Classification of Ultrasound Images of Joints using Neural Networks

Alfredo Funicello, Shihab Hamati

October 5, 2022

Abstract

This project is an exploration and application of the value that Neural Networks can bring to the medical field. Specifically, Convolutional Neural Networks (CNN) are exploited to the multi-class classification problem of correctly identifying ultrasound images from patients.

The work is divided into two main sections. The first section implements and tunes a CNN to classify which of three joints does an image belong to (knee, elbow, ankle, or none). An initial CNN inspired by our literature review and based on the architechture of VGGNet is analyzed and enhanced through guided hyper-parameter tuning. The best performing CNN architecture for this task achieves 0.895 accuracy.

The second section implements a CNN to identify which of 3 standard cross-sections used by radiologists was used to capture the ultrasound images of knees. In addition to the above optimization approaches of section 1, in this section the use of transfer learning is explored, both from the ImageNet pretrained CNN and from section 1's CNN. It was found that training a CNN from scratch only achieved slightly better overall accuracy of 0.919.

1 Introduction

This ML project applies neural networks, in particularly Convolutional Neural Networks, to a multi-class classification problem in the field of medical images. The project and the process of writing this report, have been an opportunity to deepen the understanding of CNNs and work with actual real life data in the form medical ultrasound scans from real patients. The work is an exploration of a use case towards a full project of the Lab aimed to increase access to joints disease detection devices to vulnerable populations of patients which currently have to undergo the checking procedure in person in an hospital multiple times per week. The report covers the process of solving a multi-class classification of the joints from the ultrasound images, the categories include Elbow, Knee, Ankle and Other for other types of joints; the task included revision of related literature, network architecture design and exploration. The second task consists in angle detection from pictures of knees only; both neural network classification, transfer learning and k-means clustering on extracted features have been compared as possible approaches.

The datasets used in this project are from real patients provided by the Policlinico of Milan, and the project is part of a collaboration with the EveryWare lab at the University of Milan. Furthermore, access to the lab's server and A100 GPU allowed this project to conduct the numerous CNN experiments, which would be challenging on regular laptops otherwise.

The authors of this report are grateful to EveryWare's kind support with their time, their resources, and their domain knowledge.

2 Background information

2.1 What is machine learning

Machine Learning (ML), or automated learning, is the process through which a computer gets programmed so that it can learn from the input we give it. The objective of the operation is that the computer, by observing the patterns in the data that it gets passed, which represent past experience

on the problem, can output a program hopefully able to perform a prediction task when a new, before unseen, example from the said problem is shown.

The problems in which past data is used to predict the future are called Data Inference problems. ML is a tool through which different Data Inference problems can be addressed; the two most common approaches are Supervised Learning, where prior labels of past data are available for the algorithm to see, and Unsupervised Learning, where the algorithm comes up with its own data labels from the data.

Since the project is based on the Supervised Learning this framework of approach is briefly presented.

The goal of the Supervised Learning procedure is to find, through a learning algorithm, a mathematical function (called predictor) that associates data points and labels, both coming from the same unseen distribution.

A learning problem can be formally defined through the couple (D, l) where D is an unknown *statistical distribution* generating the points x and l is a *loss function*. The entirety of all the possible points the data domain is called X. Given a certain point $x \in X$, x_i is referred as the *ith*-feature of x. Y is the set of all possible labels for a given data point in X. Y can be either numerical $Y \in R$, in instances named regression problems, or categorical with $|Y| < \infty$, as it is the case for this project, classification problems.

A predictor is a function $f : X^d \mapsto Y$ and the key to solving an inference problem is finding a predictor which is able to generate predictions $\hat{y} = f(\tilde{x})$ with small error. The amount of error from a prediction can be observed by computing the aforementioned *loss function* $f : Y \times Y \mapsto R$ represented as $l(y, \hat{y})$ which compares the true y label of the point x to the predicted label $\hat{y}$ output of the predictor on the point x with a measurement chosen depending on the nature of the problem. The predictor functions are output of a *learning algorithm* A which receives in input a set S of examples (x, y), drawn independently from D, referred as the training set; $A(S) = f : X \mapsto Y$.

2.2 Neural Networks and Deep Learning

Neural networks are powerful prediction tools for solving inference problems. They have shown good performances in many complex tasks such as speech recognition or image recognition. Their structure is loosely modelled around the functioning of biological neurons; a network consists of multiple basic computing devices, the neurons, that are connected to each other creating complex networks of communication.

A neural network can be described as a directed graph whose nodes correspond to neurons and edges correspond to links between them. Each neuron receives as input a weighted sum of the outputs of the neurons connected to its incoming edges. *Feedforward* networks are networks where the underlying graph does not contain cycles and are the most basic architecture for prediction purposes. They're composed by multiple node layers, divided in an input layer, one or more "hidden layers" and an output layer; each of the nodes in the layers is connected to the next layer by weighted edges, which, along with the bias parameters, belonging to the nodes, are the numerical parameters that get updated as the learning procedure proceeds.

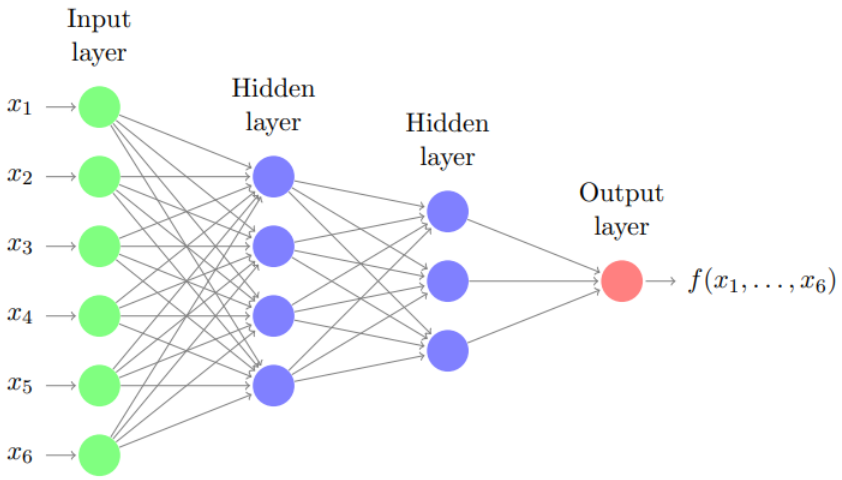


Figure 1: An illustration of a multylayered FeedForward Neural Network, courtesy of the handouts of Professor Cesa Bianchi

We can call $G = (V, E)$ the directed acyclic graph describing a feedforward neural network, each node j in the hidden layers and output layer computes $g(x) = \sigma(w^T x)$ where w is the vector of the weights associated to the edges $(i, j) \in E$, x is the vector of the values of the nodes i on the preceding layer connected to node j and σ is the *activation function* of choice.

The weights $w_{i,j} \in R$ associated with every edge $(i, j) \in E$ are represented by the weight matrix W with dimensions $|V| \times |V|$ where 0 is assigned to edges $(i, j) \notin E$.

The graph G, the weight matrix W and the activation function σ define the function $f_{GW\sigma}$ computed by the network. Given that we define as $w(j)$ the vector of weights w_{ij} for every node $i \in V$ that has an edge going into j, $v(j)$ as the vector of values v_i for every node $i \in V$ that has an edge going into j, and j is a node in all layers but the input one; the computation of $f_{GW\sigma(x)}$ for the data point $x^d \in R^d$ is computed as follows:

- Each node in the input layer takes value $v_i = x_i$
- Each remaining node takes value $v_j = \sigma(w(j)^T v(j))$
- $v_k = f_k(x)$ where k is the k-th output node

The vector $f(x) = (f_1(x), ..., f_n(x))$ where each of the values is from an output node will be the output of the network.

The weight matrix W changes as the training processes, the values get updated through stochastic gradient descent,

$$w_{i,j} \leftarrow w_{i,j} - \eta_t \frac{\partial l_{Z_t}(W)}{\partial w_{i,j}} \tag{1}$$

where $l_t(W)$ denotes the loss of the net on the example t. Given that minimizing the training error based on W is an NP-hard problem, the algorithm empirically finds a local minima of the loss function. The procedure to perform gradient descent on neural networks is called *Backpropagation*.

2.3 Convolutional Neural Networks

One of the major problems in the field image recognition is that distinctive features of an object can appear in various locations of the input image.

Since the precise location of a feature is not relevant to the classification of the subject, Convolutional layers using kernels, based on the concept of weight sharing, have been employed in custom designed architectures for image recognition problems called Convolutional Neural Networks.

Convolutional Neural Networks are analogous to traditional neural networks in that they are comprised of neurons that self-optimise through learning. A notable difference is that CNNs are primarily

used in the context of pattern recognition within images. These networks, employing some particularly engineered layers, are able to maintain the spatial information of the inputs (images) which is an important feature making them more suited for image-related pattern recognition tasks; all the while also reducing the number of parameters required to set up the model which is an important accomplishment towards avoiding overfitting, a danger that becomes particularly insidious when dealing with feature rich objects such as image data. [ON15]

CNN architectures leverage two specifically engineered types of layers; convolutional layers and pooling layers that attach to a set of basic fully connected layers in the tail of the architecture as the diagram in figure 3 shows.

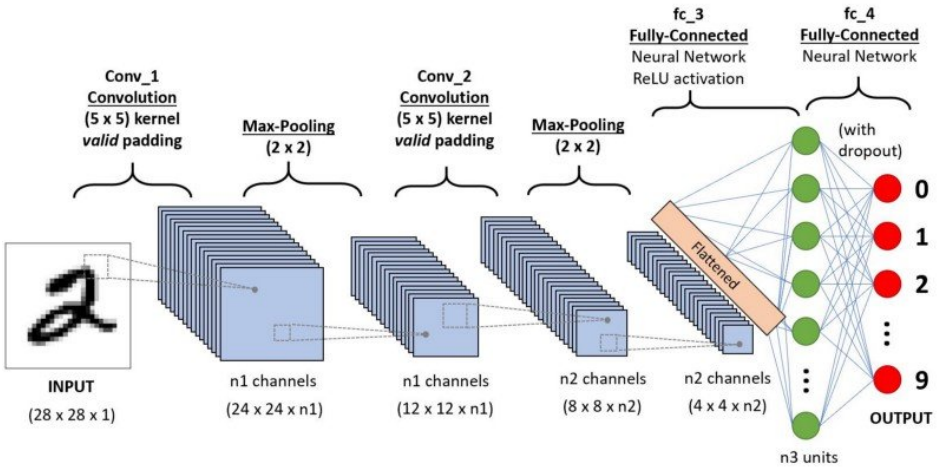


Figure 2: Diagram of a CNN employed for handwritten digits recognition. [RP19]

A convolutional layer contains a set of filters (kernels), parameters of which are to be learned throughout the training. These kernels, usually of much smaller size compared to the input image, convolve over the image by sliding across the height and width performing the dot product and creating an activation map; the output of a convolutional layer is exactly the set of activation maps each one result of a single filter.

