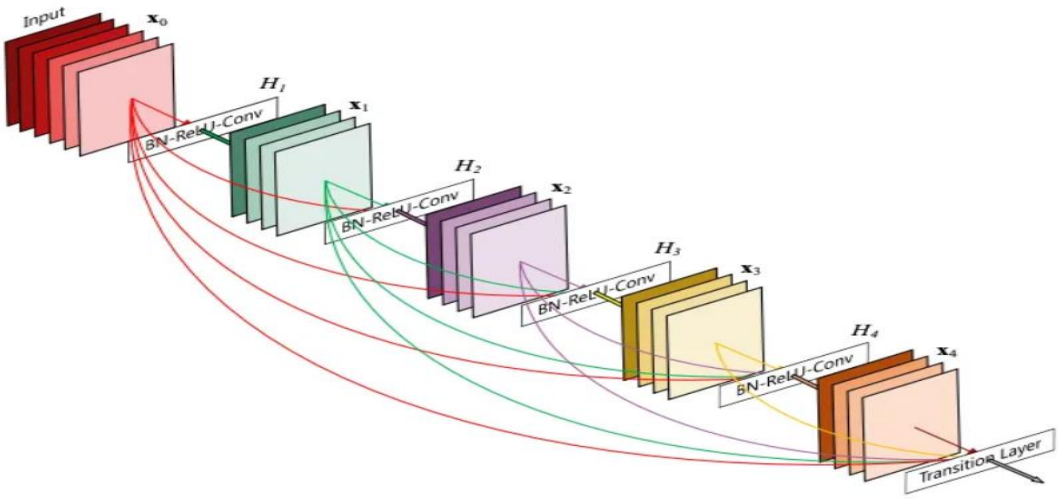

Figure 12: DenseNet201 architecture
Source: https://medium.com/nocoding-ai/densenet121-760df192f12d

Classification Report:

	precision	recall	f1-score	support
Cofield	0.87	0.71	0.78	86
Depuy	0.51	1.00	0.68	70
Tornier	0.94	0.85	0.89	88
Zimmer	0.85	0.42	0.56	84
accuracy			0.73	328
macro avg	0.79	0.74	0.73	328
weighted avg	0.81	0.73	0.73	328

Figure 13: DenseNet201 Classification Report

5.6. Vision Transformer

In the ViT architecture, originally designed for natural language processing, the transformer model is adapted for image classification tasks. Rather than processing word embeddings, it handles two-dimensional image data by dividing it into smaller patches. Each patch is flattened into a vector and mapped to a higher-dimensional space using a trainable linear projection. Additionally, a class embedding is introduced before the sequence of embedded patches, representing the classification output. Positional information is incorporated via one-dimensional positional embeddings. The encoder part of the Transformer architecture processes the embedded patches, and the classification head, employing an MLP with Gaussian Error Linear Unit (GELU) non-linearity, generates the final classification output. Thus, ViT utilizes the Transformer encoder to handle input sequences of embedded image patches, enabling effective image classification.

