Knee Images Classification Using Transfer Learning

Kamel Rahouma and Ahmed Salama

November 18, 2020

Knee Images Classification using Transfer Learning

Kamel Rahouma[1] and Ahmed Salama[2]
[1] Faculty of Engineering, Minia University, Minia, Egypt
[2] Egyptain Space Agency, Cairo, Egypt.
kamel_rahouma@yahoo.com, ah_salama_eng_eg@yahoo.com

Abstract

Deep learning techniques are very common and mostly used in the last few years because it has a good ability for feature extraction. Convolution Neural Network (CNN) is a deep learning technique, which concerns with classification problems. In this work, the transfer learning technique is applied in training a CNN to classify images of knee to their respective classes. The proposed method uses the *NASNet Mobile* Network pre-trained Convolution Neural Network.

1 Introduction

The goal of learning techniques is to train a CNN [1] that would be able to classify knee diseases into these two classes Norm or not norm. The Magnetic Resonance Imaging (MRI) of the knee images [2], collected by Stanford ML [3], is used in this work. Each MRI image has three scans taken from three different planes e.g. sagittal, coronal and axial. Each scan consists of a number of slices [4] with fixed size 256X256 pixel. The number of scan slices is not necessary to be fixed even if for the same MRI image. The dataset classifies the knee injuries [5] into three main categories for each plan: abnormal, Anterior Cruciate Ligament (ACL) [6] and meniscus [7].
The main problem in medical research field is the lack of datasets [8] and the time taken for data processing which is always making a problem to build up a high performance system. However, using transfer learning, we can overcome this problem.
Transfer learning [9], [10] is the process that uses the weights from pre-trained networks based on large dataset [11]. The pre-trained networks have already learnt how to extract features such as edges, lines, curves etc. [12].
To train a CNN model from the scratch successfully, we need a huge dataset and a high computational power, which is missing in most of the cases. The convolution layers are often the most computationally time consuming parts of the process. Using those weights helps the network to converge to a good score faster than training from the scratch.

Each pre-trained CNN has its own characteristics. Choosing a pre-trained CNN is a tradeoff between network characteristics; accuracy, speed, and size. Figure 1 shows the validation accuracy versus prediction time for most common pre-trained network based on imageNet dataset [13, 14]

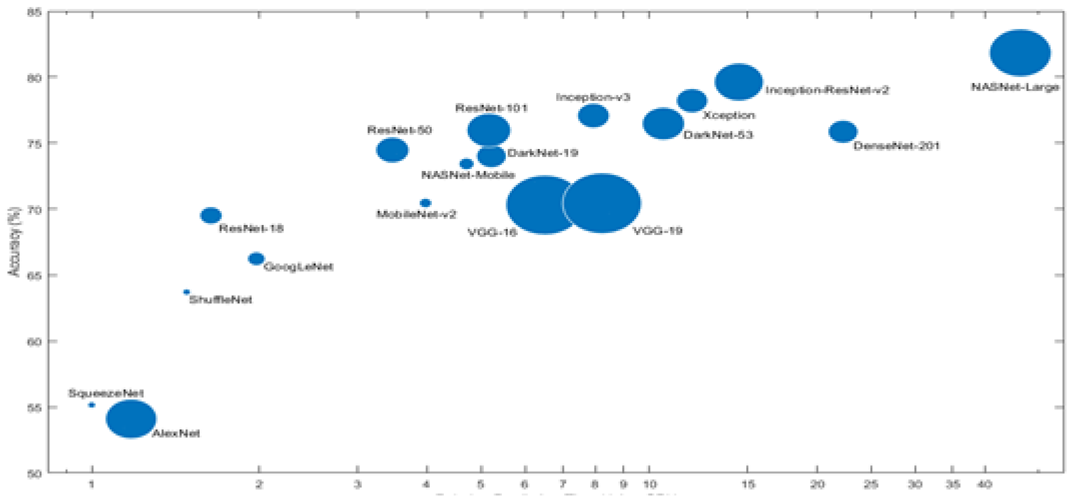


Figure 1: Relative Prediction Time Using GPU

The rest of the paper is organized as follow. In section (2), we discuss the general structure of CNN. Section (3) discusses briefly some related work, including the use of transfer learning in the medical field. Section (4) describes the main methodology followed by algorithm steps and finally the proposed system flow chart. Section (5) shows the experimental results and section (6) highlights some conclusions and proposes some ideas for future work to improve the system.

2 Convolution Neural Network

In this section, we will discuss the general structure of CNN then describe the main concept of the transfer learning. CNN consists of many sequential layers [12] e.g. stack of layers that are connected together like the bricks assemble. These layers allow sequence of transformations for the input data. One can classify these layers to: the input layer, the convolutional layer, the batch normalization layer, the ReLU layer, the cross channel normalization, the max and average pooling layers, the dropout layer, the fully connected layer and the output layer. Figure 2 shows the main CNN layers [15]

The input layer deals with images as arrays of pixels with three dimensions; height, width and depth which takes only two values 1 for gray image and 3 for colored images. A convolution layer [16] contains a set of filters used to extract the image features. The hyper-parameters of these filters are learned during the training process. All the features obtained by all filters construct a fully feature map for each image [17]. Equation 2 gives a formula that calculates the number of learnable parameters for a certain convolution layer [18].

$$\text{Paramters number} = ((\text{shape of width of the filter} * \text{shape of height of the filter} * \text{number of filters in the previous layer} + 1) * \text{number of filters}) \quad (1)$$

