Convolutional Neural Networks for the Classification of 3D Medical Images

Luís Henrique Vieira Pereira
Instituto Superior Técnico, Lisboa, Portugal

December 2020

Abstract

Medical imaging is a fundamental screening and diagnostic tool. Healthcare professionals can nowadays rely on various types of image modalities of the human body, including three-dimensional images such as magnetic resonance images. However, with increasingly more information and image complexity, the pressure that radiologists are subjected to is ever increasing, and the resulting fatigue can lead to diagnostic errors. Machine learning mechanisms have been proposed for the analysis of medical images, although most previous work has dealt with inputs involving two dimensions. This work proposes an approach based on convolutional neural networks, combined with recurrent neural networks, for the classification of three-dimensional medical images. The proposed architecture aims to extract features from the individual slices of the three-dimensional image, using a convolutional network, and correlate them with the three-dimensional nature of the original images using a recurrent neuronal network. Experiments were carried out with different architectures to classify three-dimensional images of the knee, leveraging a publicly available data-set. The results show that the main model presented in this work, based on the ResNet architecture and LSTM units, is efficient for the classification of this type of images, despite being relatively simple.

Keywords: Artificial Intelligence, 3D Image Classification, Deep Learning, Convolutional Neural Networks, Recurrent Neural Networks

1. Introduction

Medical imaging is a fundamental screening and diagnosis tool with spread use in modern medicine. Common types of medical imaging include computed tomography (CT), positron emission tomography (PET), and magnetic resonance imaging (MRI). These scans give detailed three-dimensional (3D) images of human organs and can be used to detect infection, cancers, traumatic injuries and abnormalities in blood vessels and organs. With the advances of modern medicine and imaging, the modalities of medical imaging that use more than a single image for diagnosis, have started to become more readily available, with better image quality. That in turn grows the databases of medical images and with the advent of high performance computers, machine learning methods can now play a crucial role in assisting clinicians in the analysis of medical images, providing numerous benefits, from improving workflow to supporting clinical decisions.

However, in medical data that comprises multidimensional and multi-planar images, traditional approaches often fail due to the complexity of the data being handled. As a consequence, implementations related to the analysis of three dimensional medical images, e.g. MR imaging, remain not well explored, though it must be said that in recent years work in this domain, i.e. deep learning publications on the analysis 3D images, has been growing rapidly.

Existing approaches based on 3D deep neural networks have indeed been tried, and successfully gained on traditional image analysis methods, enabling significant progress in medical imaging tasks [1, 2]. Most of these methods rely on convolutional neural networks (CNNs), extended to 3D data, which have proven to be dependable on the classification and segmentation of two dimensional (2D) images. More recently, frameworks that combine CNNs and recurrent neural networks (RNNs) have been pushed forward [3], looking at the classification problem from a different perspective. This article intends to expand the previously developed methodologies, by studying novel deep learning methods for 3D image classification, implementing a deep CNN combined with a RNN for the classification of 3D medical images, and evaluating the proposed method on an established standard benchmark dataset, namely MRNet [4].

The rest of this paper is organized as follows.

Section 2 presents fundamental the concepts on machine learning applied to image classification, together with related work on 3D image classification. Section 3 presents the proposed methodology while Section 4 details the evaluation procedure and the obtained results, comparing them to other methods using the same dataset. Finally, Section 5 summarizes our conclusions and presents directions for future work.

2. Concepts and Related Work

This section is divided into two other subsections. Section 2.1 reports fundamental concepts on machine learning that are relevant for this work, such as Convolutional Neural Networks (CNNs). Section 2.2 presents previous studies conducted in the field of 3D medical image classification.

2.1. Deep Learning Concepts

Neural networks are, at the most basic level, composed of perceptrons units which are connect to each other in layers, in order to map inputs into targeted outputs. These layers can be seen as nested functions whose parameters can be trained directly to minimize a given loss function computed over the outputs and the expected results.

In its simplest form, a single-node neural network computes a single output from multiple real-valued inputs by composing a linear combination according to input weights, and then transforming the output through an activation function. In a mathematical form, Equation 1 shows how this can be written, where *y* refers to the output prediction, $\mathbf{x} = (x_1, \cdots, x_n)$ is the vector of inputs, **w** denotes the vector of weights, *b* is a bias term, and $\phi(.)$ is an activation function.

$$y = f(x) = \phi\left(\sum_{i=1}^{n} w_i \times x_i + b\right) = \phi(\mathbf{w} \cdot \mathbf{x} + b) \quad (1)$$

Granting that a single-node neural network has limited mapping ability, the conjunction of several of these nodes into blocks can be used to build a more complex model. A Multi-Layer Perceptron (MLP) builds on this idea, as it consists of a set of nodes forming the input layer, one or more hidden layers of computation nodes, and an output layer of nodes. The information flows from the input layer through the network layer-by-layer, until it reaches the output. MLPs are also typically refered to as Feed-forward networks and in the case of a single hidden layer, MLP can be mathematically interpreted as:

$$y = f(x) = \phi(\mathbf{B} \times \phi'(\mathbf{A} \cdot \mathbf{x} + \mathbf{a}) + \mathbf{b}) \quad (2)$$

In Equation 2, **x** is a vector of inputs and *y* a vector of outputs. The matrix **A** represents the weights of the first layer and **a** is the bias vector of the first layer, while **B** and **b** are, respectively, the weight matrix and the bias vector of the second layer. The functions $\phi'(.)$ and $\phi(.)$ both stand for an element-wise non-linearity, as result of activation functions respectively associated to nodes in the hidden layer, and in the output layer.