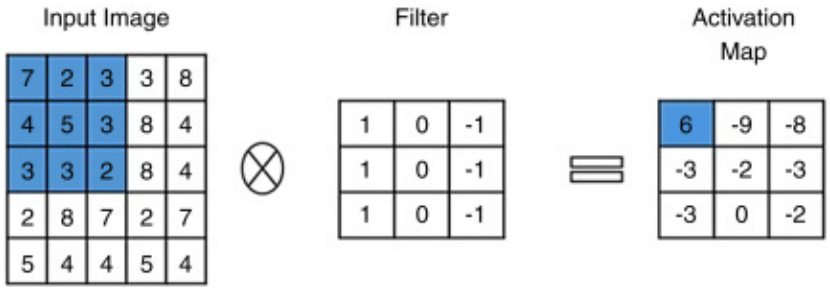


Figure 3: Example of a convolution operation computed by a convolutional layer using a filter [MW21]

Pooling layers perform a sample of the input by sliding over the image and performing a down-sampling operation such as taking the max value recorded in the sampled portion, when talking about MaxPooling layers, or averaging the values in the portion, when talking about Average pooling layers; an example of the operation and its result is in figure 4. These layers usually sit after a convolution layer further reducing the parameters in the system.

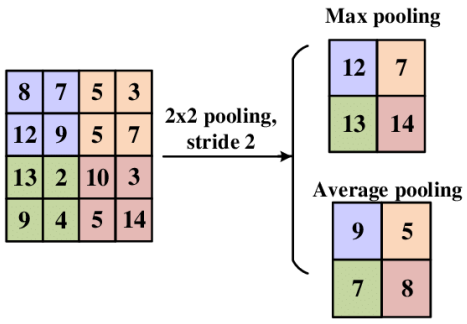


Figure 4: Example of the max and average Pooling operation [MW21]

The final section of a CNN is composed by a set of fully connected layers, the intuition behind this kind of architecture is that once the specialized layers will have gathered important features from the images, through the activation maps, the activation of each of these features will then be interpreted just like a normal NN would do leading to an output prediction. Figure 5 showcases the activations result of the first convolutional layer of a CNN after training on the MNIST database; as it can be seen the CNN collected some common characteristics from the handwritten digits.

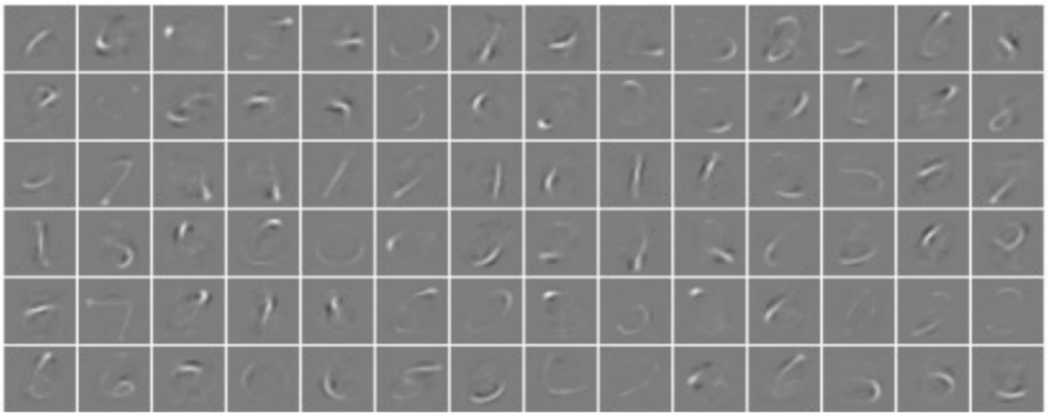
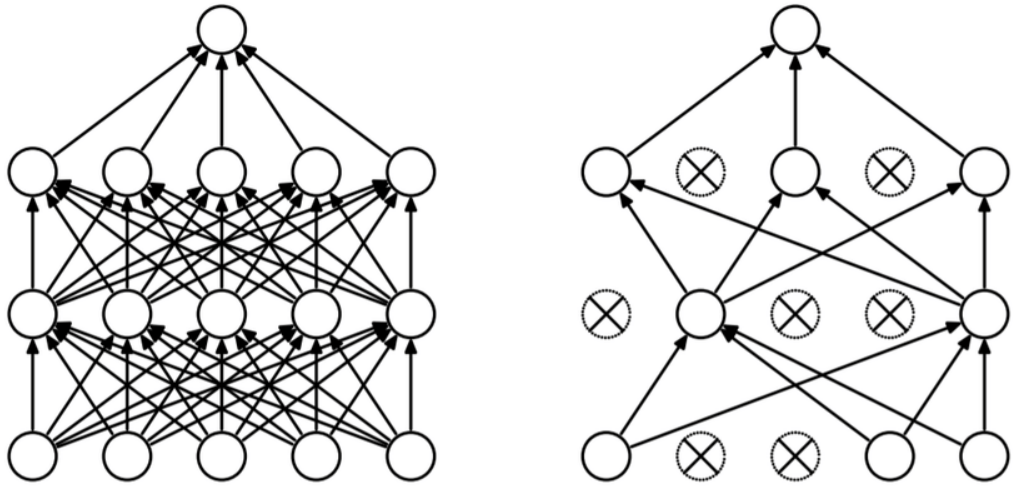


Figure 5: The different maps of activations result from the kernles of the first convolutional layer of a CNN after training on the MNIST database [ON15]

A layer that is not only characteristic of Convolutional Neural Networks, but is of interest given that it has been employed in the architecture presented in this report, is the dropout layer; the layer selects and sets units coming from the preceding layer to 0 with a random probability. It offers a very computationally cheap and remarkably effective regularization method to reduce overfitting and improve generalization error, which as already stated is of extreme importance when working in the image field but proves useful in deep neural networks of all kinds. Figure 6 shows a practical representation of the functioning of the layer.



(a) Standard Neural Net (b) After applying dropout.

Figure 6: Randomly selected inputs coming from previous layers are set to 0, essentially killing the connection for that learning step[Ara20]

2.3.1 Literature Review

A fundamental aspect of the work process of this project has been a thorough review of the pertinent literature, starting from the work that led to the birth of the CNNs from Yann LeCun.

The first important paper from Yann LeCun et al., part of a 10 year research process that will give grounds for modern CNNs is "Backpropagation Applied to Handwritten Zip Code Recognition"; the backpropagation algorithm is employed in the paper for constructing a classification network for the problem of recognizing handwritten digits. The important breakthrough is the observation that, in the environment of complex tasks such as image recognition, the ability of learning networks to generalize can be greatly enhanced by some design choices that aim to constrain locally the feature computation achieving a reduction in the number of free parameters in the network [LBD+89].

In the paper for the first time the learning network is directly fed images rather than feature vectors and the architecture presented is a convolutional neural network. Figure 7 showcases the 3 hidden layers H1 and H2 that output feature maps and end in a fully connected layer.

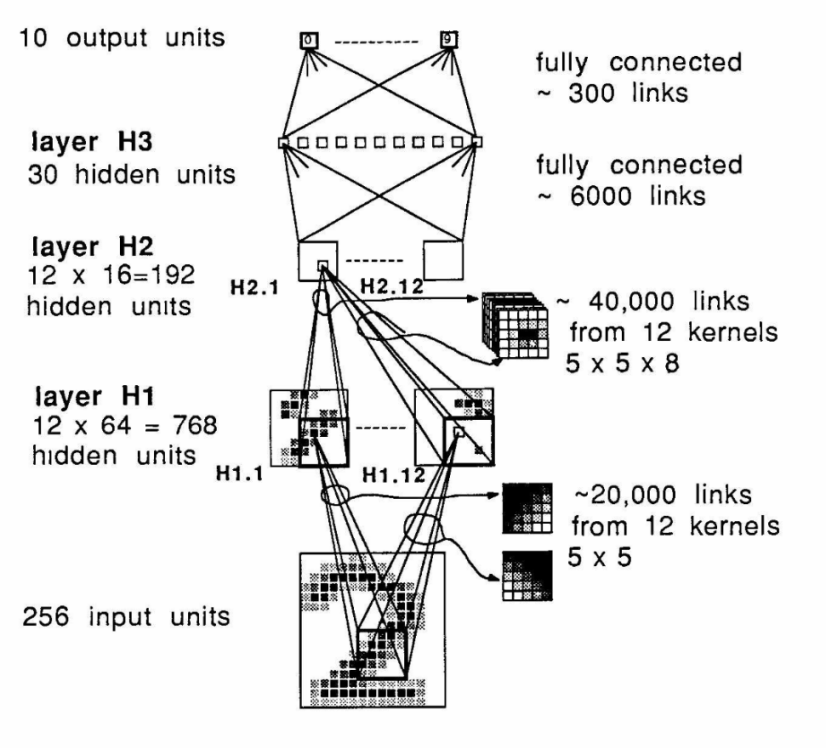


Figure 7: The architecture presented in [LBD+89]

In the successive paper "Generalization and Network Design Strategies" LeCun showcases some more architecture design approaches to the handwritten digit recognition problem.

It is shown that dropping position specific information can be afforded and greatly reduces the size of the space of possible functions that the network can generate, without overly diminishing its computational power.

According to LeCun et al. "constrained networks", namely networks employing convolutional layers, are especially better suited to the field of image recognition, compared to fully connected architectures, given the advantages of being able to extract local features and combining them to form higher order features with a grade of invariance to position in the image.

Fully connected networks illustrated in the paper show poor generalization performances, while impressive performances on the test set are achieved by the two convolution equipped networks. "Constrained network" and "Constrained network 2", ass called in the table from the paper in figure 8, differ just on the amount of kernels and on the deepness.