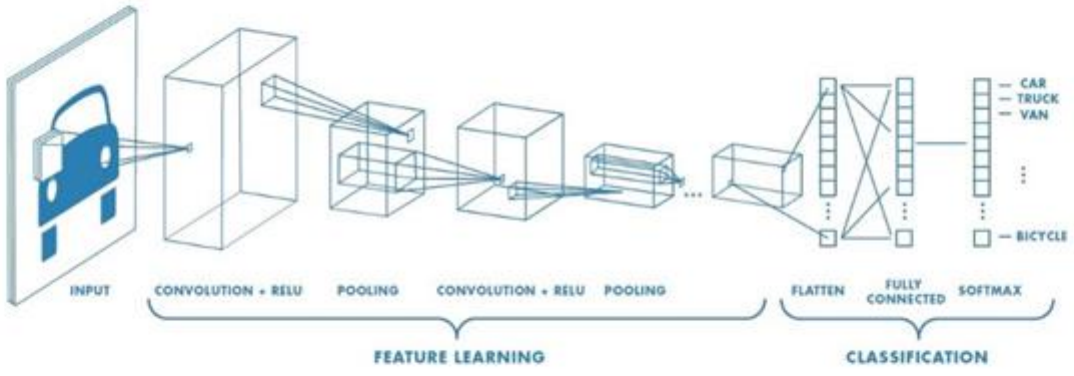


Figure 2: The main CNN layers

The batch normalization layer is used to normalize the input between convolutional layers and nonlinearities, such as ReLU layers [19]. In general, features may take scale values; for example, one can have features from 0 to 1 and some from 1 to 1000 for example. By normalization, the input value is changed between 0 and 1. Batch normalization is important to solve the internal covariate shift problem [18]. Also it speeds up training of convolutional neural networks and reduce the sensitivity to network [20]. The Rectified Linear Unit or ReLU layer is a linear function that will output the input directly if it is positive and otherwise, it will output zero [21]. It is the default activation in CNN as it solves the vanishing gradient problem [22]. Equation 2 gives the ReLU activation function:

$$R(z) = \begin{cases} z & z > 0 \\ 0 & z \leq= 0 \end{cases} \quad (2)$$

The local response normalization layer is used to solve the lateral inhibition problem; the excited neuron capacity subdues its neighbors. This problem occurs as ReLU activation layer has unbounded activations. Therefore, the high frequency features will control neighbors. The function of this layer is to make normalization around the local neighborhood of the excited neuron [23].

The Max and average pooling layer, or down-sampling layer, reduces the dimensionality of each feature map and retains the most important information of an image. Thus, it reduces the number of parameters to learn. This will reduce the computation cost [24]. The most two common methods used in pooling are the average pooling and the max pooling that summarize the average presence of a feature and the most activated presence of a feature respectively. The dropout layer (optional) is used to drop out a random set of activation layers setting them to zero. This dropout makes sure that the network is not getting too “fitted” to the training data and thus helps alleviate the over fitting problem [25].

The flatten layer takes the features obtained from the previous layers then converts them to one dimensional vector that can be an input to the next stage [26]. The fully connected layer is a fully connected NN, which receives a single row of features, calculates the NN weights, and finally applies the activation function to predict the correct label [27]. The transfer learning refers to taking a model, trained before on large dataset, and transfers the knowledge to a smaller dataset. Knowledge transfer means using the same model features extractor for the new smaller dataset after making some tuning on the pre-trained model. The tuning is accomplished by removing the last predicting layer of the pre-trained model as shown in Figure 3.

This work is based on pre-trained model NASNet Mobile. The NASNet Mobile is a CNN, which is trained on more than a million images from the ImageNet dataset [28]. This network can classify images into 1000 object categories; pen, many animals, etc. The network accepts images with size (3,224,224) and the size of feature at last layer equals 1056.

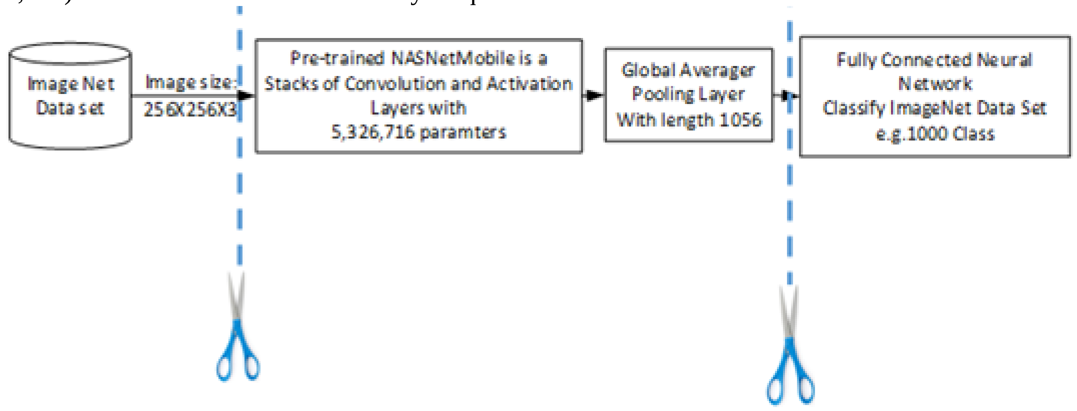


Image Net Data set
Image size: 256X256X3
Pre-trained NASNetMobile is a Stacks of Convolution and Activation Layers with 5,326,716 paramters
Global Averager Pooling Layer With length 1056
Fully Connected Neural Network Classify ImageNet Data Set e.g.1000 Class

Figure 3: The pre-trained NasNetMobile on imagenet data set

3 Literature Review

In this section, we briefly consider some of the related work that is most relevant to our approach e.g. using of transfer learning in medical field. Sonit Singh et. al. tried to use pre-trained network in medical image classification and compared the performance of the traditional pipeline of handcrafted features with multi-label learning algorithms with end-to-end deep learning features for the concept detection task [29]. Sihong Chen et. al. built a large-scale 3D medical dataset 3DSeg-8 using transfer learning [30]. Paras Lakhani et. al. applied the transfer learning in the medical field for a small dataset and got a good accuracy which is difficult to obtains if we train network from the scratch because building the network from the scratch requires a huge dataset. In this work, authors use pertained InceptionV3 [31]. Hak Gu Kim et. al. introduced a new approach based on transfer learning. Their work proposed a modality-bridge transfer learning which employs the bridge database in the same medical imaging acquisition modality as the target database [32].