A neural network such as MLP is trained by adapting weights and biases to optimal values, so that the intermediate computations, used to define the function, match their optimal parameters. Normally, models are trained using iterative gradient-based optimizers, that lower the cost function. Gradient descent is used to minimize the cost function, by adjusting weights and biases of the network in the opposite direction of the gradient. Adaptive Moment Estimation (Adam) is an algorithm extensively used for this purpose, that calculates adaptive learning rates for each parameter [5]. To apply gradient descent training in a deep neural network the most popular learning technique is back-propagation algorithm, that consists of two steps. In a forward pass the predicted outputs are evaluated, and in a backward pass the error calculated from the predicted outputs in the output layer is propagated backwards throughout the layers, updating the weights of layers responsibility for a portion of the error.

For more complex applications, such as image processing, MLPs have limitations due to the number of parameters associated with images. MLP networks use dense interactions between every input and output unit making their use prohibitive. Convolutional Neural Networks (CNNs) tackle this problem by having the neurons within a layer only connecting to a small region of the layer preceding it, and thus using common parameters to process all these small regions.

In more detail, CNNs are typically comprised of three types of layers. Convolutional layers determine the output of neurons connected to local regions of the input, through the calculation of the scalar product between the layer's weights and the region connected to the input volume, followed by an activation function. The depth of the output produced by a convolutional layer corresponds to a number of filters. Each filter is convolved across the spatial dimensionality of the input (1D, 2D, 3D), producing a feature map. These maps are stacked along the depth dimension to form the full output volume from the convolutional layer. Pooling layers follow convolution layers, and down-sample each feature map independently, reducing both the height and width, while persevering the depth intact. A commonly used type of layer is max-pooling, which returns the maximum value in the pooling window. Finally, the latter layers are combined with fully-connected layers similar to those in MLPs, that seek to emulate the desired outputs and generate the final representations.

Another example of neural networks are Recurrent Neural Networks (RNNs), designed for processing sequential data. RNN architectures take just the current input instance, but also what was perceived one step back in time. This can be useful for processing sequences of images, consisting of multiple slices. More formally, given a sequence $x = (x_1, x_2, \cdots, x_t)$, a standart RNN updates its recurrent hidden state $\mathbf{h}_t$ by sequentially processing the input sequence and computing:

$$\mathbf{h}_t = \phi'(\mathbf{W} \cdot \mathbf{x}_t + \mathbf{U} \cdot \mathbf{h}_{t-1}) \tag{3}$$

In brief, we have that the hidden state $\mathbf{h}_t$ at time step t is a function of the input at the same time step $\mathbf{x}_t$, modified by a weight matrix $\mathbf{W}$. This result is added to the hidden state of the previous time step $\mathbf{h}_{t-1}$, after multiplied by its own hidden-state-to-hidden-state matrix $\mathbf{U}$. Previous studies indicate that traditional RNNs can suffer from vanishing gradients, leading to difficulties in training deep models. A method to deal with this problem is the use of Gated Recurrent Units (GRUs) [6] or Long Short-Term Memory (LSTM) recurrent units, proposed by Hochreiter et al. [7]. This latter approach is detailed in the next section of this paper.

2.2. Deep Learning Methods for 3D Medical Image Analysis Tasks

In a very short time, deep learning techniques have become an alternative to many machine learning algorithms that were traditionally used in medical imaging. With the appearance of larger data sets of labeled medical images, deep learning methods to perform classification or segmentation tasks have achieved performances similar or even better of clinical experts [4, 8]. Recent examples of publicly available data sets, supporting this type of developments, include the MRNet data-set [4] for the classification of MR knee scans.

Previous work on the analysis of two dimensional medical images, such as X-rays, has proved to be successful. For instance Rajpurkar et al. [9], found that an algorithm based on Densely Connected Convolutional Networks (DenseNet) can detect and localize lesions at a comparable rate to radiologists. With decreasing computational costs and more availability of better graphic processing units, three dimensional medical images also became possible targets for for deep learning methods.

A state-of-the-art approach by Korolev et al. [10] constructed a full 3D CNN, trained to classify Alzheimer's Disease (AD) using MRI data from the Alzheimer's Disease Neuroimaging Initiative (ADNI). In their work, some well-known baseline 2D deep architectures, such as VGGNet and ResNet, were converted to their 3D counterparts. However models like these have millions of parameters and are harder to train on smaller data-sets.

A different approach followed by Bien et al. [4] is another example for the use of deep learning on the task of classifying 3-dimensional medical images. The authors proposed the use of a well-know 2D deep architecture, AlexNet [11], to perform feature extraction on each slice. They then combined the vectors obtained to generate an encompassing representation, before finally using a logistic regression model to generate a single prediction for each exam. On the MRnet dataset, the authors managed to obtained a high classification accuracy (namely 85%, 86,7% and 72,5% on abnormal detection, ACL tear detection, and meniscal tear detection respectively) which compared fairly with the clinical expert's accuracy.

Another application of deep learning for evaluating knee MR images was brought forward by Liu et al. [12]. The authors developed a fully automated deep learning–based cartilage lesion detection system by using a joint segmentation and classification convolutional neural network. The proposed model was trained on a small data set of T2-weighted MR knee images and the obtained results were compared to practicing clinicians. The results indicated a high overall diagnostic accuracy for detecting cartilage lesions.