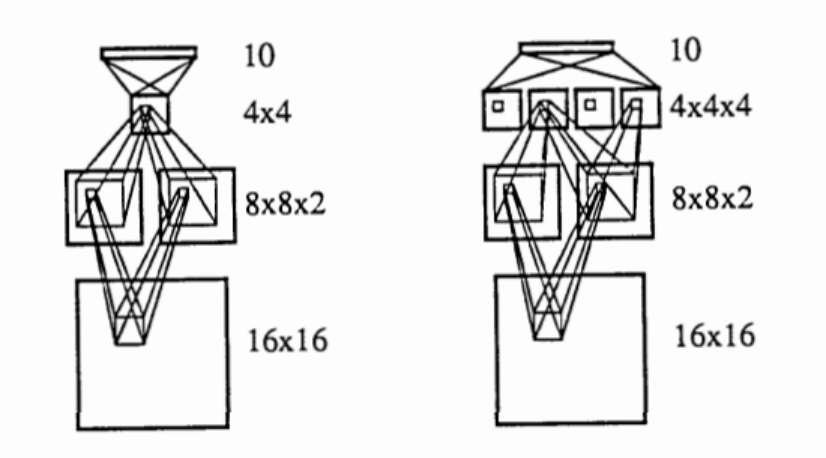


Figure 8: The Constrained network architecture and the Constrained Network 2 architecture presented in the paper [LeC89]

network architecture	links	weights	performance
single layer network	2570	2570	80 %
two layer network	3240	3240	87 %
locally connected	1226	1226	88.5 %
constrained network	2266	1132	94 %
constrained network 2	5194	1060	98.4 %

Table 1 Generalization performance for 5 network architectures. Net-1. single layer; Net-2: 12 hidden units fully connected; Net-3 2 hidden layers locally connected; Net-4: 2 hidden layers, locally connected with constraints; Net-5· 2 hidden layers, local connections, two levels of constraints. Performance on training set is 100% for all networks

Figure 9: Results from the different architectures employed in the paper [LeC89]

The results of the paper, shown in figure 8, display the clear superior performance of the convolutional architecture. The difference in performance between the two contrained networks highlights how deepness of the net, in relation to the amount of kernels employed, seems to be an important design element of the architecture with preference for deeper networks with more kernels. Interestingly enough, contrary to fully connected NNs, the research points out that performances and the amount of built-in knowledge goes up as the number of free parameters in the network goes down. [LeC89]

In a successive paper LeCun et al. consolidate the conclusion that a fully connected network with enough discriminative power for a complex task such as image recognition would have too many parameters to be able to generalize correctly [CBD+90]. The solution to the problem of creating feature detectors that are able to perform shape recognition and combine local features is to scan the input image with a single neuron that has a local receptive field, and store the states of this neuron in corresponding locations in an output called a feature map. This operation is equivalent to a convolution with a small size kernel, followed by a squashing function; it will be necessary to have multiple feature maps, extracting different features from the same image [CBD+90]. The architecture presented in the paper introduces the pattern of the sequential pairing of a convolution layer followed by a pooling

layer; the CNN presents 4 hidden layers of which H1 and H3 are convolutional layers and H2 and H4 are average pooling layers composed by 4 non-overlapping sections of size 4x4 which perform a sort of sampling of the feature space. The average pooling introduce a certain level of invariance to distortions and translations leading to an interestingly useful less precise coding of the location of the features [CBD+90].

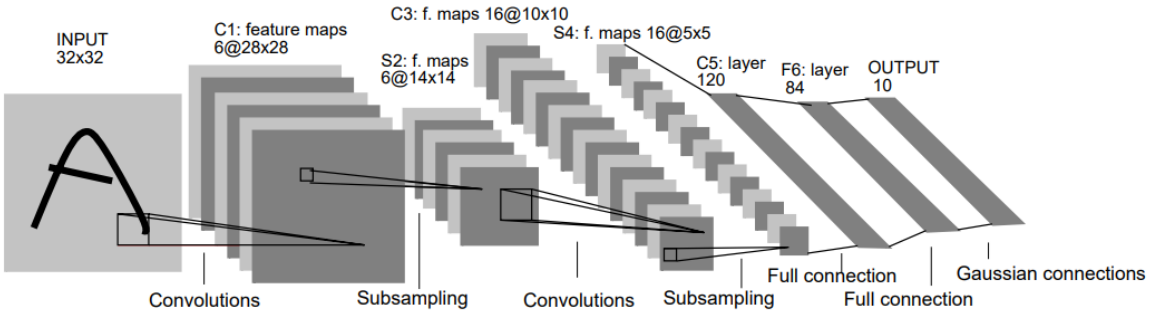


Figure 10: Architecture of the Convolutional Neural Network presented in [LBBH98]

LeCun et al's research culminates in the paper "Gradient-Based Learning Applied to Document Recognition" (1998) [LBBH98], various state of the art methods of character recognition are reviewed and the LeNet-5 architecture is presented, the results of the comparison between the various methods show the CNN outperforming all the others.

The outlined architecture, illustrated in figure 14, presents the convolution-pooling design pattern. Feature map creation through a convolutional layer followed by reduction of the spatial resolution of the map has the objective of reducing the precision with position distinctive features are encoded lowering the sensitivity of the output to shifts and distortions; as a side effect it also reduces importantly the amount of parameters, e.g. the first pooling layer ap1 that reduces halves the pixels of the feature maps from c1.

Interestingly the feature maps in c2 are only connected to a subset each of the feature maps coming from ap2, two are the reasons cited, to reduce the amount of parameters and to force the feature maps to extract different features by getting different sets of inputs; the first six c3 feature maps take inputs only from contiguous subsets of three of maps coming from ap1, the next six from subsets of 4 and the last takes input from the entirety.

Layer c3 is a convolutional layer with 120 feature maps, each unit of which are connected to a 5x5 neighborhood of the ap2 layer, since the size of the feature maps of ap2 is also 5x5 the output is 1x1, the result is that c3 is basically a fully connected layer.

The choice of 84 parameters for the fully connected layer fc4 is based on the Gaussian connection that connects it to the output vector. Traditionally the image recognition process was composed of a feature extractor and a trainable classifier; this feature extractor was designed by the researches through their observation of the examples. The Gaussian Connection choice takes a page from these manually designed feature extractors, in fact, it essentially is a calculation of the Euclidean Radial Basis function with a bunch of artificially fixed weights. The output value from the fully connected layer will be subtracted by these fixed weights, and the square of the difference will be summed up, this calculation won't be plugged into any activation function, instead it will be the actual output vector of the network. The weights set for the Gaussian Connection in LeNet-5 are pixel matrixes that mimic each Arabic numeral, as it can be seen in figure 11. The output values y_i are calculated as follows:

$$y_i = \sum_j (x_j - w_{ij})^2 \tag{2}$$

where $x_j j = 0, 1, 2, ..., 83$ are the 84 output values of the fully connected layer that can be reshaped in a matrix with 12 rows and 7 columns, $w_{ij} i = 0, 1, 2, ..., 9 j = 0, 1, 2, ..., 83$ are the artificially designed fixed weights that can be shaped as a 12x7 matrix and are basically pixel matrixes that mimic the Arabic numerals. The outcome can be interpreted as vector of penalty terms measuring the fit between the i-th input pattern and the manually associated pattern for the i-th class.

Figure 11: Fixed weight patterns (reshaped as a 12x7 matrix) of the Gaussian connection between the fully connected layer and the output vector [LBBH98]

This approach is clearly constrained by the filter values not being dynamically learned, but manually set, and therefore making it not re-usable in a different context, it also requires researchers to gain experience by experimenting on the specific sample images in order to ensure good performances; it has therefore been abandoned in current times.

A major breakthrough in architecture design for Convolutional Neural Networks was the research by Alex Krizhevsky et al. that led to AlexNet, an architecture that was able to exploit the great advancements in both hardware and data availability; the net won the ImageNet Large Scale Visual Recognition Challenge 2012, a competition on a dataset composed by 1000 real life images of 1000 classes, by a large margin. The big advancement of the architecture was that for the first time it demonstrated that features obtained by learning, instead of manually designed ones, could lead to a good performance, advancing from the before employed paradigm of setting the kernel values manually.

The architectures of AlexNet and LeNet are remarkably similar, the significative difference is that AlexNet is much deeper, uses the ReLU activation function and employs dropout layers as it can be seen in figure 15.

Given the much bigger image in output (227x227x3) the first convolutional layer c1 has a much bigger 11x11 kernel compared to that of LeNet, in order to have a bigger window needed to capture objects in the photos. AlexNet has additional convolutional layers and the effective number of feature maps is ten times that of LeNet, followed by 2 huge fully connected layers; these computationally heavy design choices are enabled by the big hardware leaps of the times such as GPUs. The enourmous complexity of the fully connected layers that generates a very big amount of parameters to train is kept down by the two 0.5 dropout layers.

An improvement on the computation side is gained by the choice of the ReLU activation function compared to that of Tanh, ReLU is far simpler since it doesn't have any exponentiation operation and makes training more reliable in situations in which the gradient of the sigmoid could take values of zero.

AlexNet, and its huge amount of parameters, has been eventually surpassed by more modern architectures but represents a key step moving from shallow networks to the deep networks used nowadays.

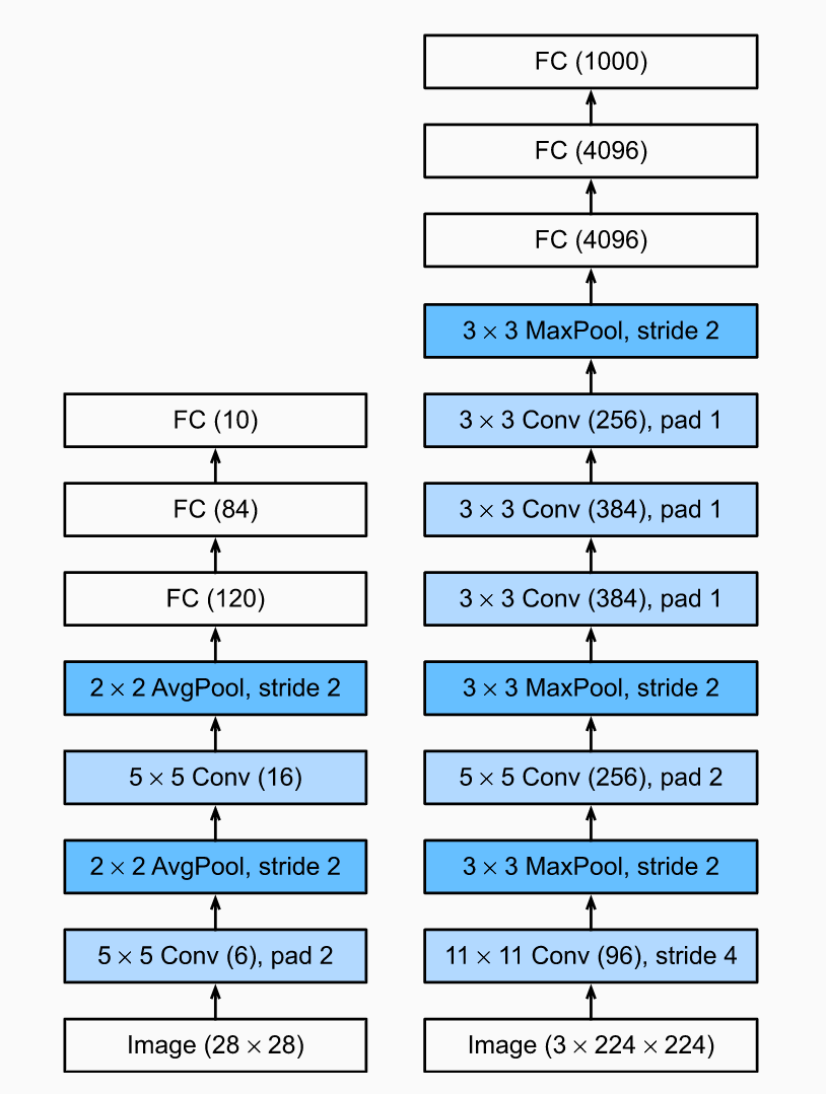


Figure 12: Architecture of LeNet (left) compared to AlexNet (right) [ZLLS21]

Later progress in design principles led researchers to thinking more in terms of blocks of layers than single layers; this idea of using blocks composed by a convolutional layer, the ReLU activation and maxpooling layer originated from the VGG group at Oxford. One of the explored concepts was understanding if multiple convolutions before downsampling could offer an advantage over using bigger kernels. It was shown by Simonyan and Zisserman that deep narrow CNNs would outperform shallow architectures [SZ14].