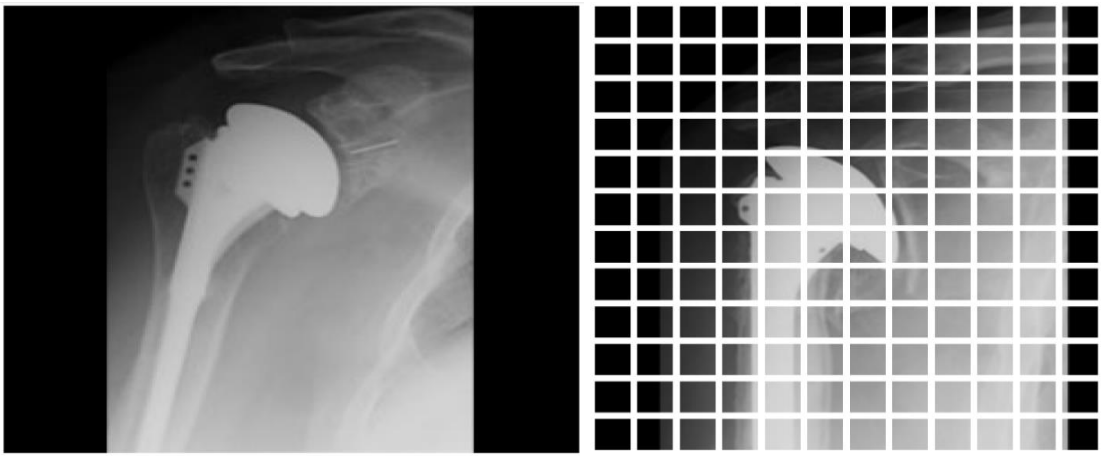

*Figure 14: Splitting image into 20 * 20 patches*

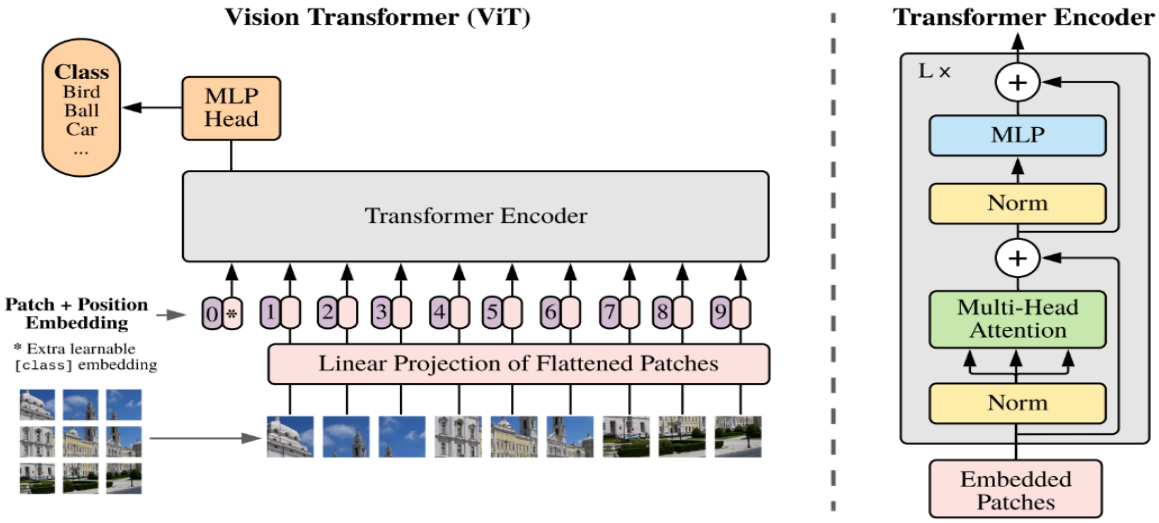

Figure 15: Vision Transformer Architecture
Source: [transformers-everywhere-patch-encoding-technique-for-vit]

Classification Report:

	precision	recall	f1-score	support
Cofield	0.57	0.40	0.47	82
Depuy	0.68	0.86	0.76	59
Tornier	0.45	0.69	0.55	98
Zimmer	0.42	0.21	0.28	89
accuracy			0.52	328
macro avg	0.53	0.54	0.52	328
weighted avg	0.51	0.52	0.50	328

Figure 16: Vision Transformer Classification Report

5. Result and Analysis

As previously mentioned for all the models tested, we utilized identical augmented data and hyperparameters during both training and testing. Of all the models evaluated, DenseNet201 exhibited the highest performance, so we fined tuned it more and achieved an accuracy of 73%.

Model	Accuracy	Precision	Recall	F1-Score
VGG16	54	29	54	38
ResNet152	54	29	54	38
X-Net	68	74	68	67
Inception ResNet V2	64	68	64	63
DenseNet 201	70	76	70	70
DenseNet 201 Fine-Tuned	73	81	73	73
Vision Transformer	52	51	52	50

Table 1: Score of all the implemented models

In Figure 10, the validation loss and test loss remain consistent after a certain number of epochs. Also, from the confusion matrix we can analyse that the most of the labels predicted by the model is true. However, it's evident from the figure that the model's performance is not optimal, likely due to a lack of sufficient data.