Nokivok et al. [8] reported on the use of a Convolutional Long Short-Term Memory (C-LSTM) network in 3D scans, to address the issues caused by implementing a 3D CNN approach. In brief, the proposed model processes 3D volumetric scans as a time-series of 2D slices, using time distributed convolutions. It then feeds the output of the convolutions onto a bidirectional C-LSTM block, in order to leverage spatio-temporal correlations of the order-preserving slices. The neural network showed competitive and sometimes superior performance on liver and vertebrae segmentation tasks, leaving the authors to plan about future use of this model on other imaging tasks such as classification.

A similar approach to the three dimensional classification solution proposed by in this work, was presented by Liu et al. [3] in 2018. The paper proposed a framework of a conventional CNN and a Gated Recurrent Unit (GRU) to learn and classify Fluorodeoxyglucose Positrons Emission Tomography (FDG-PET) images, sequenced into two dimensional slices. The architecture achieved a good performance on the classifications of alzheimer's disease (AD) and mild cog-

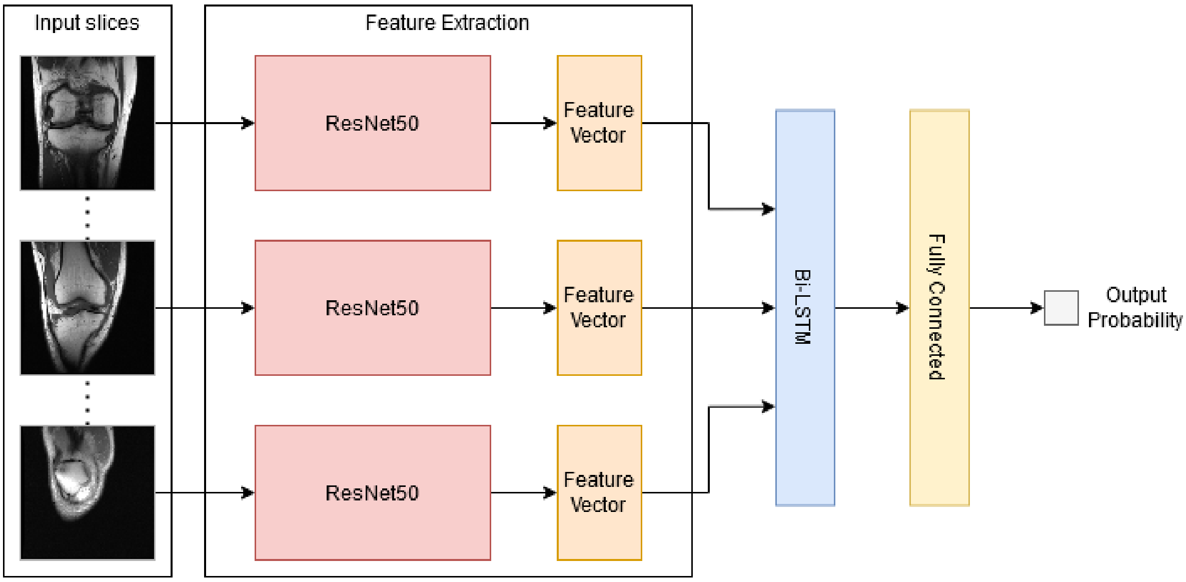

Figure 1: Overview on the proposed CNN-LSTM architecture.

nitive impairment results (MCI), on images of the ADNI dataset.

3. Methodology

This section describes the approach used to address the task of classifying three dimensional multi-planar medical images using deep learning techniques. Using current state-of-the-methods, the model combines the use of convolutional neural networks and recurrent neural networks. Figure 1 illustrates the overview of the approach implemented approach.

Section 3.1 addresses the application of Residual Neural Networks (ResNets) for extracting meaningful features from the images, while Section 3.2 describes in detail the architecture of the complete model with the LSTM component for processing sequence of slices. In Section 3.3, an alternative model is presented, taking the already proposed approach and unifies the single labels into a multi-label neural network. The implementation of the models relied mostly on the Keras[1] deep learning library, using as computational backend TensorFlow2[2]. Resources from other libraries, such as scikit-learn[3] and scikit-image[4], were also used for specific operations.

3.1. Feature Extraction with Residual Neural Networks

With the increase of computing power availability, deep learning architectures started to become more popular as they proved to be a breakthrough in image classification tasks. To accomplish image recognition and classification tasks, deep models often involve stacking multiple convolutional and pooling layers in a network for producing a feature vector, followed by fully-connected layers that produce a final classification. The evolution of deep learning with convolutional neural networks can be seen from the introduction of LeNet by LeCun et al. [13], with seven layers, to more recent approaches as the VGGNet by Simonyan and Zisserman [14], that had as much as 19 layers. This push in depth resulted in improvements for image processing accuracy. However as network depth increases, it was noted that accuracy gets saturated and then degrades rapidly.

The Residual Neural Network (ResNet), introduced by He et al. [15], proposes to address the issue of degradation by applying residual blocks. In more detail, ResNet models are based on deep residual learning. Instead of trying to stack layers to fit a desired mapping $H(x)$ directly from x, these layers are designed deliberately to fit a residual mapping $F(x)$. Formally, the stacked non linear layers are made to fit $F(x) = H(x) - x$, recasting the original mapping into $F(x) + x$.