4 Methodology

The main idea of this work is to use the pertained *NASNet Mobile* instead of building new CNN classifier from the scratch to extract 1056 features from the last layer after removing the top layer then designing the classifier for the images into two classes normal or not normal as follows:
1) Extract the slices for each image, extract the feature for each slice using pertained CNN NASNet Mobile.
2) Calculate the max pooling for all slices features for the same image to get one feature for each image to reduce the processing time e.g. reduce the features for each image from image slice number X 1056 to 1056.
3) Gather the features for each plane e.g. in the used data set we have 1130X1056 feature for the training dataset and 120X1056 features for the test dataset.
4) Classify the images into two classes (norm/not-norm) for each plane. In this work, two approaches are introduced as shown in Figure 5;

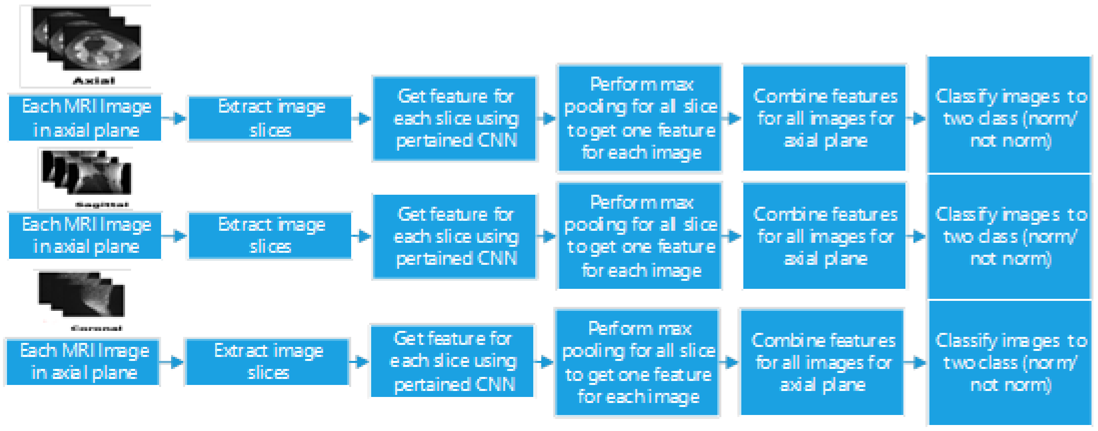

Axial
Each MRI Image in axial plane
Extract image slices
Get feature for each slice using pertained CNN
Perform max pooling for all slice to get one feature for each image
Combine features for all images for axial plane
Classify images to two class (norm/ not norm)
Sagittal
Each MRI Image in axial plane
Extract image slices
Get feature for each slice using pertained CNN
Perform max pooling for all slice to get one feature for each image
Combine features for all images for axial plane
Classify images to two class (norm/ not norm)
Coronal
Each MRI Image in axial plane
Extract image slices
Get feature for each slice using pertained CNN
Perform max pooling for all slice to get one feature for each image
Combine features for all images for axial plane
Classify images to two class (norm/ not norm)

Figure 4: The proposed system block diagram

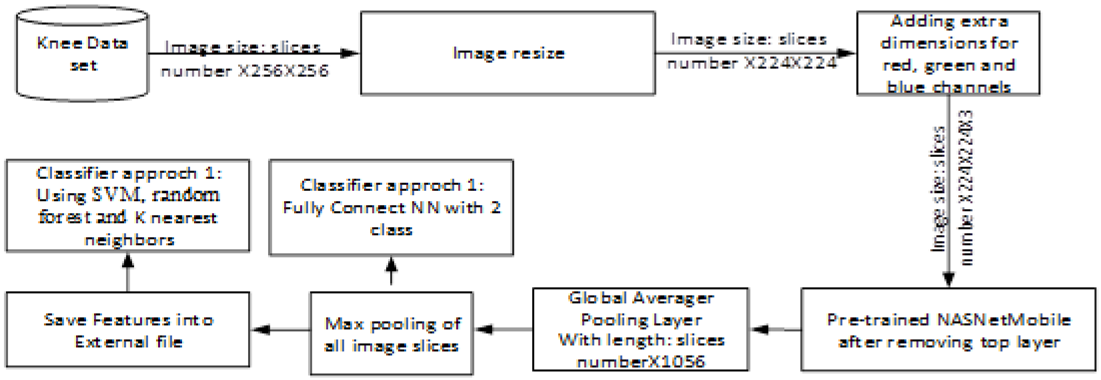

Knee Data set
Image size: slices number X256X256
Image resize
Image size: slices number X224X224
Adding extra dimensions for red, green and blue channels
Image size: slices number X224X224X3
Pre-trained NASNetMobile after removing top layer
Global Averager Pooling Layer With length: slices numberX1056
Max pooling of all image slices
Save Features into External file
Classifier approch 1: Using SVM, random forest and K nearest neighbors
Classifier approch 1: Fully Connect NN with 2 class

Figure 5: The proposed system

The first approach works by tuning the weights of the pre-trained top layer; e.g. modify the fully connected neural network weights to match with image features. The second approach works by storing the image features produced from the pre-trained network after removing top layer, then using a new different layer to classify the dataset. In this work, we use three classifiers: Support Vector Machine (SVM), random forest and K nearest neighbors. In the following we give the algorithm we apply in this work.