The VGG family of architectures are CNNs composed by sequential VGG blocks which consist of a sequence of convolutions with 3x3 kernels and padding of 1 (to keep height and width) followed by a 2x2 maxpooling later with a stride of 2 (which halves the height and width after each block), a diagram of the architecture in figure 13.

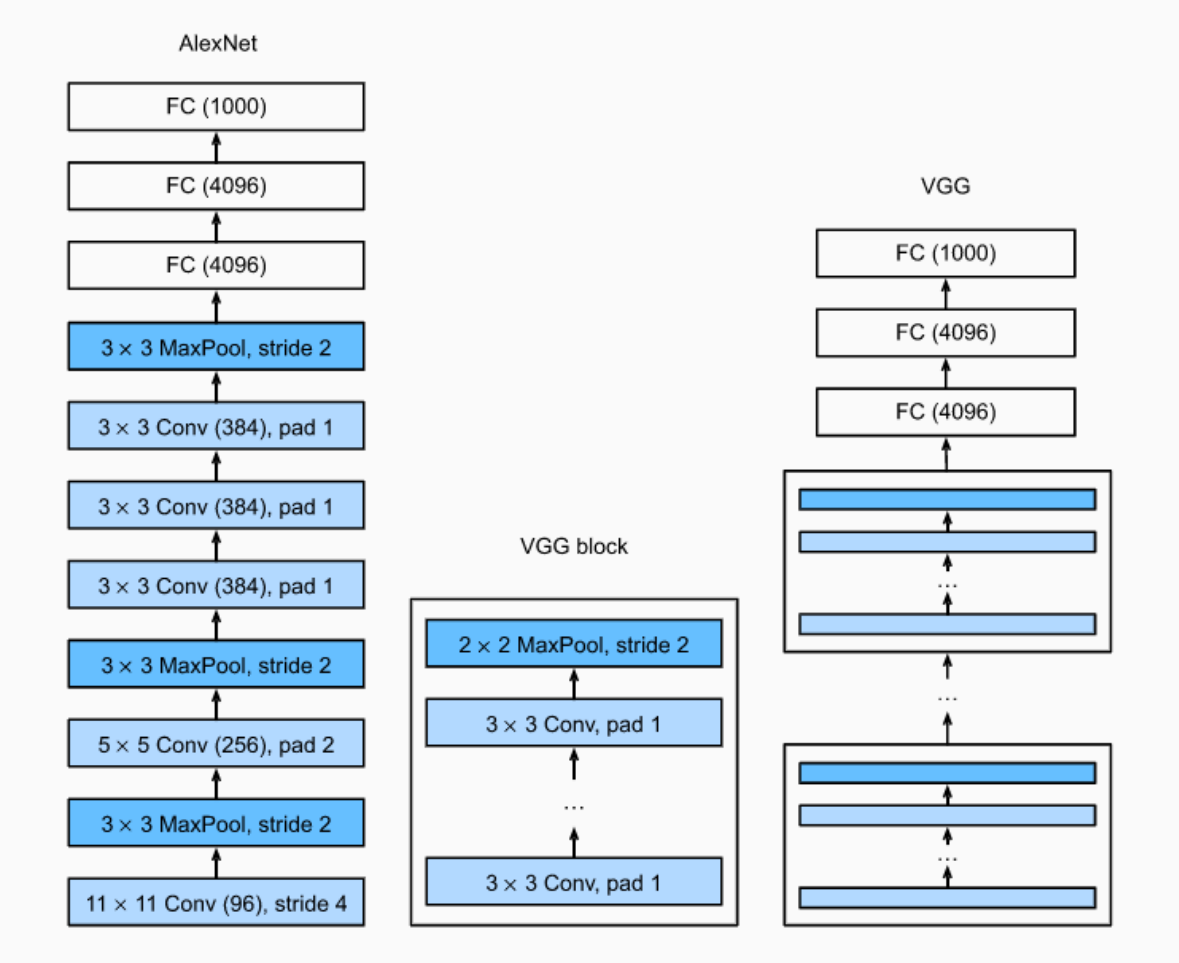


Figure 13: Architecture of AlexNet compared to VGG Networks [ZLLS21]

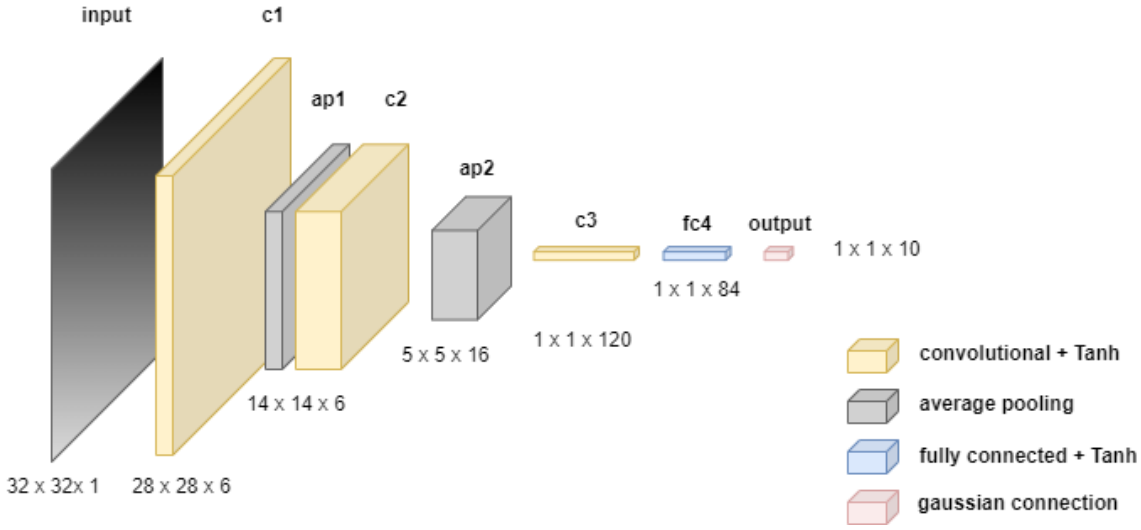


Figure 14: Diagram of the Lenet-5 architecture presented in [LBBH98]

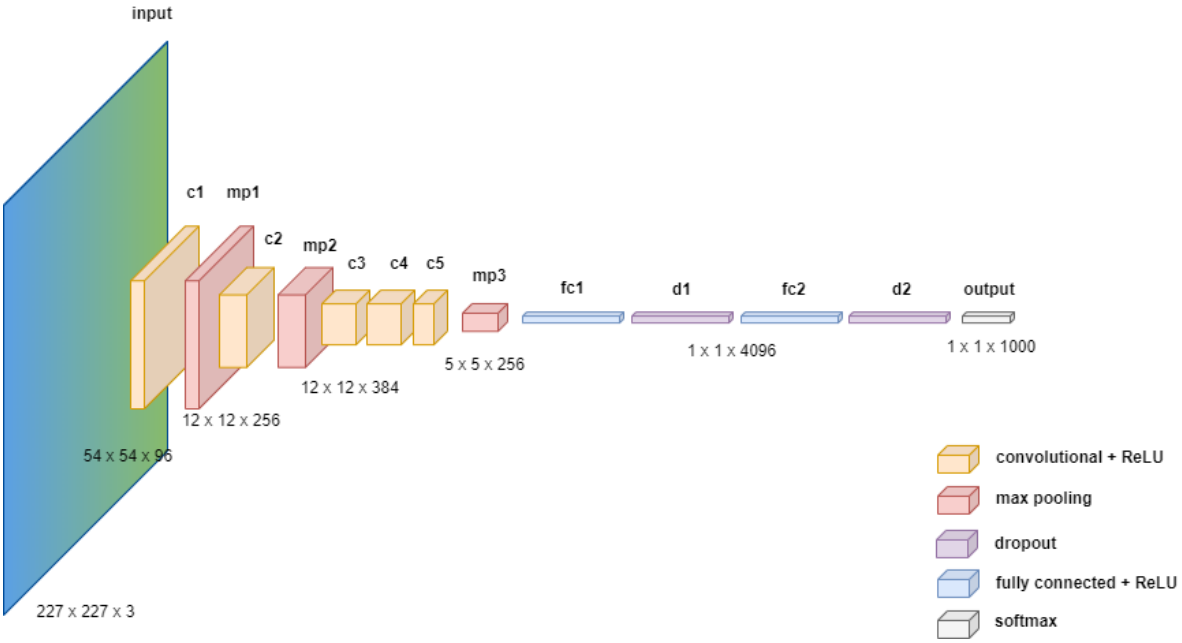


Figure 15: Diagram of the Alexnet architecture presented in [KSH12]

