Pick a Bone with Machine Learning: Classification of Shoulder Implants

Manny Lazalde
Carnegie Mellon University
mlazalde@andrew.cmu.edu

Ivy Nuo Chen
Carnegie Mellon University
ivynuochen@cmu.edu

Michael Vinciguerra
Carnegie Mellon University
mvincigu@andrew.cmu.edu

Abstract

Patients with chronic shoulder pain may elect to undergo Total Shoulder Arthoplasty (TSA), effectively replacing their shoulder joint with an implant to improve quality of life. These implants tend to degrade over time, and sometimes require repair or replacement. Proper identification of the implant is necessary prior to surgery and can cause issues when the information is unknown. To identify the implant, X-rays are taken and examined by medical professionals which can lead to a delay in surgery. In this work, shallow and deep machine learning models are studied for the use of classifying X-ray images of shoulder implants made by different manufacturers. Through data augmentation and transfer learning, our ResNet-50 architecture proves promising for properly classifying implants by manufacturer.

1. Introduction

Shoulder pain is one of the leading causes of musculoskeletal pain, with rotator cuff tears and shoulder arthritis acting as the leading cause of most reported shoulder pain cases [8]. This shoulder pain significantly decreases the quality of life of patients, making simple tasks prior to injury extremely difficult.

Depending on the severity of the shoulder injury, many patients undergo Total Shoulder Arthroplasty (TSA) to replace the shoulder joint with an implant [1]. TSA has been proven to significantly increase quality of life of patients through reducing pain and increasing patient range of motion [1].

Over the span of many years, however, the TSA implant is prone to degradation and sometimes requires surgical replacement or repair. Since there are several different models and manufacturers of the shoulder implants, the surgeon must confirm the type of implant residing in the patients body prior to surgery. While normally not an issue, there are times when this may be unknown information. For instance, patients may move countries, or the medical records may not explicitly list the make and model of the implant [7].

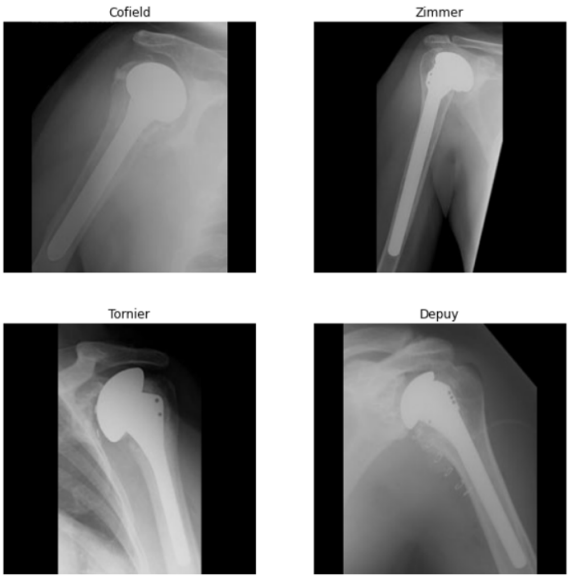

Figure 1. Sample of data showing the different manufacturer's implants. The details that separate these different models cannot be easily discerned at a glance.

In order to determine the model and manufacturer of the shoulder implant, an X-ray is taken of the patient and examined by medical experts. These medical experts must be familiar with all the different types of shoulder implants, requiring significant effort to properly identify the correct implant. Each manufacturer has slight differences in design, as seen in Figure 1. As can be seen, the differences between each implant are often small details such as the shape of a specific part or the number of holes in another part.

A more efficient and faster method to classify the shoulder implants by model and manufacturer would be to leverage the power of machine learning. Deep learning with Convolution Neural Networks (CNN) have been proven ef-

fective in many X-ray image classification problems [4]. In this study, we aim to prove that machine learning can properly identify TSA implants by manufacturer. Although the model of the implant is also needed by the surgeon, we were unable to find enough data to properly classify the implant by model and manufacturer. However, with more data, the methods proposed in this study should be able to properly classify the implant by manufacturer and model.

Through this study, it can be shown that deep learning is able to correctly classify shoulder implants from X-ray images with a high accuracy and F1 score. The ResNet-50 architecture proved to be the most promising method, but only with transfer learning and data augmentation. Shallow machine learning methods proved ineffective for solving this classification problem.

2. Related Work

While there has been much work in the medical imaging community on classifying X-rays, work on classifying shoulder implants from X-Ray images has been rather scant until recently. In 2018, there was a work on detection and segmentation of shoulder implants from X-ray imaging. This work only serves to isolate the TSA implant in the image; the work did not yet attempt classification. [6].