This method of residual learning is adopted in blocks of few stacked layers, also called residual building blocks. In these building blocks, a stack of layers learns the residual mapping $F(x)$, and the

[1] https://keras.io
[2] https://www.tensorflow.org
[3] https://scikit-learn.org
[4] https://scikit-image.org

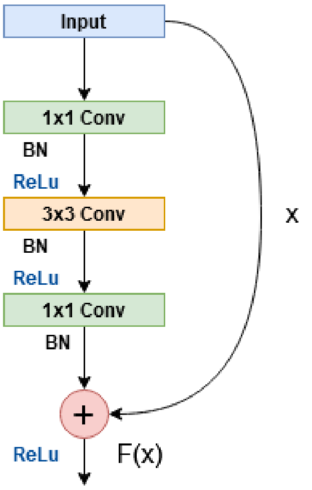

Figure 2: Illustration of a Residual Block.

operation $F + x$ is performed by a shortcut connection, also called identity mapping, together with an element-wise addition. The identity mapping does not introduce extra parameters nor computation complexity and, in cases were the dimensions of x and F are different, a linear projection can be performed by the shortcut connection to match the dimensions. An example of a residual block can be seen in Figure 2, were the stacked layers are three different convolutional layers, and a shortcut connection from the input is added to the output of the last layer in the block, represented as $F(x)$.

The ResNet applied in our work consists of 50 layers, were the residual blocks have a bottleneck design. The stack that composes the block, consists of two 1×1 convolutional layer responsible for reducing and increasing dimensions, leaving the 3×3 layer in the middle with a smaller input and output. ResNet50, as it is also known, has an input dimension of $224 \times 224 \times 3$, where the last dimension represents the color channels.

3.2. CNN-LSTM Architecture

When dealing with a classification task including 3D images, a simple and direct way of extracting spatial features would be to build a three dimensional CNN. However, when dealing with large 3D images (in our case $224 \times 224 \times 24$ voxels) a deeper CNN is required to be able to accurately classify those images. Going deeper means that a bigger number of training samples is needed to achieve good performance, although extensive data sets of medical images, specially 3D images, are not readily available. This creates a problem, as 3D CNNs are no longer a solution to the classification and recognition tasks of medical images, due to their large number of learnable parameters and low number of samples in existing data-sets, that lead to the low performance in these deep models [16].

In this work we propose a new classification model, that is supported by a combination of a 2D CNN and an RNN, that learns the features of 3D knee MR images and classifies them in terms of abnormal exams, anterior cruciate ligament (ACL) and meniscus tears. As 3D images can be interpreted as times series of 2D slices, a method to capture features on 2D images can be used together with another method to extract the correlated features between slices, cooperating as one to learn and acquire the full 3D spatial features and improve image classification.

To better extract features from the 2D slices from the decomposed 3D image, the ResNet50 network was leveraged to produce feature vectors. Pretrained weights on ImageNet [17] were used to save computational power and quickly identify basic features (e.g.,edges), as training a model from scratch generally demands a larger data set than the one used in this work. Because MRNet slices are very different form the images on ImageNet, our model needed to train on the data set being fed, so fine-tuning was done and the later layers of the residual model were unfroze, as first-layer features are general and last-layer features are more specific [18]. To obtain the desired feature vector from the input slices, the last fully-connected layer was removed from the original ResNet50 network, so that the last layer was an average pooling layer outputting a vector of $n \times 1 \times 1000$, were *n* is the number of input 2D slices for each 3D image. Time-distributed wrappers were applied to the residual network, as this allows decomposing the three dimensional image as intended into several 2D image slices and apply every layer of the model to those slices.

Following the feature extraction done to the individual slices, an inter-slice extraction of features must be done to fully capture the 3D nature of the original image. Then, in order to weight in spatio-temporal correlations within the order-preserving sequence of slices a recurrent neural network must be applied, as the connections between nodes on a RNN form a directed graph along a temporal sequence. These networks can be applied in our case to obtain inter-slice features, as the time-distributed wrapped residual network outputs sequential data from our 3D input.

To fulfill the main intuition of extracting inter-slice features, and correlating features from correlating slices, an RNN type layer was added after the ResNet50 output. A LSTM layer was chosen, as this is able to model relationships between inputs separated by large sequences of data, by having a recurrent hidden state regulated through gates.

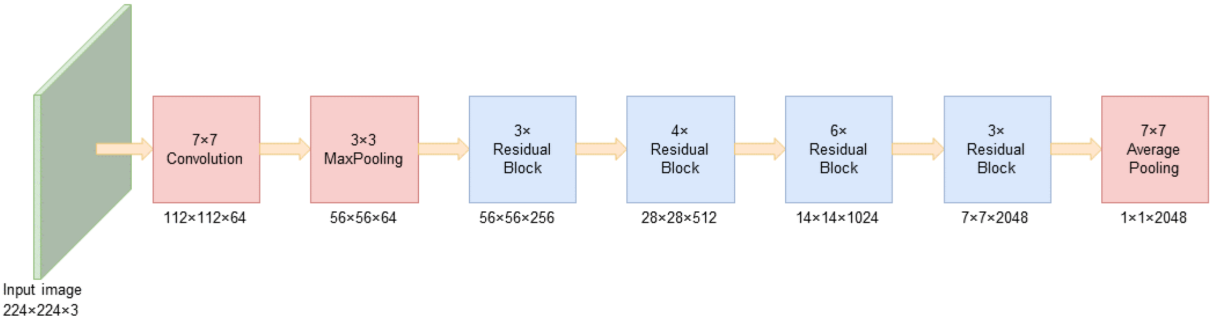

Figure 3: Illustration of the ResNet50 architecture.