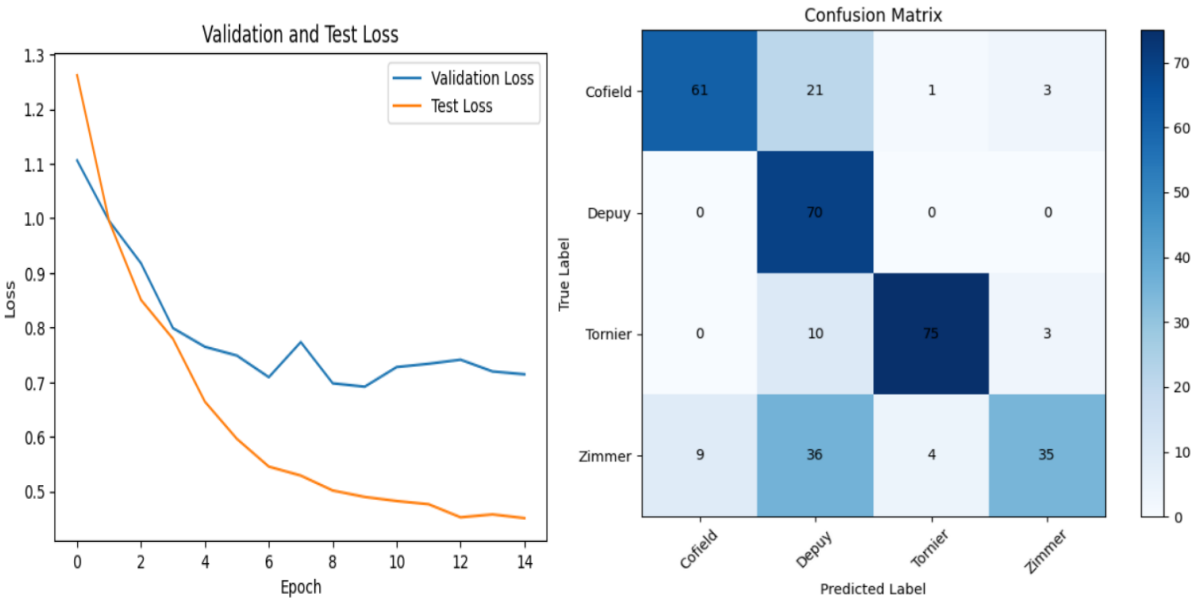

Figure 17: Validation, test loss graph and confusion matrix for DenseNet201

6. Conclusion

In our study, we compared multiple pre-trained CNN models and the Vision Transformer, an innovative attention-based model. While Transformer models show promise in replacing many CNN architectures, they demand substantial data and computational resources. Due to these constraints, the Transformer we used did not outperform in accuracy for our classification task. Nevertheless, our research highlights the potential of CNN models in identifying shoulder implant manufacturers and stresses the importance of exploring diverse methodologies for enhanced accuracy and efficiency.

Enhancements such as data augmentation, transfer learning, and cross-validation have been employed to boost model performance. With expanded datasets and potentially more manufacturers, these deep learning models could achieve even better results, leading to the development of a practical tool to aid doctors in real-world scenarios.
Future advancements may involve exploring alternative deep learning models and tackling challenges in optimizing the performance of the Vision Transformer. Additionally, there is potential for integrating image segmentation techniques with the classification task and further development of a dedicated tool for shoulder implant classification.

7. References

- Bullock, J., Cuesta-Lázaro, C., & Quera-Bofarull, A. (2019, March). XNet: a convolutional neural network (CNN) implementation for medical x-ray image segmentation suitable for small datasets. In Medical Imaging 2019: Biomedical Applications in Molecular, Structural, and Functional Imaging (Vol. 10953, pp. 453-463). SPIE. *XNet: a convolutional neural network (CNN) implementation for medical x-ray image segmentation suitable for small datasets (spiedigitallibrary.org)*
- Gowri Shankar. (2021). Transformers Everywhere - Patch Encoding Technique for Vision Transformers(ViT) Explained. *https://gowrishankar.info/blog/transformers-everywhere-patch-encoding-technique-for-vision-transformersvit-explained/*
- Nocoding AI. (2023). DenseNet. medium. *https://medium.com/nocoding-ai/densenet121-760df192f12d*
- Rahul Gomes. (2022). A Comprehensive Review of Machine Learning Used to Combat COVID-19. researchgate. *https://www.researchgate.net/figure/The-structure-of-ResNet-50-model-19-distributed-under-a-CC-BY-SA-40-license-234_fig5_362394275*
- Rohini G. (2021). *Everything you need to know about VGG16*. medium. *https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918*
- Siladittya Manna. (2019). Building Inception-Resnet-V2 in Keras from scratch. medium. *https://medium.com/the-owl/building-inception-resnet-v2-in-keras-from-scratch-a3546c4d93f0*