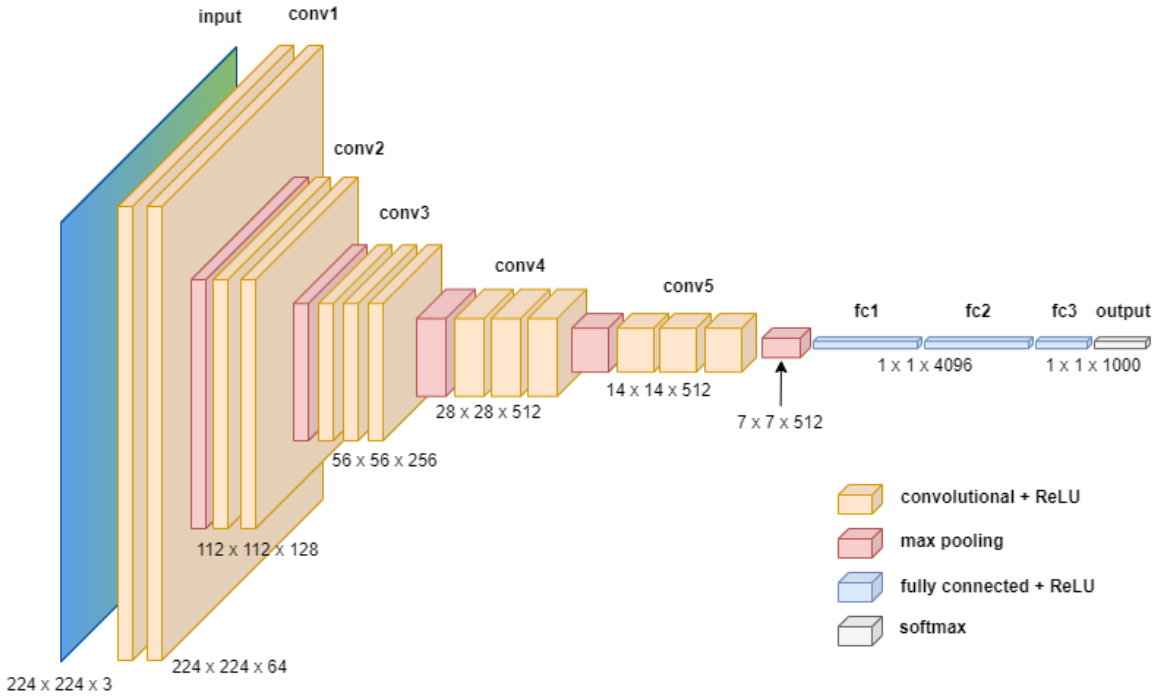


Figure 16: Diagram of the VGG16 architecture presented in [SZ14]

3 Starting point

3.1 Hardware and Tools.

The work has been carried out on an Ubuntu20.04 virtual machine, accessible via the SSH protocol, provided by the University of Milan's Department of Informatics. The machine features a100 nvidia with 10 GBs of memory.

The Tensorflow framework, and his Tensorboard result interface, was chosen as technology to work with.

3.2 Privacy concerns

The pseudo-anonymized ultrasound images used to train the neural network for the project, originating from the Policlinico of Milan's own server, are real ultrasound outputs from patients of the structure. "Personal data concerning health", as defined by Art.4 and specified by Recital 35 of the GDPR, include all data pertaining to the health status of a data subject which reveal information relating to the past, current or future physical or mental health status of the subject; this definition includes information about the person collected in the course of the provision of health care services such as an examination of a body part. Greater legal care that has to be taken when handling data belonging to these "Special categories of personal data", as defined by Art.35, since they are especially sensitive towards the unique identification of the data subject; for this purpose an NDA has been signed by the members of the team to prevent the unauthorized disclosure of the data subject of the project.

3.3 Description of Data

The dataset was composed by 8693 1024x768 `.jpg` files of ultrasound joint scans from 326 unique patients, of which 137 are from unknown patient source. The class distribution is shown in table 1, images labelled as `other` are scans not belonging to any of the previous classes.

Class	Quantity
Ankle	1795
Elbow	2567
Knee	3056
Other	1275

Table 1: Class distribution between the dataset

The pictures presented some initial inconsistencies due to the variance of the size and position of the ultrasound section and the placing of probe related information on the sides as it can be seen in figure 17. An algorithm based on Morphological Snakes for image segmentation, provided by Marco Colussi, has been used to isolate the ultrasound portion of the screen and preemptively crop out the unnecessary information in order to reduce both weight and noise.

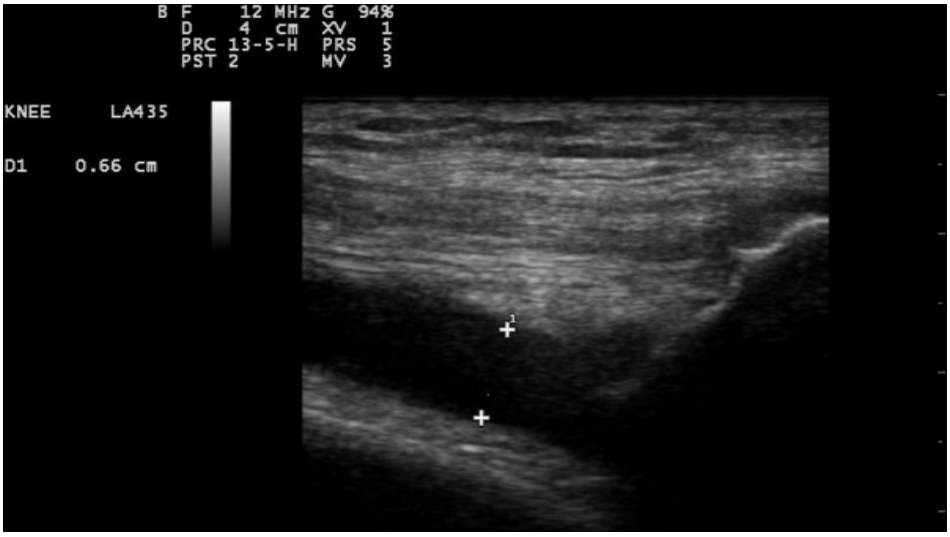


Figure 17: Ultrasound image of a knee [AGP+11]

4 Joint Classification

The first task of the problem consisted of the multiclass classification of the joint image dataset described in section 3.3. The literature review process led to the exploration of a VGG inspired architecture. As illustrated before the main idea of a VGG architecture is stacking consecutive convolutional blocks prior to the pooling layers and followed by the fully connected output block. Figures 18 and 19 display some diagrams of the final presented architecture.

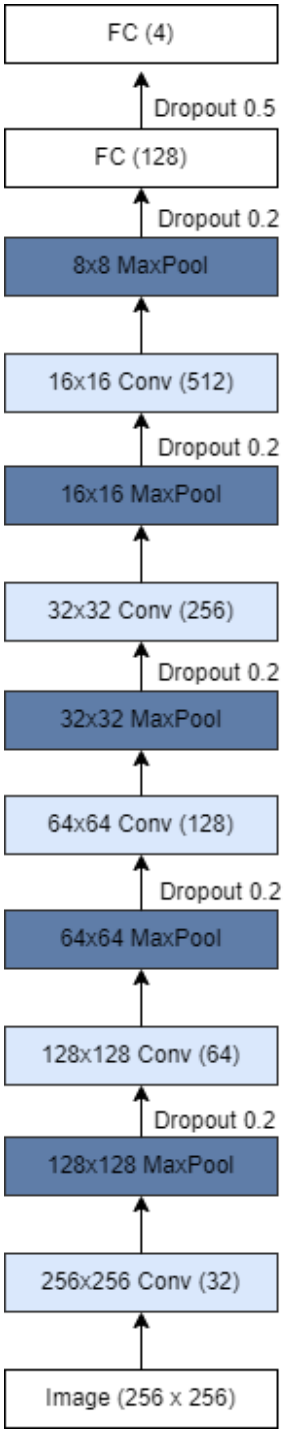


Figure 18: Diagram of the architecture presented for the joint classification problem

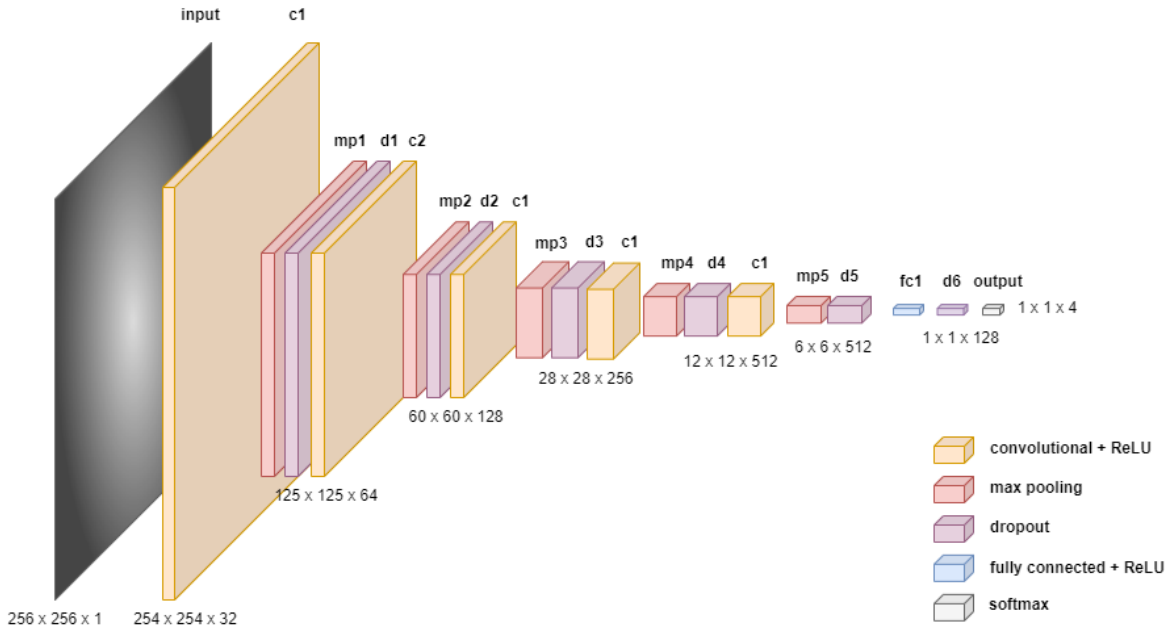


Figure 19: Process of the final architecture presented for the joint classification problem

4.1 Architecture design exploration

An architecture based parameter GridSearch was not time-feasible so several possible decisions have been explored singularly.

The first parameter that has been explored is the number of total parameters of the net. The relationship between the amount of convolutional blocks to number of parameters has been examined and is shown in table 2.

The number of trainable parameters is determined by 2 competing factors. More convolutional blocks create more convolution filters to be trained, but the subsequent maxpooling layer reduces the number of pixels in those filters. Initially, the pooling operation reduces the trainable faster than are added by increasing the activation map number. The formed relationship is a convex parabola and the lowest number of trainable parameters is achieved at 5 blocks.

Having 1 or 2 convolutional blocks limits the network's ability to extract space invariant features since the resulting final activation maps would be very big compared to the initial image. Furthermore, there is a virtual roof of number of parameters which would lead to overfitting or to feature maps being too small to contain actual information if pooled too much.

The machine was also not able to handle the large number of trainable parameters at either ends of the number of convolutional blocks. So based on both points mentioned, the impact of 3, 4, 5, and 6 convolutional blocks has been explored on a train test split and can be seen in figure 20. 5 convolutional blocks have been chosen given the observed very similar performance by architectures with an higher number of parameters.