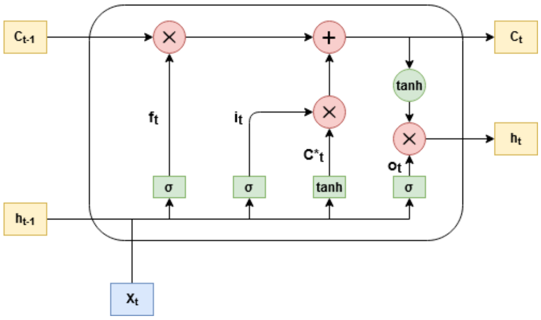

Figure 4: LSTM unit.

The key to a LSTM is the cell state and the ability to update it by using gates. At time step *t*, that in our model corresponds to a slice t for a given input sequence of 3D images, a sigmoid layer called forget gate $\mathbf{f}_t$ decides which part of the memory cell will be forgotten or kept, an input gate $\mathbf{i}_t$ controls which values are going to be updated and a *tahn* gate $\mathbf{C^*}_t$ creates a vector of new candidate values. These last two results are combining to create an update to the cell state. Lastly, a gate $\mathbf{o}_t$ produces an output based on a filtered cell state. These gate values are calculated through linear combinations of the current input $\mathbf{x}_t$ and the previous state $\mathbf{h}_t$ with a sigmoid function (σ). An LSTM unit can be formally defined as follows.

$$\begin{aligned}\mathbf{f}_t &= \sigma(\mathbf{W}_f \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f)\\ \mathbf{i}_t &= \sigma(\mathbf{W}_i \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i)\\ \mathbf{C^*}_t &= \tanh(\mathbf{W}_C \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_C)\\ \mathbf{o}_t &= \sigma(\mathbf{W}_o \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o)\end{aligned} \tag{4}$$

$$\begin{aligned}\mathbf{C}_t &= \mathbf{C}_{t-1} \times \mathbf{f}_t + \mathbf{C^*}_t \times \mathbf{i}_t\\ \mathbf{h}_t &= \mathbf{o}_t \times \tanh \mathbf{C}_t\end{aligned} \tag{5}$$

The main intuition for designing the complete architecture was that the features of the correlated slices should also be correlated. To do so, a LSTM layer is added to our model at the end of the time-distributed part, so as to impose this correlation explicitly. In the model, a bidirectional extension for the LSTM unit (bi-LSTMs) was used to enable the network to learn spatio-temporal correlations of the slices in a forward direction ($\overrightarrow{h_{it}}$), and in a backward direction ($\overleftarrow{h_{it}}$). Both states from the independent LSTM cells are concatenated, $h_{it} = [\overrightarrow{h_{it}}, \overleftarrow{h_{it}}]$, providing a more wide-raging summary of the inter-slice features.

The resulting output from the bidirectional LSTM layer was then combined with a fully-connected layer, before a final layer containing a sigmoid activation function outputs a prediction in the range of $[0, 1]$. This end to end network, that can be seen in Figure 1, was entirely trained using the Adam optimizer [5] with a learning rate starting at 0.0001. This small learning rate was chosen due to the impact high learning rates have on pre-trained networks, as we risk losing previous knowledge by distorting the CNN weights too soon and too much. To calculate the error and propagate it via back-propagation, the binary cross-entropy loss was employed as the loss function.

Due to MRNet data set containing different image planes for each same training sample (i.e. different 3D images for the axial, coronal, and sagittal plane of a sample), nine networks were trained in total, one for every plane and classification task. In order to simplify the results obtained, and hopefully improve the quality of the prediction, a logistic regression was trained to combine the output probabilities of each plane on the same classification task, and generate a single prediction. This method was selected as it allows us to give less weight to a less accurate network, and more weight to a better performing network in the final prediction. To this end, three logistic regression models were trained, one for each classification task (i.e. abnormal exams, ACL tear, and meniscus tear).

Table 1: Statistical characterization of the data set used in the experiments

Statistics	Training	Validation
Exams with abnormality (%)	913 (80.80)	95 (79.17)
Exams with ACL tear (%)	208 (18.41)	54 (45.00)
Exams with meniscal tear (%)	397 (35.13)	52 (43.33)
Exams with ACL and meniscal tear (%)	125 (11.06)	31 (25.83)
Total number of exams	1,130	120

3.3. A Multi-label Approach

As an alternative to the CNN-LSTM approach presented in the previous section, a multi-label architecture was also proposed with the intention of simplifying and speeding up the process of training the neural networks proposed, as training 3 networks (i.e., one for each plane) is faster than training 9. This model relies heavily in the CNN-LSTM architecture, as it features the same ResNet model pre-trained on ImageNet weights for intra-slice feature extraction, and a LSTM layer to correlate slices and capture inter-slice features.