There was a paper published in early 2020 that utilizes deep learning to classify TSA implants from X-ray imaging. [7]. The study presented here is modeled off of the one in this paper as a basis for comparison and confirmation of previous results. The dataset utilized in this study matches the one in the previous 2020 paper for comparison purposes. Additionally, the same deep learning and shallow methods reported in that paper are studied here. Ultimately, this study would provide a baseline for this project through the comparison of results; however, additional work is provided here to differentiate the work from the previous paper. In particular, this study explores additional image augmentation techniques to further boost the accuracies for classifying the dataset. In comparison to the previous work referred to in this study, the methods discussed here were able to achieve much higher performance metrics, likely due to the introduction of more data augmentation.

3. Data

The X-ray images of TSA implants were obtained from a dataset available from the University of California at Irvine machine learning repository. This dataset contains 597 x-ray images, each labeled per manufacturer of the TSA implant as either "Cofield", "Depuy", "Zimmer" or "Tornier".

The most important detail to note with this dataset is the class imbalance that exists within it. To be more exact, the dataset is comprised of the following: 83 Cofield, 294 Depuy, 71 Tornier, and 149 Zimmer. A naive solver could guess that each image belongs to Depuy and achieve a 49.2% accuracy. Therefore, achieving a high accuracy on this dataset is necessary to determine that the solver is better than guessing a single class.

Figure 2. Sample of Augmented Data. From top to bottom, left to right: normal, random rotation, random translation, random shear, random scaling, random cropping, horizontal flipping, vertical flipping, and gamma contrasting.

Data standardization was applied to each image by normalizing the pixels such that each image was 250x250x3 pixels and that each pixel value was within the interval of [0,1]. Once the images were standardized, they were saved to another file for future use in all algorithms.

3.1. Data Augmentation

Deep learning requires a large amount of data to train the model. While a large amount of time was spent gathering the 597 images used in this project, this data is insufficient to train an accurate model. Data augmentation is the process of applying transformations to images to produce more training data for a machine learning algorithm. Random translations, rotations, shearing, horizontal flips and vertical flips were applied to each image to generate an addition five images per input, thus creating a total data size of 3,582 images for the deep learning model. More transformations were originally applied, such as random scalings, gamma contrast changes and cropping. However, due to RAM limitations within Google Colab, this additional data could not be utilized. Additionally, for the shallow methods, only random translations, rotations and shearing could be given to the different classifiers to avoid running out of RAM, for a total of 2,388 input images. Images were augmented using the $imgaug$ python library. The effects of the different

transformations for a single image are shown in Figure 2.

4. Classification Experiments

Classification was performed using a variety of Shallow methods, including Random Forest, Logistic Regression, K-Nearest Neighbors. Deep methods were also utilized through the VGG-16 and ResNet-50 architectures. These methods were chosen to demonstrate the variety of classifiers and their performance.

4.1. Shallow Methods

Similar to the referenced work [7], shallow methods of machine learning were applied on the dataset with 10-fold cross validation for averaging the results over the randomized models. The results displayed in Table 1 and Table 2 are the average of the validation folds. Three models were studied in particular, both with and without data augmentation. XGBoost was also attempted initially but issues with Google Colab's RAM prevented implementation of this method. Hyperparameters used for each model were chosen as a point of comparison to model the performance of the models studied by Urban et Al[7]. All of the models operate directly on the pixels in each image. These models are primarily used as a benchmark for measuring the performance of the ResNet-50 architecture reported later in this work. *Sklearn* modules were used for creating the different shallow implementations.

4.1.1 Random Forest

A Random Forest classifier with 500 trees using entropy as the loss criterion was studied. The hyperparameters used in the model match those reported in the original work for comparison. This model uses 500 decision trees to determine what the most important features (pixels) of each image is and models the importance by trying to sort each image as efficiently as possible in order to reduce entropy. This model suffers from the fact that memory prevents the team from studying each individual pixel as a root of a tree. Therefore, 500 estimators were used to get a good estimate while not taxing the system. The results for this model both with and without data augmentation are shown in Table 1.

4.1.2 Logistic Regression

A Logistic Regression solver with stochastic average gradient ascent (SAGA) using L2 regularization was employed. Convergence was not achieved using this model for the hyperparameters. The results of this model with and without data augmentation are shown in Table 1.