Algorithm steps:

1- Define Pre-trained Model NASNet.
2- Determine MRI images folder paths for each plane; *sagittal_image_path*, *coronal_image _path* and *axial_image _path*.
3- Read the training data labels e.g. train_abnormal_labels, train_ acl_labels, train_ meniscus _labels.
4- Read the test data labels e.g. test_abnormal_labels, test_ acl_labels, test_meniscus_labels.
5- Loop 1 : for each MRI images paths which defined in step 2
6- Loop 2: for each image
7- Store each image slices into array, each array has size (Number of slicesX256X256).
8- Resize the image size to be fit with Pre-trained Model *NASNet* e.g. each image has size will be (Number of slicesX224X224).

9- Adding Extra Dimensions for red, green and blue channels to fit with Pre-trained Model *NASNet* e.g. each image size will be (Number of slicesX224X224X3)
10- Extract the feature for all images slices using Pre-trained Model *NASNet* last layer e.g. feature size for each image will be (Number of slicesX1056).
11- Apply max pooling for each image slices to reduce the feature size from (Number of slicesX1056) to be 1056 e.g. feature size for each image will be (1056)
12- End Loop 2
13- Store features for all MRI images for sagittal, coronal and axial plane e.g. train_abnormal_features, train_ acl_ features, train_ meniscus _ features.
14- End Loop 1
15- Perform the model training by using
    15.1- Perform the model training by tunning pre-trained fully connected NN for each plane.
    15.2- Replace fully connected top layer with different classifier e.g. SVM, Random forest and K nearest neighbors

5 Results

After getting the features using pertained CNN NASNet Mobile, one can classify the features by two methods:

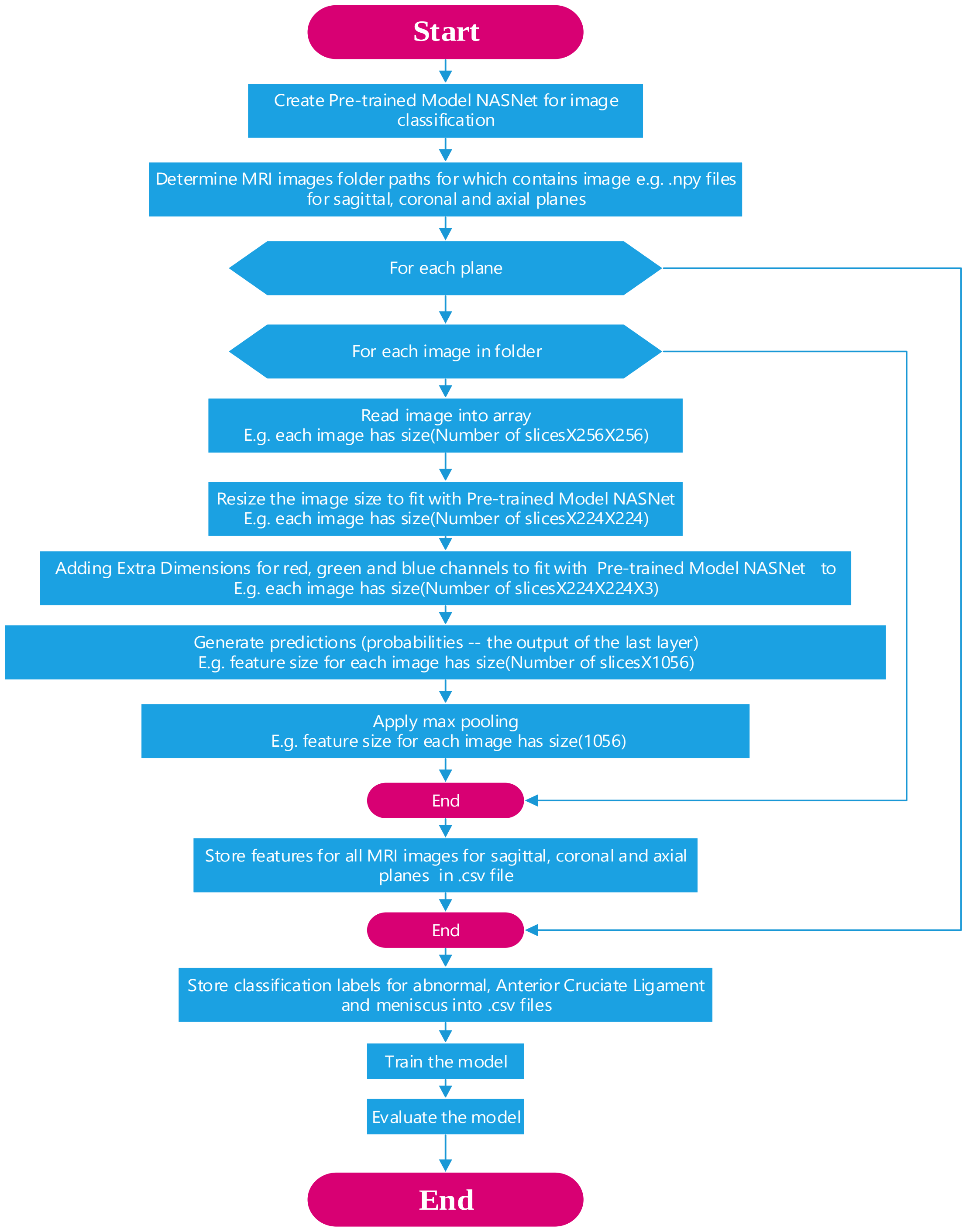
5.1 Method 1: Using fully connected neural network's top layer for classification after tuning:

In this approach, a pre-trained model is used to extract the features form images, then to tune the fully connected layer to match with the size of features. Use a neural network with 256 neurons in the input layer with 0.5 dropout out rate and finally one output layer with Sigmoid activation function to classify the image. Figures 7 - 9 show the training and validation accuracy/loss in the three planes e.g. Axial, Coronal and sagittal when we classify knee abnormal disease, Meniscus and ACL respectively according to the used dataset. Also from these figures, one can notice that the maximum obtained accuracy for all planes is nearly 91%, 85% and 88% for knee abnormal disease, Meniscus and ACL respectively.