In more detail, this network has a common branch to all outputs composed of non-trainable layers of the ResNet50, time-distributed as mentioned before, before branching out into the different outputs. Each branch contains the remaining unfrozen layers of ResNet, minus the fully-connected layer, as it was removed. The different branches are able to train over the intended targets. The resulting feature vector is fed onto the branch Bi-LSTM layer, before passing through a fully-connected layer with a sigmoid activation function, to finally output a prediction in the range of $[0, 1]$. It is important to notice that the branches do not share parameters between themselves. This multi-label approach was also trained using the Adam optimizer, and the same learning rate and loss function as the previous CNN-LSTM model.

4. Experiments and Discussion

This section presents the experimental evaluation of the proposed deep learning architecture. Section 4.1 describes the data-set and its characteristics, together with the pre-processing and augmentation methods utilized. Section 4.2 presents the obtained results obtained. The first experiment details the results over the MRNet data-set with base models, while the second experiment presents the obtained results with the proposed models on the knee MRI classification task.

4.1. Data Set Analysis and Experimental Methodology

As previously mentioned, with increasingly more databases of medical images available, machine learning methods can now leverage this data, with the objective of improving workflows and playing a crucial role in assisting clinicians. An example can be the data set used in this work. MRNet [4] provides knee MRI exams performed at the Stanford University Medical Center between January 1, 2001, and December 31, 2012, that were manually reviewed in order to build a data set of 1,250 knee MRI examinations.

The data set contains 1,008 (80.64%) abnormal exams, with 262 (20.96%) anterior cruciate ligament (ACL) tears and 449 (35.92%) meniscal tears. ACL tears and meniscal tears occurred concurrently in 156 (12.48%) exams. Examinations were performed with a standard knee MRI coil and a routine non-contrast knee MRI protocol. From each exam, sagittal plane T2-weighted series, coronal plane T1-weighted series, and axial plane PD-weighted series were extracted to constitute the data set. The number of images in these series ranged from 17 to 61 (mean 31.48, SD 7.97). The exams are split into a training set (1,130 exams from 1,088 patients) and a validation set (120 exams from 113 patients). These figures can be seen in close detail on Table 1.

Taking into account the range of the number of slices in each image, pre-processing the data set was needed, as typical neural networks only allow a fixed size of data to be fed into. The original images from the data set have a $n \times 256 \times 256$ size voxel, were *n* is the number of slices present in each image. To fix the number of slices, interpolation was applied to every sample to resize the number of sequences to 24. This number of slices was chosen due to its closeness to the mean average, and reduced size while still maintaining relevant information. Image size was also altered to fit the ResNet with pre-trained weights on ImageNet, as these weights were trained on 224×224 images, so a re-scaling was done from 256×256.

A data augmentation strategy was also utilized to help reduce over-fitting as the original data set is of a small size. The employed technique is based on the method presented by Hendricks et al. [19] called AugMix. This data augmentation scheme improves on previous techniques by mixing together the results of several augmentation chains in convex combinations, avoiding aggressive augmentation methods and augmentation primitives in chain that can lead to quick image degradation. This technique was applied to our data set in a limited fashion, as medical images should not consider some of the transformations used in the origi-

Table 2: Comparison between three deep learning algorithms for image classification.

Model	Axial				Coronal				Sagittal			
	Accuracy	Precision	Recall	AUC	Accuracy	Precision	Recall	AUC	Accuracy	Precision	Recall	AUC
VGG16												
Abnormal	0.825	0.870	0.916	0.832	0.850	0.897	0.916	0.739	0.775	0.895	0.811	0.829
ACL	0.775	0.765	0.722	0.825	0.625	0.585	0.574	0.697	0.700	0.514	0.556	0.787
Meniscus	0.650	0.566	0.827	**0.796**	0.675	0.603	0.731	0.751	**0.683**	**0.630**	0.654	0.731
ResNet50												
Abnormal	**0.867**	0.869	**0.979**	0.890	0.792	0.872	0.863	0.757	0.833	**0.912**	0.874	**0.895**
ACL	**0.808**	**0.830**	0.722	**0.849**	0.717	0.700	0.648	0.806	0.633	0.593	0.593	0.674
Meniscus	0.608	0.532	0.808	0.667	0.617	0.550	0.635	0.689	0.625	0.554	0.692	0.722
DenseNet201												
Abnormal	0.808	0.909	0.802	0.820	0.758	0.859	0.832	0.666	0.858	0.906	0.916	0.846
ACL	0.633	0.561	**0.852**	0.650	0.550	0.500	0.019	0.544	0.758	0.805	0.611	0.767
Meniscus	0.592	0.520	0.750	0.662	0.675	0.594	0.789	0.680	0.658	0.571	**0.846**	0.732

nal paper, and each transformation has to be replicated to all slices in a sample. Small gamma variations, rotations and translations were performed in the data-set which allowed us to double the number of training samples.

Model training was done with batches of 32 instances, using the Adam optimizer [5] with a learning rate starting at 0.0001, that could be redefined if validation loss had stopped improving employing a technique available on the Keras library called *ReduceLROnPlateau*. The number of epochs was also defined through a criteria based on a validation loss, stopping when that metric had not improved in 8 epochs. To assess the quality of the predictions, the following metrics were used: accuracy, macro precision, macro recall, and area under the ROC curve (AUC).