4.1.3 K-Nearest Neighbors

A K-Nearest Neighbors classifier with $k = 35$ using the Euclidean Distance metric was applied. K-Nearest Neighbors determines the label of a new sample by finding the label that matches the majority of the k neighbors around it, or in this case, the 35 neighbors around it. A large amount of neighbors were used in this model since there were only four classes for all of the data and a lot of the pixels could overlap or be very close for very different manufacturing models. The results of this model with and without data augmentation are shown in Table 1.

4.2. Deep Learning

In recent years, deep learning has revolutionized image classification. Deep learning has allowed for ever-increasing accuracy on difficult problems, paving the way for novel applications. In particular, CNNs allow for end-to-end learning without requiring tedious hand engineered features of the data. CNN's are able to detect low/mid/high level features through the use of convolutional filters in the network, also allowing for a significant reduction in trainable parameters as compared to a standard neural network [4]. Pooling layers where neighboring pixel values are combined typically through an average or max also allow CNN's to perform classification with a certain degree of translation invariance [4].

In our study, hand crafted VGG-16 and ResNet-50 architectures were made for deep learning. These architectures have been proven to work on this problem with success based on [7], while simultaneously being simple enough to implement by hand in TensorFlow and Keras. Each model was coded in TensorFlow and Keras and then training/testing the models on the fashion MNIST dataset to verify their efficacy. These results are shown in Tables 1 and 2. Note that no hyperparameter tuning was performed on these models for the fashion MNIST dataset, and their performance can likely be improved with longer training and tuning. Once the accuracies became high enough, the models were applied on the TSA implant dataset. Successful results with the VGG-16 on the TSA implant dataset were unachievable; however, the ResNet-50 showed significantly improved classification accuracy through image augmentation, transfer learning, and hyperparameter tuning on the same dataset.

4.2.1 Transfer Learning

Transfer learning is an essential process conducted when the data available for a particular problem is limited, as in this situation for an X-ray implant classification problem with only around 600 images. The idea is that a given architecture has been trained on the ImageNet dataset for image classification on millions of pictures [2]. The associated

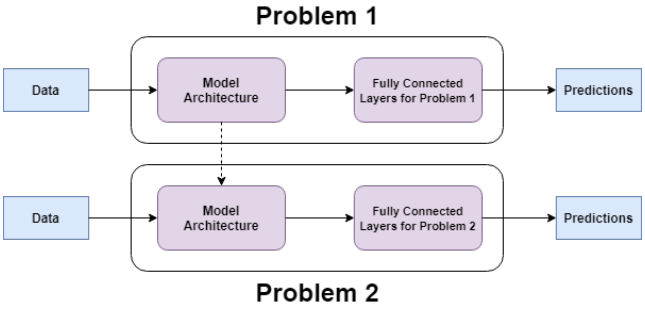

Figure 3. A diagram for understanding Transfer Learning. Weights are taken from one model that has been trained over many images for use in another model. Then, fully connected layers are used to specialize the model to a new problem.

weights with the pretrained network can be adopted for use on a similar network architecture. Many of the early to mid layers in the network will serve to perform basic shared featurization of the images, such as edge detection. Then, the model is trained on the last couple of fully connected layers to specifically apply it to the problem studied in this work. This overall process of transfer learning ensures that the classifier is maximizing its performance. A diagram explaining transfer learning is shown in Figure 3.

4.2.2 Hyperparameter Tuning

Another important aspect of deep learning involves hyperparameter tuning. Hyperparameter tuning is adjusting all the various parameters within the deep learning model to improve performance. Some hyperparameters that were examined include changing the optimizer algorithm, adjusting the learning rate, changing the batch size, setting an exponentially decaying learning rate, and even freezing layers to perform fine tuning of the deep learning models with transfer learning. The best results are displayed in Tables 1 and 2.

4.2.3 VGG-16

The VGG-16 is an extremely popular CNN architecture that is commonly utilized in image classification problems. While fairly straightforward and relatively simple to implement by hand, the VGG-16 is an extremely large network with approximately 138 million parameters [5]. The architecture utilized in this work is composed of many 3x3 convolutional filters with a stride of 1 and max pooling layers of 2x2 filter with stride of 2. Then, these layers are fed into several fully connected layers to an output layer with a softmax activation.