5.2 Method 2: Storing features then classifying them

In this approach, the extracted features from a pre-trained model are first stored into an external file then classified by a different classifier instead of using fully connected neural network. In this work, we use three classifiers: SVM, random forest and k nearest.

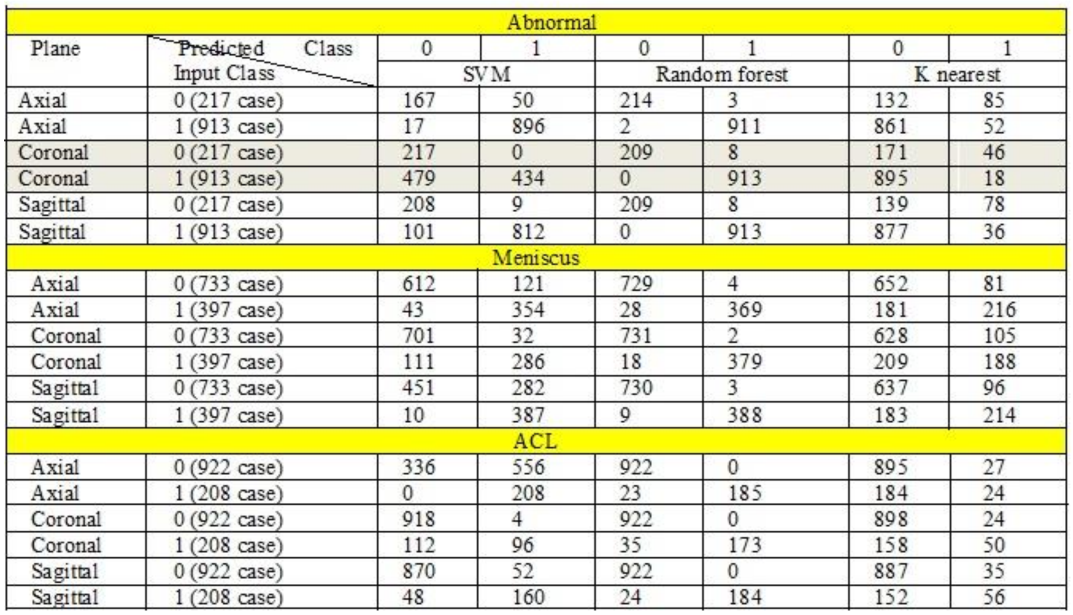
Table 1 A compares between the confusion matrix for the SVM, Random forest and K nearest neighbor algorithms. Class 0 represents normal cases and class 1 represents abnormal cases. Table 2 compares the total accuracy obtained for the SVM, Random forest and K nearest neighbor's algorithms; from the results, it is clear that the best algorithm is Random forest



Start
Create Pre-trained Model NASNet for image classification
Determine MRI images folder paths for which contains image e.g. .npy files for sagittal, coronal and axial planes
For each plane
For each image in folder
Read image into array E.g. each image has size(Number of slicesX256X256)
Resize the image size to fit with Pre-trained Model NASNet E.g. each image has size(Number of slicesX224X224)
Adding Extra Dimensions for red, green and blue channels to fit with Pre-trained Model NASNet to E.g. each image has size(Number of slicesX224X224X3)
Generate predictions (probabilities -- the output of the last layer) E.g. feature size for each image has size(Number of slicesX1056)
Apply max pooling E.g. feature size for each image has size(1056)
End
Store features for all MRI images for sagittal, coronal and axial planes in .csv file
End
Store classification labels for abnormal, Anterior Cruciate Ligament and meniscus into .csv files
Train the model
Evaluate the model
End

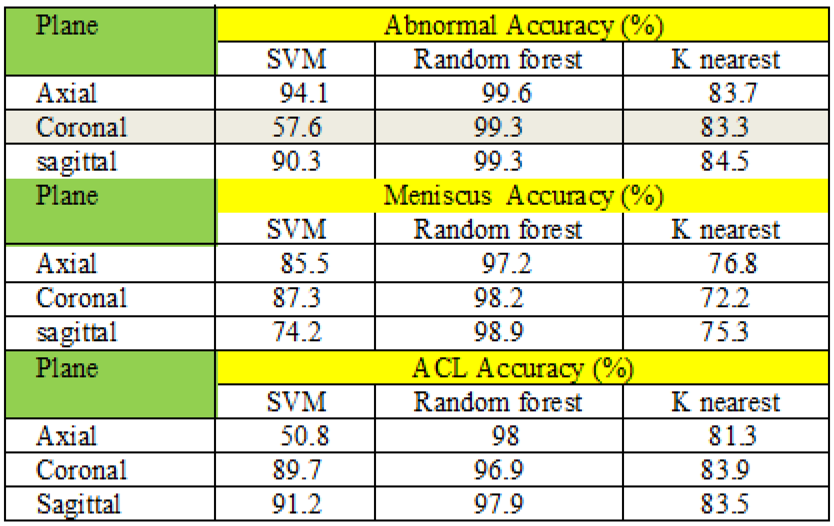
Figure 6: The proposed algorithm flow chart

Table 1: The confusion matrix for Abnormal, meniscus and ACL diseases in all planes for SVM, Random forest and k nearest classifiers