4.2. Experimental Results

To choose which convolutional neural network would better fit our needs as a feature extractor for intra-slices, three different architectures were tested on the knee MRI classification task. These models include VGG16, ResNet50, and DenseNet201. All were trained in the MRNet data set as out-of-the-box, architectures pre-trained with ImageNet weights. To allow the use of these models with 3D images, time-distributed wrappers were used, and a global average pooling layer was added to the end, in order to to group the slices together. The predictions were obtained through a sigmoid activation layer.

Table 2 presents the results obtained across all models on the MRNet data set. In the first set of experimental results, it is possible to verify that the ResNet50 architecture, with pre-trained weights on ImageNet, outperformed the other models on abnormality and ACL tear classification, while DenseNet201 seemed to be the worse model except on the sagittal plane were it outperform the rest of the architectures. The VGG16 model managed to be perform better on the meniscal tear task on the sagittal plane. Globally, these models with small alterations were able to achieve good results. However, the most stable architecture in all labels and planes was ResNet50, as it performed reliably better than the other methods, accomplishing an AUC of 0.895 in the abnormal classification task with sagittal plane images. This result was expected, as VGG16 is a shallower network than ResNet50, and DenseNet201 is a very deep network and thus had learning limitations due to the small data set.

Table 3 presents results obtained for the knee MRI classification task in the MRNet data set, with the proposed approaches in this work, namely the CNN-LSTM network and the Multi-label CNN-LSTM model. This set of experimental results was obtained after the predicted probabilities of the models for the different planes were combined using logistic regression. The most beneficial series, determined from the coefficients of the fitted logistic regression, were the sagittal plane for abnormalities, the axial plane for ACL tears and the coronal plane for meniscal tears, for the CNN-LSTM architecture. For the alternative model the most beneficial series were the axial plane for abnormalities and ACL tears and the coronal plane for meniscus tears.

Table 3: Comparison between our CNN-LSTM model and the alternative multi-label approach.

Model		**Accuracy**	**Precision**	**Recall**	**AUC**
CNN-LSTM + Logistic Regression	Abnormal	0.908	0.906	0.908	0.839
	ACL	0.875	0.877	0.875	0.870
	Meniscus	0.700	0.712	0.701	0.706
Multi-Label Model + Logistic Regression	Abnormal	0.858	0.880	0.858	0.660
	ACL	0.842	0.857	0.842	0.831
	Meniscus	0.700	0.706	0.701	0.701
MRNet [4]	Abnormal	0.850	-	-	0.937
	ACL	0.867	-	-	0.965
	Meniscus	0.725	-	-	0.847
Unassisted general radiologist [4]	Abnormal	0.894	-	-	-
	ACL	0.920	-	-	-
	Meniscus	0.849	-	-	-

The CNN-LSTM network managed to achieve better results than those achieved by the Multi-

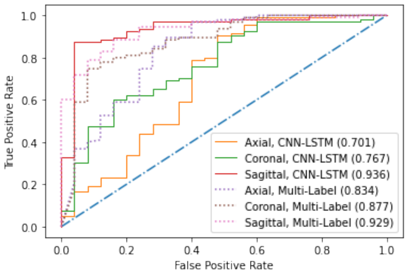

Figure 5: Illustration of the abnormal exam ROC curves.

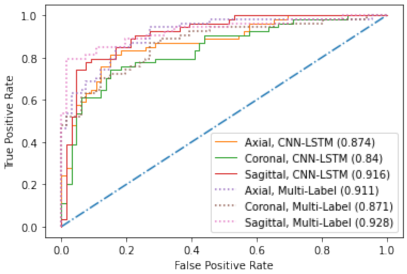

Figure 6: Illustration of the ACL exam ROC curves.

label Model in all categories. The main architecture attained a top accuracy of 0.908 on the abnormal task, and a top AUC of 0.870. Results for the multi-label model managed to be acceptable as well, as it performed better than the better base model. However, it was understandable that the simpler model would fare better than the more complex, as more parameters with a small data set can lead to degradation of results.

In Figure 5, AUC results are presented for the abnormal classification tasks, were we can verify that the better performing network is CNN-LSTM in the sagittal plane, with an AUC of 0.936. This explains why the sagittal plane was the most beneficial series when classifying abnormalities, as it performed much better than the other planes. When looking at Figure 6, it is observable that the AUC values are very similar between the models. The same can be said with Figure 7, that illustrates the AUC values of the meniscal tear classification task.

When comparing our best results, obtained with the CNN-LSTM approach, to the original implementation of MRNet on the same data set by Bien et al. [4], we verify that for the AUC metric, the MRNet model performed better in abnormality tear detection, ACL tear detection, and meniscal tear detection, with AUC values of 0.937, 0.965 and 0.847, respectively. However, in the accuracy metric, our model performed better in abnormality detection and ACL tear detection when compared against MRNet (which obtained 0.850 and 0.867, respectively, in that metric). Additionally, when compared to results obtained by unassisted general radiologists in abnormality detection, provided in the MRNet paper, both of our models have no significant differences in the performance metrics.

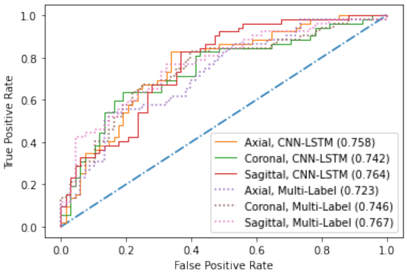

Figure 7: Illustration of the meniscal exam ROC curves.