4.2.4 ResNet50

The 50-layer Residual Network, commonly known as ResNet-50, is a popular CNN architecture that has been proven effective for transfer learning problems. There are many different variants of the ResNet, but the ResNet-50 provides sufficient network depth for this problem. The ResNet-50 attempts to address the degradation problem in deep learning, or when increasing network depth causes accuracy to become saturated and then degrade rapidly [3].

The ResNet architecture addresses this problem with a key novel feature: a skip connection. This skip connection is essentially adding the input of several stacked convolutional layers to the output. The addition of a skip connection decreases the likelihood of vanishing gradients as well as allows the network to learn identity mappings [3]. These identity mappings prevent the network accuracy from dropping after saturation.

5. Results and Discussion

As discussed above, data on the classification performance was collected. All data performance data are shown in Tables 1 and 2. Each table metric is calculated by taking the averages of the validation splits over a 10-fold cross validation. Note that the MNIST scores are an exception to the 10-fold cross validation rule, since these were for testing purposes only. Also note that several RAM issues occurred in Google Colab for the augmented data ResNet-50, so scores are calculated from a single fold.

5.1. Classification Accuracy

The simplest way to benchmark performance of each classification method is by calculating the accuracy.

Accuracy is calculated as:

$$accuracy = \frac{correct\ prediction}{total\ predictions\ performed}$$

The accuracy of each classifier is shown in Table 1. The accuracy gives a baseline for comparing the rate at which classifications are correct. From the data, it can be observed that the deep methods perform best. As previously mentioned, however, a naive solver could achieve a 49.2% accuracy by always guessing a single class due to the large class imbalance. Thus, accuracy alone may not be the best representation of performance of a model.

5.2. F1 score

In addition to finding the accuracy for each classifier, the F1 score was calculated. The F1 score takes into account the precision and recall of classifiers and gives a better sense of the classification performance.

The F1 score is defined as:

$$F1 = 2 \times \frac{1}{\frac{1}{precision} + \frac{1}{recall}}$$

Classification Method	Augmented Data?	Accuracy
Random Forest	No	55
Random Forest	Yes	52
Logistic Regression	No	47
Logistic Regression	Yes	42
KNN	No	51
KNN	Yes	50
VGG16	No	49
ResNet50	No	48
ResNet50 - TL	No	67
ResNet50 - TL	Yes	99
VGG16 - MNIST	No	93
ResNet50 - MNIST	No	87

Table 1. Performance of Models with Accuracy in Percent. TL stands for "transfer learning".

Precision is defined as:

$$precision = \frac{true\ positives}{true\ positives + false\ positives}$$

Recall is defined as:

$$recall = \frac{true\ positives}{true\ positives + false\ negatives}$$

The F1 scores for each classifer are shown in Table 2. The F1 score more accurately reflects the performance of the classifiers. When the number of false negatives and false positives are taken into account, the F1 metric allows for a clearer insight into model performance without the distortion from the class imbalance.

Classification Method	Augmented Data?	F1 Score
Random Forest	No	0.36
Random Forest	Yes	0.24
Logistic Regression	No	0.34
Logistic Regression	Yes	0.28
KNN	No	0.25
KNN	Yes	0.33
VGG16	No	0.16
ResNet50	No	0.41
ResNet50 - TL	No	0.62
ResNet50 - TL	Yes	0.99
VGG16 - MNIST	No	0.93
ResNet50 - MNIST	No	0.87

Table 2. F1 Scores

5.3. Confusion Matrix

In addition to F1 scores, confusion matrices can help demonstrate the performance of the classifiers implemented. The confusion matrix does so by denoting predicted vs true labels of each image in the dataset, which allows for analysis into how the classifier performs between classes. In the analysis, confusion matrices such as those seen in Figures 4 and 5 were utilized to look into performance of the classifiers.