Plane	Predicted Class / Input Class	SVM 0	SVM 1	Random forest 0	Random forest 1	K nearest 0	K nearest 1
Abnormal							
Axial	0 (217 case)	167	50	214	3	132	85
Axial	1 (913 case)	17	896	2	911	861	52
Coronal	0 (217 case)	217	0	209	8	171	46
Coronal	1 (913 case)	479	434	0	913	895	18
Sagittal	0 (217 case)	208	9	209	8	139	78
Sagittal	1 (913 case)	101	812	0	913	877	36
Meniscus							
Axial	0 (733 case)	612	121	729	4	652	81
Axial	1 (397 case)	43	354	28	369	181	216
Coronal	0 (733 case)	701	32	731	2	628	105
Coronal	1 (397 case)	111	286	18	379	209	188
Sagittal	0 (733 case)	451	282	730	3	637	96
Sagittal	1 (397 case)	10	387	9	388	183	214
ACL							
Axial	0 (922 case)	336	556	922	0	895	27
Axial	1 (208 case)	0	208	23	185	184	24
Coronal	0 (922 case)	918	4	922	0	898	24
Coronal	1 (208 case)	112	96	35	173	158	50
Sagittal	0 (922 case)	870	52	922	0	887	35
Sagittal	1 (208 case)	48	160	24	184	152	56

Table 2: The classification accuracy for abnormal, meniscus and ACL diseases in all planes for SVM, random forest and k nearest classifiers



Plane	Abnormal Accuracy (%)		
	SVM	Random forest	K nearest
Axial	94.1	99.6	83.7
Coronal	57.6	99.3	83.3
sagittal	90.3	99.3	84.5
Plane	Meniscus Accuracy (%)		
	SVM	Random forest	K nearest
Axial	85.5	97.2	76.8
Coronal	87.3	98.2	72.2
sagittal	74.2	98.9	75.3
Plane	ACL Accuracy (%)		
	SVM	Random forest	K nearest
Axial	50.8	98	81.3
Coronal	89.7	96.9	83.9
Sagittal	91.2	97.9	83.5

6 Conclusions and Future work

This work we use the NASNetMobile pre-trained deep learning model to classify knee diseases. In this work also we reduce the amount of data used in training by taking the max pooling for each image slices Instead of training all image slices but gained good accuracy due to the power of retrained CNN. In this work, also we classify the features obtained from model and classify it by two methods; tuning the top layer and using external classifier. From the results we find that, classification using external classifier gives the highest accuracy for random forest classifier.

As future work, we would want to use all image slices and performing data augmentation techniques. In addition, perform image segmentation according to the diseases before performing training.

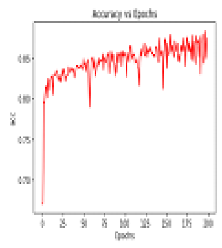
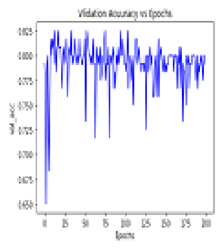
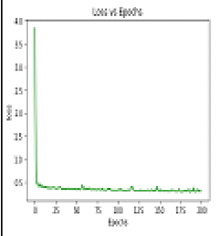
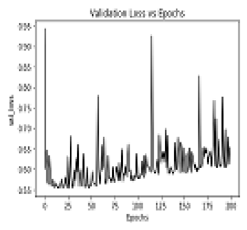
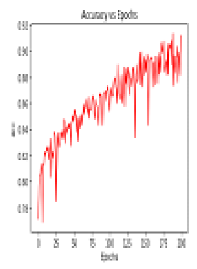
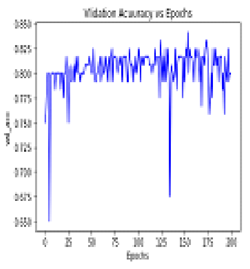
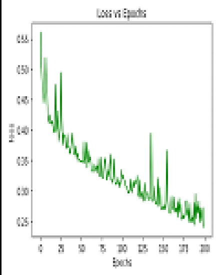
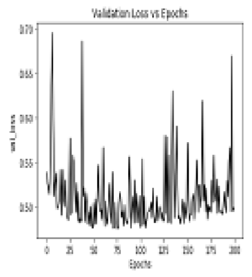
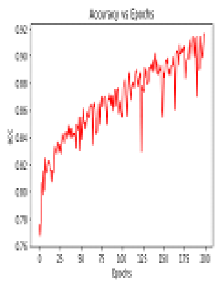
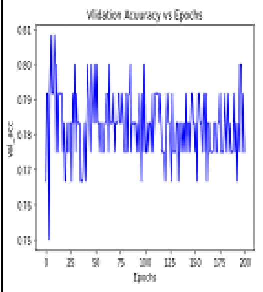
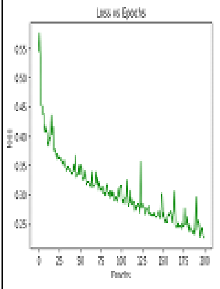
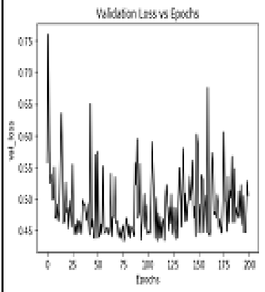
	Abnormal			
	Train Accuracy	**Validation Accuracy**	**Train Loss**	**Validation Loss**
Axial Plane				
Coronal Plane				
sagittal Plane				

Figure 7: The Train and validation accuracy/loss in all planes for abnormal disease