Conv. blocks	Parameters
1	67 109 828
2	33 573 892
3	16 870 532
4	8 777 092
5	5 762 948
6	8 385 412
7	26 213 252
8	101 190 532

Table 2: Number of trainable parameters compared to number of convolutional blocks featured in the architecture

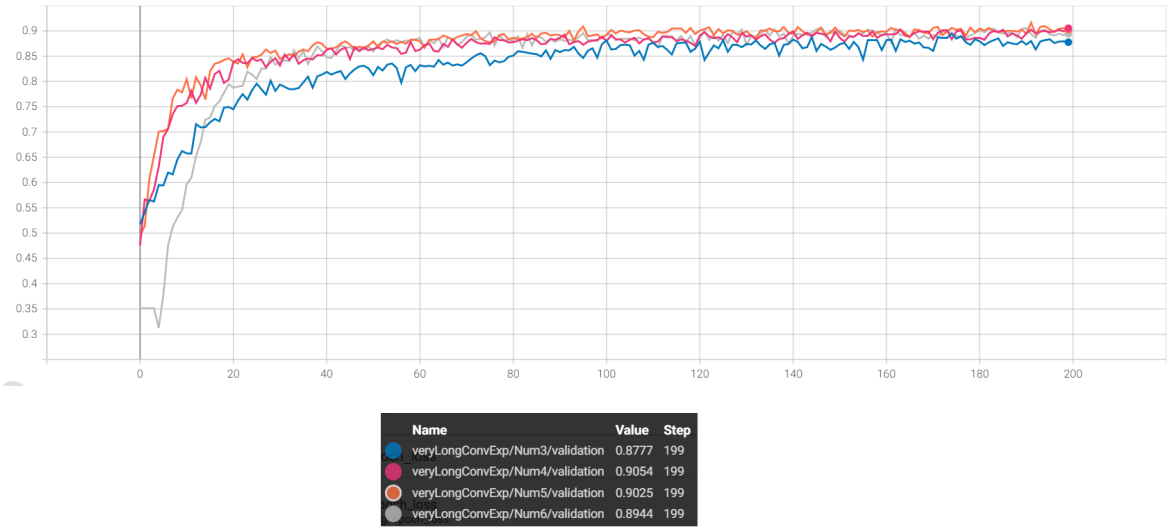


Figure 20: Validation performance of the net on different number of VGG-style convolutional blocks (the number in the name is considered to the the number of additional blocks after the first one, that is the number +1 is the correct amount of convolutional blocks

Another design choice that has been explored is the choice to include dropout layers in the architecture. The dropout layers, present in the VGG architecture, are a very easy way to regularize the network. In figure 21 the performance of the net without dropouts has been shown; a slight reduction in overfitting is achieved by inserting the dropout layers, the difference between train and validation is lower in the architecture featuring them.

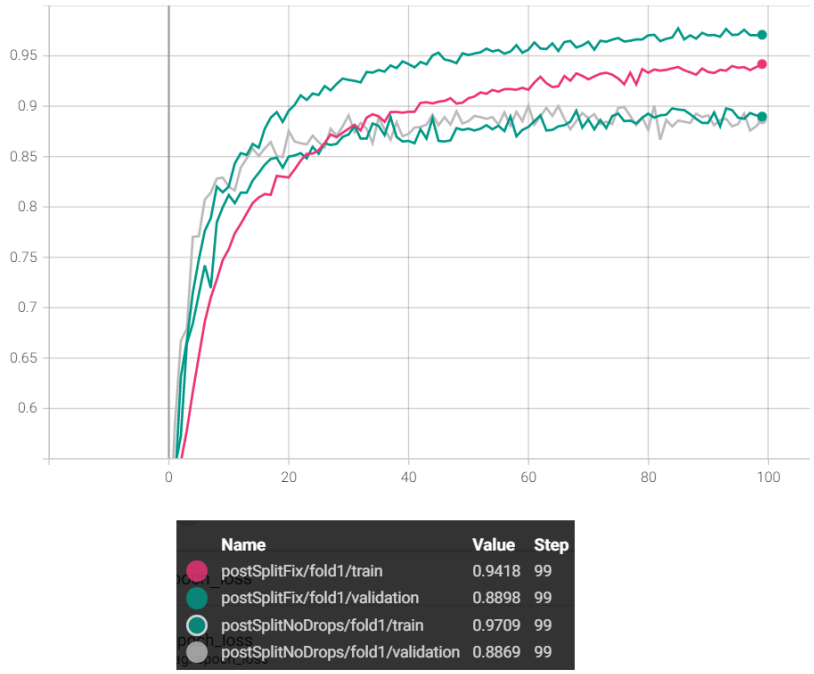


Figure 21: Train/val performances of the architecture with dropout layers, compared to without

The number of nodes in the dense layer has also been explored. A Wider 256 nodes layer, compared to a 128 would mean that more information from the extracted features would be learned by the net. Similar performance is shown by the architectures in figure 22 so the smaller layer is kept.

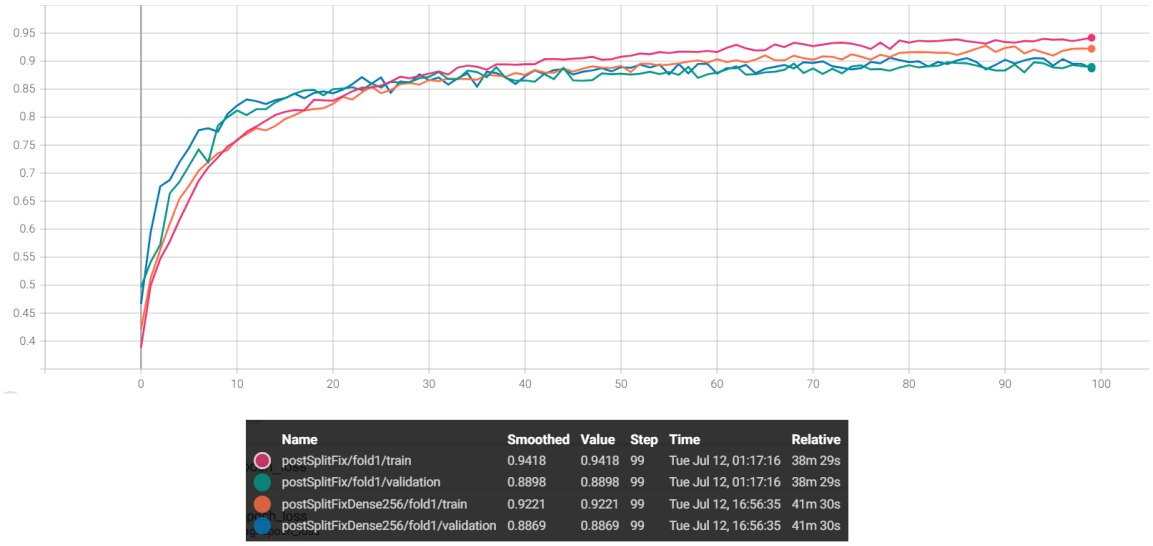


Figure 22: Train/val performances of the architecture with fully connected layers composed by 256 units, compared to 128

A final study has been conducted on the comparison between the performance of our custom designed architecture trained on the problem compared to a transfer learning approach. VGG16 trained on ImageNet has been chosen as a donor for the convolutional feature extracting block. The use of a transfer learning approach streamlines the process quite a lot given that no training is required from the biggest part of the architecture which is the feature extractor; it also minimizes the design

study time requirements given that an already functional architecture is almost entirely reused. As it can be seen in figure 23 considerable overfitting can be observed in the difference between train and validation performance in the transferred can be observed.

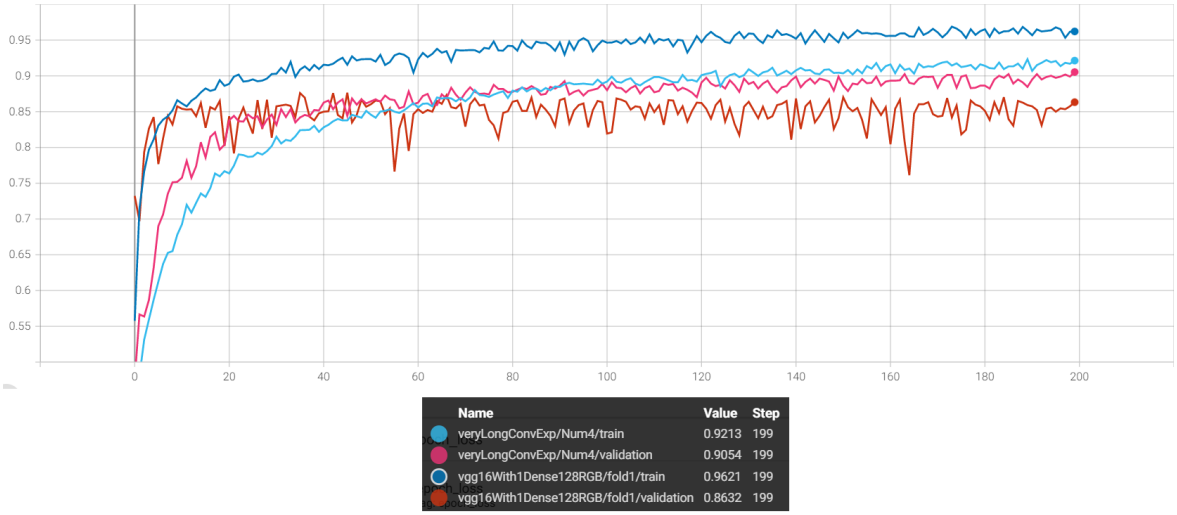


Figure 23: Train/val performances of the CNN compared to transfer learning from VGG16

4.2 Cross-Validated performance

A group-aware CV process had to be employed in order not to violate the independence assumption between testing sets and training sets.

The set of images S is composed by m pairwise disjoint subsets which represent groups of images originating from the same patient. In order not to violate the independence assumption S has been divided in 5 roughly equal-sized subsets S_i such that $\forall j = 1, ..., m$:

$$G_j \cap S_i \neq \emptyset \implies G_j \cap S_k = \emptyset \quad \forall k \neq i \tag{3}$$

Then 4 of the S_i subsets have been chosen as training/validation split, subsequently split by the same process, and 1 as a test set iterating the operation for 5 times, rotating them.

Results from the 5 fold cross validation run implemented are in the tables 3, 4 and 5 that follow.