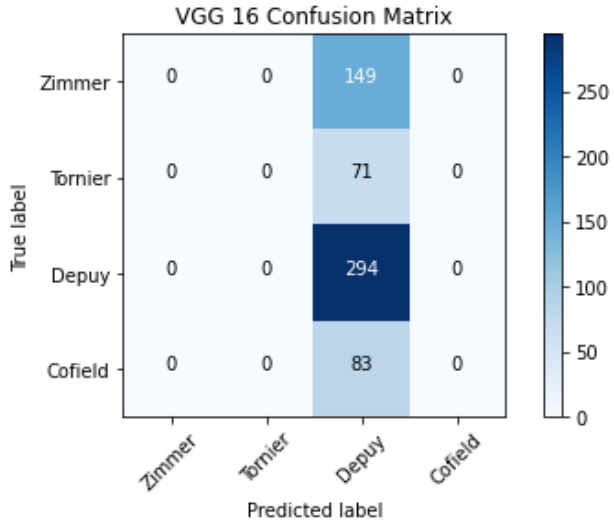

Figure 4. VGG-16 Confusion Matrix, No Transfer Learning, no Data Augmentation

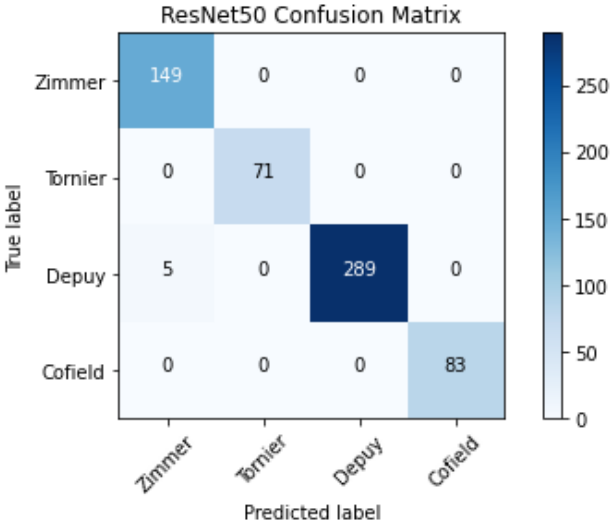

Figure 5. ResNet50 Confusion Matrix, Transfer Learning, Data Augmentation; performance on original dataset

5.4. Analysis

Through this work, deep learning is proven to be successful in classifying TSA shoulder implants based solely off of X-ray imaging. The ResNet-50 model with transfer learning on the augmented data performed the best over-

all, achieving 99% classification accuracy on the original dataset.

It is not surprising to see why the ResNet would perform well on this data - convolutional neural networks work exceedingly well on datasets with images. CNNs are able to pick out fine details in the images that can be used to distinguish each type of implant, such as the thinness of the Zimmer models in comparison to the rest, or the notches on the main ball of the implant that could be used to distinguish the other models.

The success of the model can partially be contributed to the transfer learning used. Once transfer learning is introduced, a roughly 20% increase in F1 score and accuracy for the ResNet architecture was recorded. Transfer learning thus becomes very important for improving the accuracy of the model, and demonstrates that being able to train the base of the model over a multitude and variety of images (Imagenet) can improve the generalizability of these networks for different problems; a model trained on animals, buildings, and other objects was able to become the base for the classification of shoulder implants.

The largest increase in accuracy of the ResNet-50 model came from utilizing the augmented data. A significant increase in accuracy was observed once augmented data was included in the training data. This increase is largely due to the fact that there is more training data - CNNs typically require a lot of training in order to be accurate. By utilizing the data augmentation, six times the amount of original images could be produced to improve training. For problems where large mounts of data is not readily available, data augmentation becomes vital to improve the model's training.

Data augmentation proved to only distract the shallow methods from their goals further. The shallow methods were barely able to perform better than a naive solver without data augmentation, and performed worse once the augmented data was included. Some theories for why shallow methods did not work well for this dataset include:

1. The solvers were operating on the raw RGB pixel values as features. Therefore, it is very unlikely that any features would appear to be dominant and lead the classifiers to being able to determine which features would likely lead to a correct classification. Future work in this area could try turning the images into grayscale to test this hypothesis.

2. The data augmentations serve to confuse the classifiers rather than help them. Since the images had to first be cast into a new size to standardize them, they all have various sized black bars that are introduced to keep the sizes consistent. Performing the transformations on the images also changes these new pixels, which are not going to allow us to achieve higher accuracy since they carry no information. Performing additional standardization to remove these pixels that carry little information could lead to higher accuracy.

3. The inherent bias in the dataset towards one manufacturer is difficult for shallow methods to overcome, especially with the effects of the first two hypotheses. Enforcing an even probability distribution may lead to higher results.

Another interesting aspect of our results is that the VGG-16 model architecture was unable to predict any other class other than majority class (Figure 4). However, we know that the hand coded VGG-16 model worked on the fashion MNIST dataset, so the architecture coding is not the issue. Similar behaviour was recorded in the ResNet-50 architecture, until the correct hyperparameters were selected. Therefore, it is believed that this study was unable to identify the proper hyperparameters for the VGG-16, and the model's performance suffered as a result. Given more time, further exploring other options for the VGG-16, such as coding the model in PyTorch or using the default Keras implementation, would have been possible. In troubleshooting the ResNet-50, a PyTorch implementation was created with a different set of hyperparameters for the same problem. It is a possible that a similar approach would work in the future for the VGG-16 model.