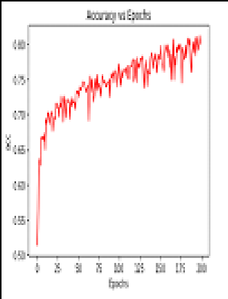
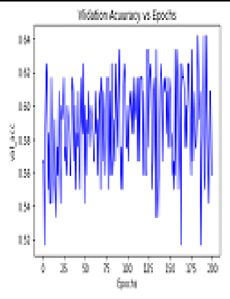
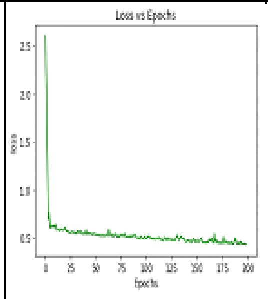
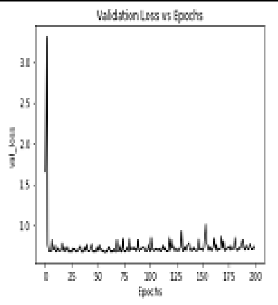
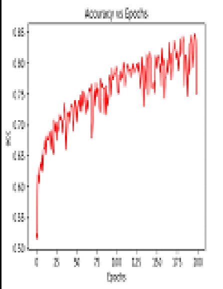
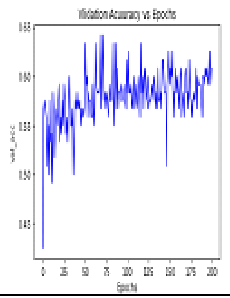
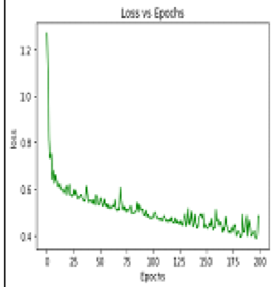
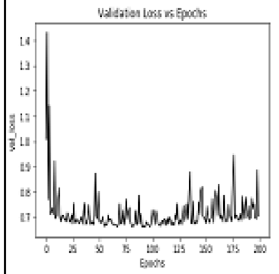
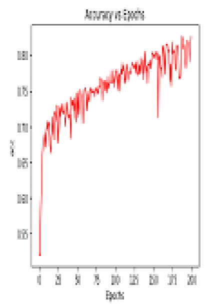
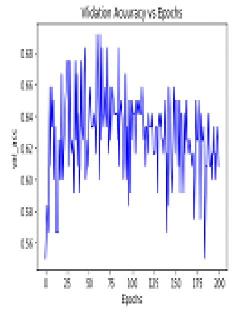
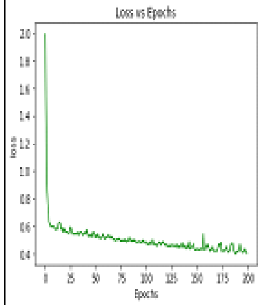
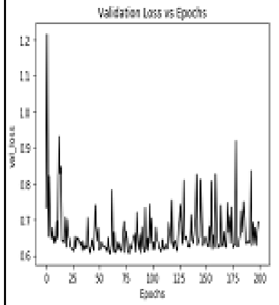
	Meniscus			
	Train Accuracy	**Validation Accuracy**	**Train Loss**	**Validation Loss**
Axial Plane				
Coronal Plane				
sagital Plane				

Figure 8: The Train and validation accuracy/loss in all planes for meniscus disease

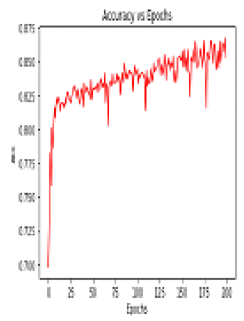
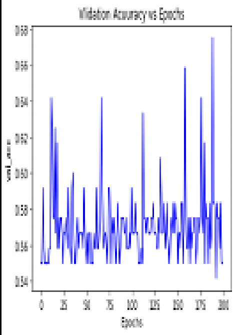
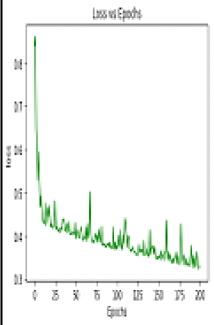
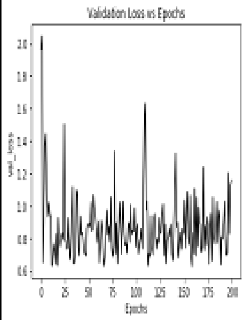
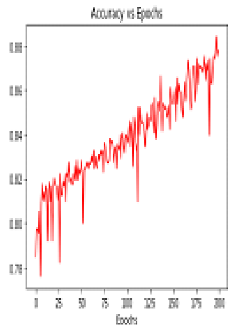
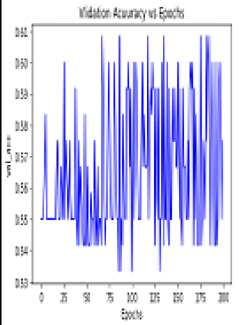
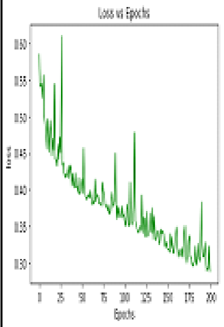
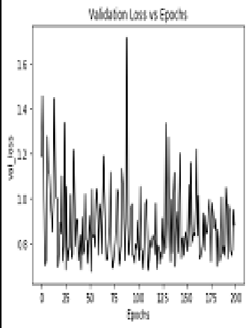
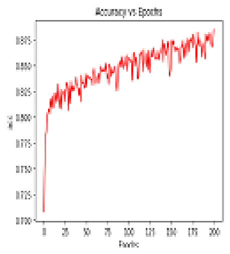
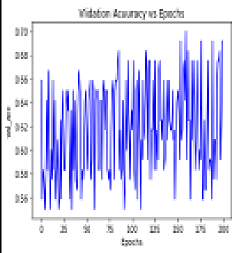
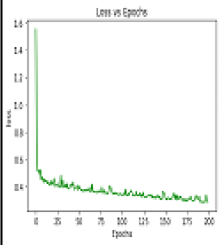
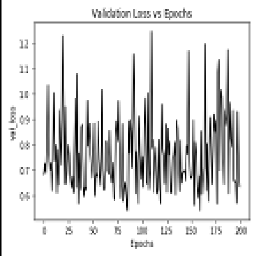
ACL				
	Train Accuracy	Validation Accuracy	Train Loss	Validation Loss
Axial Plane				
Coronal Plane				
sagittal Plane				