5. Conclusions and Future Work

This work presented an approach, based on CNNs and RNNs, to adress the task of classifying multi planar 3D medical images, specifically knee MRIs according to 3 classes. Considering previous studies reported in the field of deep learning towards the task of classifying 3D medical images, a novel neural network was introduced. The architecture starts with a convolutional neural network trained to extract features from intra-slices fed from input 3D images, followed by a recurrent neural network to correlate and extract inter-slice features, outputting a prediction in a end-to-end network. The full proposed model, adding a Long Short Term Memory (LSTM) layer after the CNN feature extraction contributed to a better performance, and added the capability of modeling relationships between slices. Logistic regression was used to combine multiple predictions from distinct planes, which also contributed for the good performance of the model, as less accurates prediction weighted less on the final output. Despite the results that were obtained, there is room for improvement. As future work, I believe that combining the several planes through a different approach could lead to performance improvements and better computational costs. Future work can also consider implementing a more recent deep learning network for feature extraction such as EfficientNet [20].

Acknowledgements

This document was written and made publicly available as an institutional academic requirement and as a part of the evaluation of the MSc thesis in Biomedical Engineering of the author at Instituto Superior Técnico. The work described herein was performed at the Instituto de Engenharia de Sistemas e Computadores, Investigação e Desenvolvimento of Instituto Superior Técnico (Lisbon, Portugal), during the period March-December 2020, under the supervision of Prof. Bruno Martins.

References

[1] R. Golan, C. Jacob, and J. Denzinger. Lung nodule detection in ct images using deep convolutional neural networks. In *Proceeding of the 2016 International Joint Conference on Neural Networks (IJCNN)*, pages 243–250, 2016.

[2] Harshit S. Parmar, Brian Nutter, Rodney Long, Sameer Antani, and Sunanda Mitra. Deep learning of volumetric 3D CNN for fMRI in Alzheimer's disease classification. In *Medical Imaging 2020: Biomedical Applications in Molecular, Structural, and Functional Imaging*, volume 11317, pages 66 – 71. International Society for Optics and Photonics, SPIE, 2020.

[3] Manhua Liu, Danni Cheng, and Weiwu Yan. Classification of Alzheimer's Disease by Combination of Convolutional and Recurrent Neural Networks Using FDG-PET Images. *Frontiers in Neuroinformatics*, 12, 06 2018.

[4] Nicholas Bien, Pranav Rajpurkar, Robyn L. Ball, Jeremy Irvin, Allison Park, Erik Jones, Michael Bereket, Bhavik N. Patel, Kristen W. Yeom, Katie Shpanskaya, Safwan Halabi, Evan Zucker, Gary Fanton, Derek F. Amanatullah, Christopher F. Beaulieu, Geoffrey M. Riley, Russell J. Stewart, Francis G. Blankenberg, David B. Larson, Ricky H. Jones, Curtis P. Langlotz, Andrew Y. Ng, and Matthew P. Lungren. Deep-learning-assisted diagnosis for knee magnetic resonance imaging: Development and retrospective validation of MRNet. *PLoS Medicine*, 15(11):1–19, 2018.

[5] Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, pages 1–15, 2015.

[6] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. Association for Computational Linguistics, October 2014.

[7] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[8] Alexey A Novikov, David Major, Maria Wimmer, Dimitrios Lenis, and Katja Buhler. Deep sequential segmentation of organs in volumetric medical scans. *IEEE Transactions on Medical Imaging*, 38(5):1207–1215, 2019.

[9] Pranav Rajpurkar, Jeremy Irvin, Robyn L. Ball, Kaylie Zhu, Brandon Yang, Hershel Mehta, Tony Duan, Daisy Ding, Aarti Bagul, Curtis P. Langlotz, Bhavik N. Patel, Kristen W. Yeom, Katie Shpanskaya, Francis G. Blankenberg, Jayne Seekins, Timothy J. Amrhein, David A. Mong, Safwan S. Halabi, Evan J. Zucker, Andrew Y. Ng, and Matthew P. Lungren. Deep learning for chest radiograph diagnosis: A retrospective comparison of the CheXNeXt algorithm to practicing radiologists. *PLoS Medicine*, 15(11):1–17, 2018.

[10] Sergey Korolev, Amir Safiullin, Mikhail Belyaev, and Yulia Dodonova. Residual and plain convolutional neural networks for 3D brain MRI classification. In *International Symposium on Biomedical Imaging*, pages 835–838, 2017.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2012.

[12] Fang Liu, Zhaoye Zhou, Alexey Samsonov, Donna Blankenbaker, Will Larison, Andrew Kanarek, Kevin Lian, Shivkumar Kambhampati, and Richard Kijowski. Deep learning approach for evaluating knee MR images: Achieving high diagnostic performance for cartilage lesion detection. *Radiology*, 289(1):160–169, 2018.

[13] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2323, 1998.

[14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd International Conference on Learning Representations*, pages 1–14, 2015.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016.

[16] Dinggang Shen, Guorong Wu, and Heung-Il Suk. Deep learning in medical image analysis. *Annual Review of Biomedical Engineering*, 19(1):221–248, 2017. PMID: 28301734.

[17] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei Fei Li. Imagenet: a large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 06 2009.

[18] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, volume 27, pages 3320–3328. Curran Associates, Inc., 2014.

[19] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. AugMix: A simple data processing method to improve robustness and uncertainty. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[20] Mingxing Tan and Quoc V. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 2019-June, pages 10691–10700, 2019.