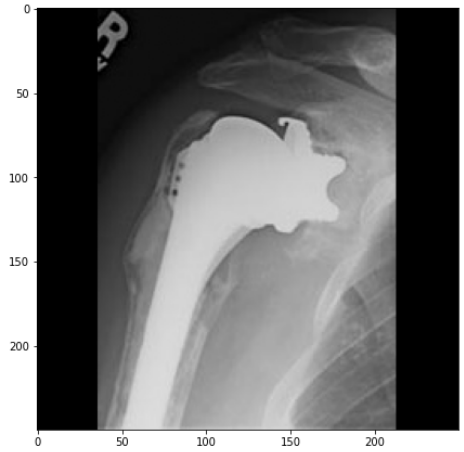

Figure 6. ResNet50, Transfer Learning, Data Augmentation - Misclassified image

The performance of our best model created here, the ResNet-50, also warranted further exploration. The confusion matrix in Figure 5 is used to analyze the results, and an incredibly high performance of the model was recorded. Upon further exploration of the misclassified images, it was observed that the images were extremely different from the standard images that tended to dominate each class. For instance, in Figure 6, it appears as though the glenoid cavity

has also been replaced with an implant, in addition to the standard TSA implant. Given the fact that this image is extremely different from others in the dataset that only have the TSA implant, it makes sense that the model is unable to properly classify it. Therefore, the ResNet50 model has excellent classification performance despite a few misclassifications.

6. Conclusion

Deep learning models are effective at classifying TSA shoulder implants through X-ray imaging. To make these models more effective, transfer learning, data augmentation, and hyperparameter tuning can be used to greatly improve classification accuracy. With more data, a tool could be created to classify TSA implants by manufacturer and model to assist in cases where repair or replacement of existing unknown TSA implants is needed. Future extensions of this work gather more data to further identify the TSA implants by model and manufacturer, as well as trying other deep learning architectures.

7. Team Contributions

For this project, Michael implemented the shallow machine learning methods, including Random Forest, Logistic Regression, and K-Nearest Neighbors. He also explored gradient boosting, but decided not to pursue it due to poor results and long computation times. Michael implemented the image augmentation pipeline, allowing us to have more training data for the deep learning methods. Manny worked on the deep learning methods, coding the ResNet50 and VGG-16 in tensorflow and keras from scratch. He also implemented the ResNet50 in pytorch for troubleshooting. Ivy assisted Manny with the deep learning methods, working on hyperparameter tuning to obtain the best performance. We all worked on the final report and presentation equally.

8. Acknowledgements

We would like to thank our group mentor and TA Akansksh Shetty for his consistent help throughout this project. He helped with our deep learning methods especially, giving us advice that allowed us to succeed. We would also like to acknowledge Professor Farimani for his teaching this semester and providing us the background needed to properly utilize these machine learning methods.

9. Supplementary Material

Note that all code can be found on the following GitHub link:

https://github.com/mannylazalde/ShoulderImplantClassification.

The dataset can be found:

https://archive.ics.uci.edu/ml/datasets/Shoulder+Implant+X-Ray+Manufacturer+Classification.

References

[1] William P Barrett, JL Franklin, SE Jackins, CR Wyss, and FA Matsen 3rd. Total shoulder arthroplasty. *JBJS*, 69(6):865–872, 1987. 1

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 3

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[4] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017. 2, 3

[5] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 4

[6] Maya Belen Cervantes Gautschi Stark. Automatic detection and segmentation of shoulder implants in x-ray images. 2018. 2

[7] Gregor Urban, Saman Porhemmat, Maya Stark, Brian Feeley, Kazunori Okada, and Pierre Baldi. Classifying shoulder implants in x-ray images using deep learning. *Computational and Structural Biotechnology Journal*, 18:967, 2020. 1, 2, 3

[8] Michelle Urwin, Deborah Symmons, Timothy Allison, Thérèse Brammah, Helen Busby, Morven Roxby, Alicia Simmons, and Gareth Williams. Estimating the burden of musculoskeletal disorders in the community: the comparative prevalence of symptoms at different anatomical sites, and the relation to social deprivation. *Annals of the rheumatic diseases*, 57(11):649–655, 1998. 1