	Ankle	Elbow	Knee	Other
Ankle	1651	29	36	79
Elbow	30	2425	46	66
Knee	59	72	2876	49
Other	158	179	99	839

Table 3: Confusion matrix for the Cross validation run

	Loss	Accuracy
1st Fold	0.347	0.908
2nd Fold	0.534	0.886
3th Fold	0.438	0.883
4th Fold	0.470	0.885
5th Fold	0.374	0.915

Table 4: Test loss and accuracy between folds for the Cross validation run

Accuracy	Train	Validation
1st Fold	0.941	0.889
2nd Fold	0.947	0.888
3th Fold	0.942	0.916
4th Fold	0.948	0.891
5th Fold	0.939	0.884

Table 5: Train and validation accuracy between folds for the Cross validation run

5 Knee Side Classification

An additional development of the project has been to further attempt to classify knee ultrasound images based on the angle the ultrasound was taken from. During the medical examination the doctor places the probe on the different sides of the knee; figure 26 shows the possible ultrasound results for the different angles which are denominated as SQR (A), Lateral (B black), Medial (B white) and Femoral (C).

The class distribution between the different angles present in the 2793 knee ultrasounds in the database is outlined in table 6.

Class	Quantity
Femoral	464
Lateral	654
Medial	1009
SQR	484

Table 6: Class distribution of knee ultrasound angles

Different approaches to this classification problem have been compared. The CNN architecture design used for the joint classification has been fully trained on the 2793 images; a transfer learning approach using the already trained feature extractor from the CNN used to solve the preceding task has also been employed, figure 24 showcases this architecture. The results of the training process shown in figure 25 show equivalent performances for the two techniques. As per the needed training time the transfer learning approach has a big upside. Unsurprisingly the validation performance of the transfer learning system takes more steps to converge to the performance of the full trained CNN but is considerably faster time-wise.

The performances on a group-aware (the split methodology for the joint classification) test set composed by 20% of the dataset, are shown in table 9. The resulting confusion matrixes in figure 8 and 8 are also quite similar.

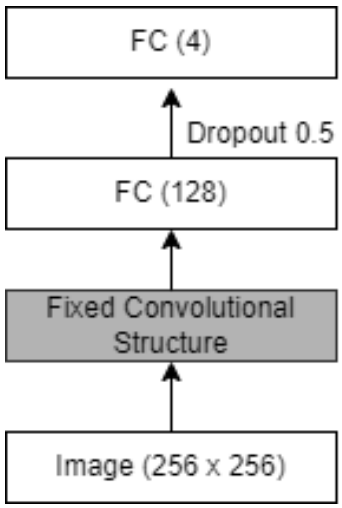


Figure 24: Architecture diagram of the transfer learning operation, the fixed convolutional structure is that of the CNN trained for the joint classification problem

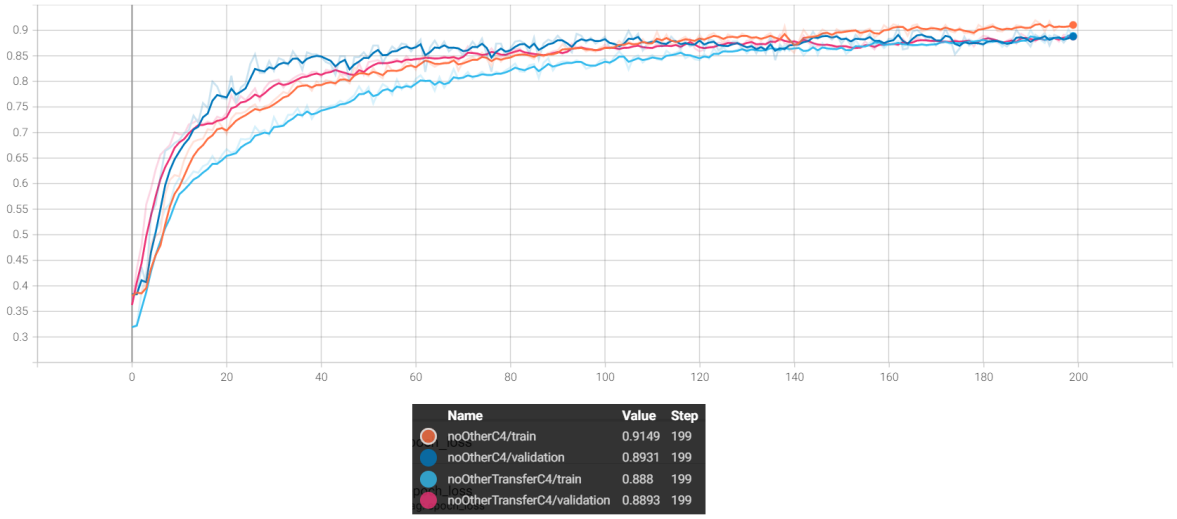


Figure 25: Log of the full training run and of the transfer learning system

	Femoral	Lateral	Medial	SQR
Femoral	90	1	1	0
Lateral	0	112	14	5
Medial	0	17	183	3
SQR	0	1	0	97

Table 7: Confusion matrix for the joint classification architecture design trained on the Knee dataset

	Femoral	Lateral	Medial	SQR
Femoral	89	3	0	0
Lateral	1	107	20	3
Medial	0	20	180	3
SQR	3	2	0	93

Table 8: Confusion matrix from the transfer learning operation on the Joint classification problem CNN

	Loss	Accuracy
Trained Cnn	0.272	0.919
Transfer Cnn	0.298	0.895

Table 9: Test loss and accuracy between folds for the Cross validation run

An additional possible approach to the classification is using the unsupervised 4-means clustering algorithm on the flattened feature vector output from the convolutional feature extractor of the CNN. The clustering results, figure 10, show a loss of granularity in the ability of the system to detect the difference between Lateral and Medial scans. These two categories of scans, as shown in figure 26, are extremely similar in its features. This empirical conclusion argues that the pattern learning capability offered by the fully connected structure of a neural network is superior to that of a simple clustering algorithm.

	Cluster 0	Cluster 1	Cluster 2	Cluster 3
Femoral	0	26	438	0
Lateral	263	387	2	2
Medial	510	498	0	1
SQR	0	27	0	457

Table 10: Unsupervised clustering on the feature vector extracted by the Convolutional feature extractor from the CNN

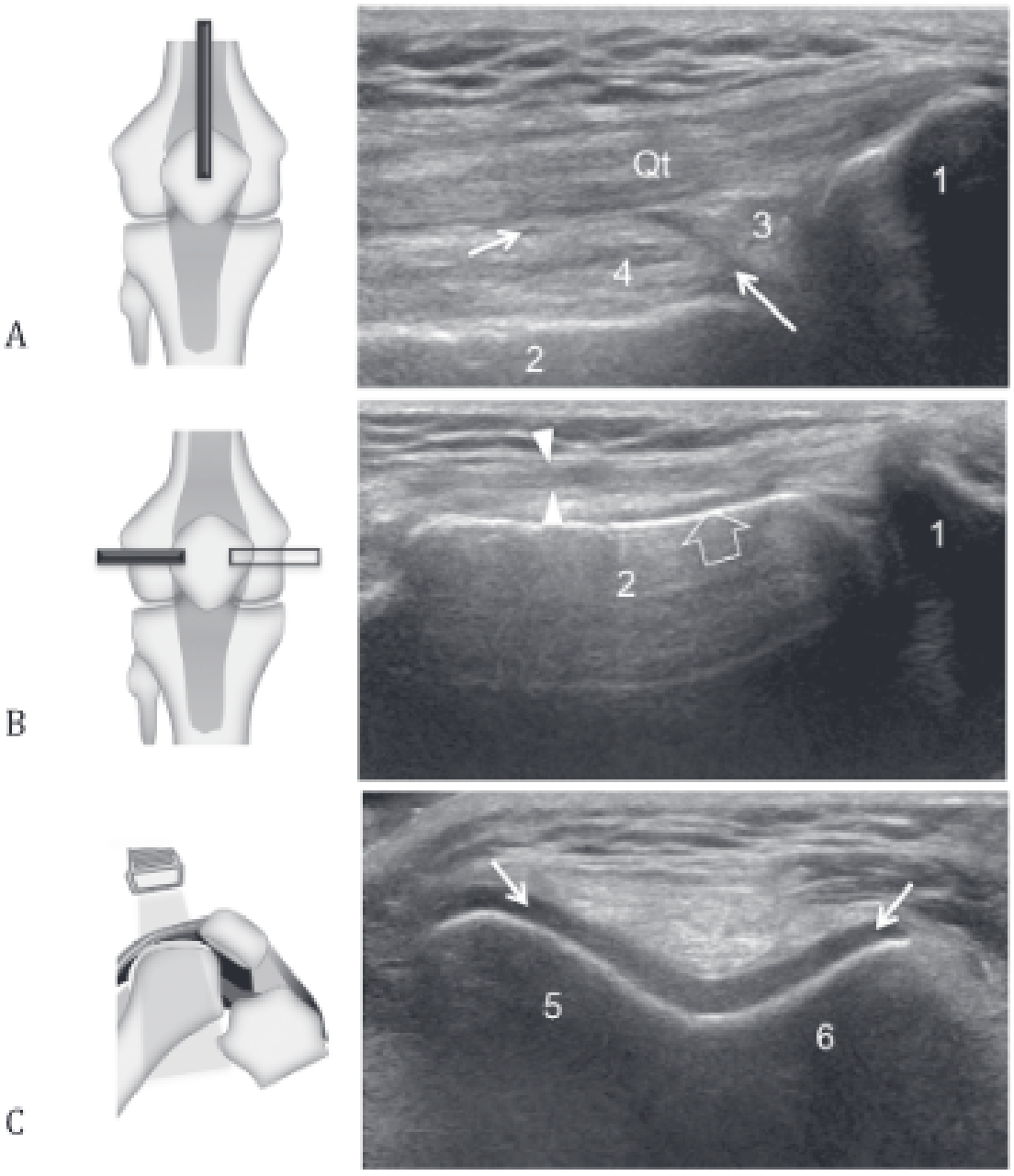


Figure 26: Illustration of the different angles depending on the position of the probe during the examination. SQR (A) scans are collected from the top of the patella, Femoral (C) are collected with the probe rotated by 90 degrees. Medial (B white) and Lateral (B black) are collected on the right and left of the patella, with the probe positioned at the same orientation, the resulting images are very similar between them.

A successive study carried out on the knee dataset focuses on the number of images made available to the network to train on and its behaviour in relation. The training process has been repeated multiple times with subsets of smaller size from the training set, 80% of it, 60%, 40% , 20% and 10% of it. As the number of available training points descends the overfitting pattern of high training, but lower validation performance, ensues, as well as the validation performance worsening progressively.