Figure 9: The Train and validation accuracy/loss in all planes for ACL disease

References

[1] H.-C. Shin *et al.*, “Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning,” *IEEE Trans. Med. Imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[2] Wradiology, “web,” *Atlas of Knee MRI Anatomy*, 2005. https://w-radiology.com/knee-mri/.

[3] S. M. Group, “web,” *A knee MRI Dataset*. https://stanfordmlgroup.github.io/competitions/mrnet/.

[4] Wikipedia, “web,” *Magnetic resonance imaging*. https://en.wikipedia.org/wiki/Medical_imaging.
[5] D. of O. S.-S. University, “web,” *Acute Knee Injuries*. http://www0.sun.ac.za/ortho/webct-ortho/plateau/knee.html.
[6] S. P. Arnoczky, “Anatomy of the anterior cruciate ligament,” *Clin. Orthop. Relat. Res.*, vol. 172, pp. 19–25, 1983.
[7] R. Śmigielski, R. Becker, U. Zdanowicz, and B. Ciszek, “Medial meniscus anatomy—from basic science to treatment,” *Knee Surgery, Sport. Traumatol. Arthrosc.*, vol. 23, no. 1, pp. 8–14, 2015.
[8] T. data Science, “web,” *Breaking the curse of small datasets in Machine Learning*. https://towardsdatascience.com/breaking-the-curse-of-small-datasets-in-machine-learning-part-1-36f28b0c044d.
[9] L. Torrey and J. Shavlik, “Transfer learning,” in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, IGI global, 2010, pp. 242–264.
[10] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2009.
[11] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, “Deep learning applications and challenges in big data analytics,” *J. Big Data*, vol. 2, no. 1, p. 1, 2015.
[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
[13] M. MathWorks, “Web,” *Pretrained Deep Neural Networks*. https://se.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networks.html.
[14] Stanford Vision Lab, “Web,” *ImageNet DataSet*, 2016. http://www.image-net.org.
[15] T. data Science, “web,” *A Comprehensive Guide to Convolutional Neural Networks*. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.
[16] E. at F. | U. C. ’19, “web,” *A Beginner’s Guide To Understanding Convolutional Neural Networks*. https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/.
[17] “Layers-of-a-Convolutional-Neural-Network @ Www.Mathworks.Com.” .
[18] T. data Science, “web,” *Understanding and Calculating the number of Parameters in Convolution Neural Networks (CNNs)*. https://towardsdatascience.com/understanding-and-calculating-the-number-of-parameters-in-convolution-neural-networks-cnns-fc88790d530d#:~:text=Number of parameters in a CONV layer would be %3A ((,1)*number of filters.
[19] T. data Science, “web,” *Batch normalization in Neural Networks*. https://towardsdatascience.com/batch-normalization-in-neural-networks-1ac91516821c.
[20] M. MathWorks, “web,” *batchNormalizationLayer*. https://www.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.batchnormalizationlayer.html.
[21] M. L. Mastery, “web,” *A Gentle Introduction to the Rectified Linear Unit (ReLU)*, 2020. https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/.
[22] M. L. Mastery, “web,” *How to Fix the Vanishing Gradients Problem Using the ReLU*. https://machinelearningmastery.com/how-to-fix-vanishing-gradients-using-the-rectified-linear-activation-function/.
[23] P. ENIGMA, “web,” *What Is Local Response Normalization In Convolutional Neural*

Networks. https://prateekvjoshi.com/2016/04/05/what-is-local-response-normalization-in-convolutional-neural-networks/.

[24] MachineCurve, “web,” *What are Max Pooling, Average Pooling, Global Max Pooling and Global Average Pooling?* https://www.machinecurve.com/index.php/2020/01/30/what-are-max-pooling-average-pooling-global-max-pooling-and-global-average-pooling/.

[25] M. L. Mastery, “No Title,” *A Gentle Introduction to Dropout for Regularizing Deep Neural Networks*. https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/.

[26] J. Jin, A. Dundar, and E. Culurciello, “Flattened convolutional neural networks for feedforward acceleration,” *arXiv Prepr. arXiv1412.5474*, 2014.

[27] M. Marchesi, G. Orlandi, F. Piazza, G. Pignotti, and A. Uncini, “Dynamic topology neural network,” 1990.

[28] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*, 2009, pp. 248–255.

[29] S. Singh, K. Ho-Shon, S. Karimi, and L. Hamey, “Modality classification and concept detection in medical images using deep transfer learning,” in *2018 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, 2018, pp. 1–9.

[30] S. Chen, K. Ma, and Y. Zheng, “Med3d: Transfer learning for 3d medical image analysis,” *arXiv Prepr. arXiv1904.00625*, 2019.

[31] P. Lakhani, D. L. Gray, C. R. Pett, P. Nagy, and G. Shih, “Hello world deep learning in medical imaging,” *J. Digit. Imaging*, vol. 31, no. 3, pp. 283–289, 2018.

[32] H. G. Kim, Y. Choi, and Y. M. Ro, “Modality-bridge transfer learning for medical image classification,” in *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, 2017, pp. 1–5.