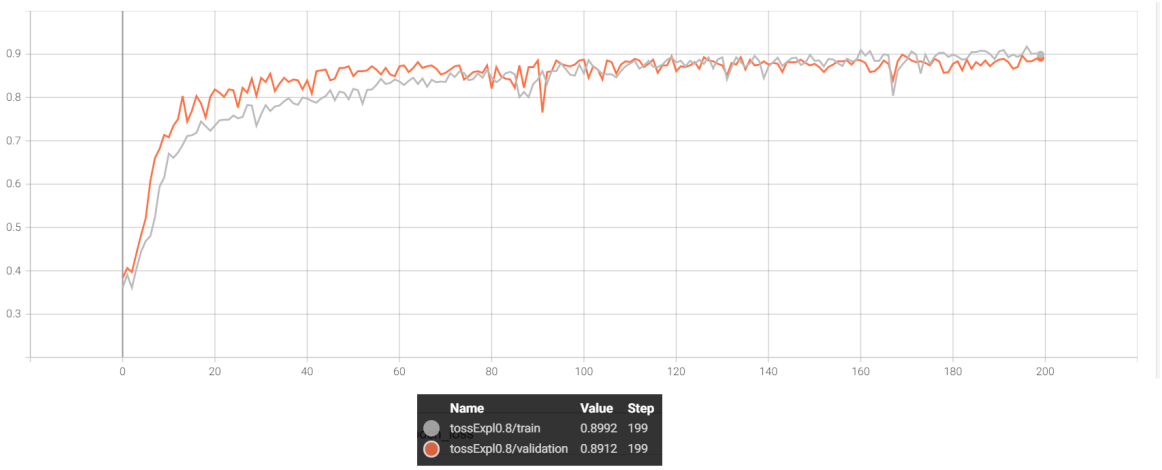


Figure 27: Training run on 80% of the initial training set

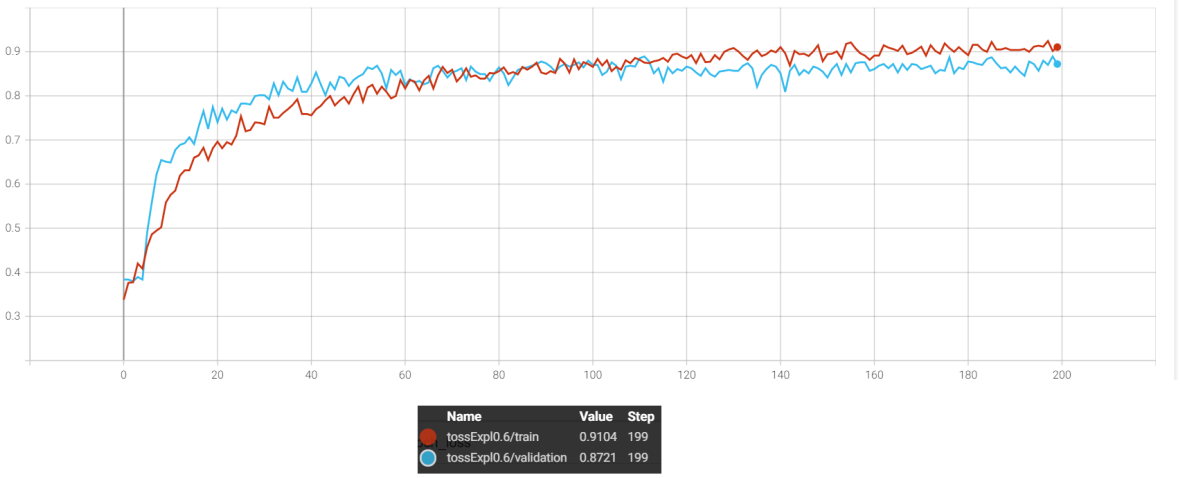


Figure 28: Training run on 60% of the initial training set

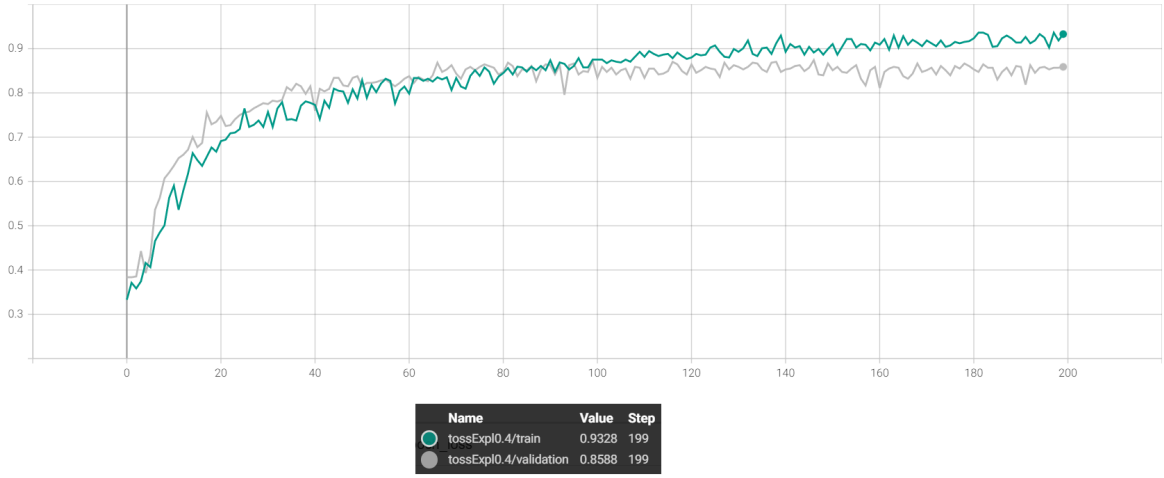


Figure 29: Training run on 40% of the initial training set

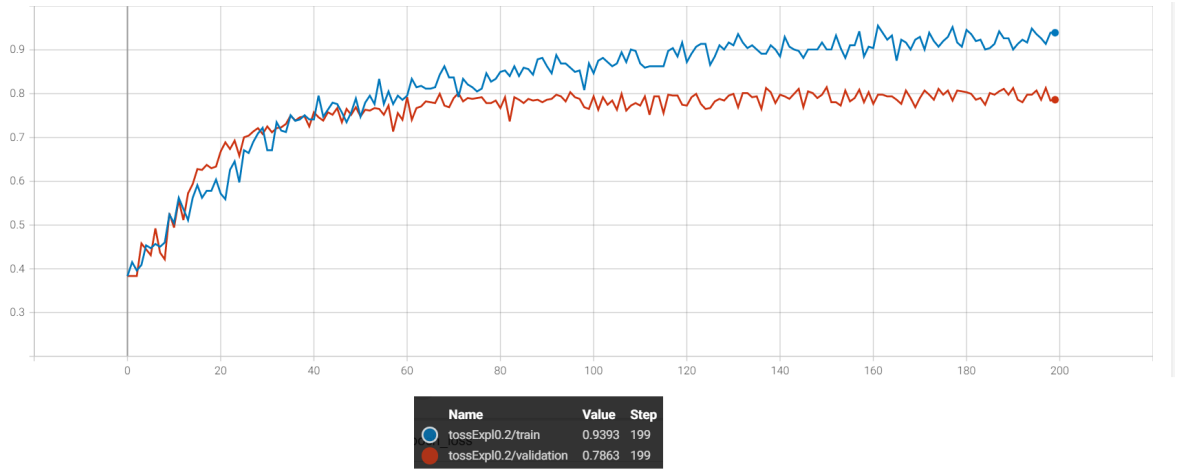


Figure 30: Training run on 20% of the initial training set

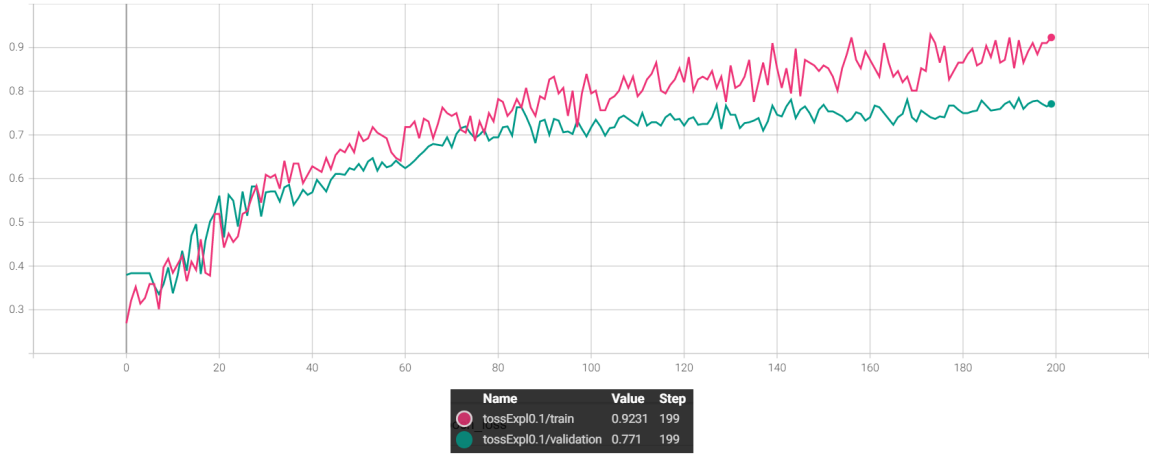


Figure 31: Training run on 10% of the initial training set

We declare that this material, which we now submit for assessment, is entirely our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of our work. We understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by us or any other person for assessment on this or any other course of study.

References

[AGP+11] Ajay Abraham, Iain Goff, Mark Pearce, Roger Francis, and Fraser Birrell. Reliability and validity of ultrasound imaging of features of knee osteoarthritis in the community. *BMC musculoskeletal disorders*, 12:70, 04 2011.

[Ara20] Issar Arab. *Variational Inference to Learn Representations for Protein Evolutionary Information*. PhD thesis, 09 2020.

[CBD+90] Y. Le Cun, B. Boser, J. S. Denker, R. E. Howard, W. Habbard, L. D. Jackel, and D. Henderson. *Handwritten Digit Recognition with a Back-Propagation Network*, page 396–404. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86, pages 2278–2324, 1998.

[LBD+89] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[LeC89] Y. LeCun. Generalization and network design strategies. In R. Pfeifer, Z. Schreter, F. Fogelman, and L. Steels, editors, *Connectionism in Perspective*, Zurich, Switzerland, 1989. Elsevier. an extended version was published as a technical report of the University of Toronto.

[MW21] Sakib Mostafa and Fang-Xiang Wu. Chapter 3 - diagnosis of autism spectrum disorder with convolutional autoencoder and structural mri images. In Ayman S. El-Baz and Jasjit S. Suri, editors, *Neural Engineering Techniques for Autism Spectrum Disorder*, pages 23–38. Academic Press, 2021.

[ON15] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *ArXiv e-prints*, 11 2015.

[RP19] Shuvendu Roy and Sneha Paul. Land-use detection using residual convolutional neural network. pages 1–6, 05 2019.

[SZ14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[ZLLS21] Aston Zhang, Zachary Lipton, Mu Li, and Alexander Smola. Dive into deep learning, 06 2021.