Diagnosing Knee MRI Scans Using Deep Learning

Jessica Wang, Mason High School

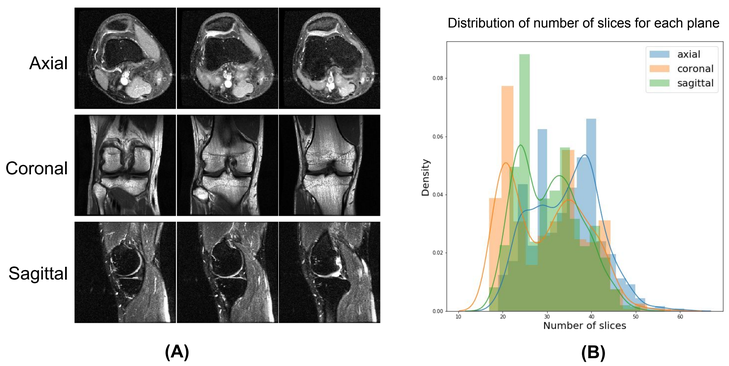
Abstract

MRNet is a machine learning competition sponsored by the Stanford Machine Learning group to encourage the development of machine learning models for knee MRI interpretation [1]. Each knee MRI exam has a sequence of cross-sectional images from three planes -- axial, coronal and sagittal. The dataset contains 1,370 knee MRI exams, which were labeled by experienced radiologist. The task is to develop a computer vision algorithm to predict three target knee pathologies (abnormal, ACL tear, and meniscal tear). The developed model was submitted to Codalab, where the competition sponsor ran the code against the withheld test dataset.

We developed a deep learning model that "reads" the MRI scans to produce a knee diagnosis. The model uses a VGG11 [2] neural network pretrained on ImageNet to extract features from each image and a WaveNet [3] module to integrate the features from the sequence of images. Finally, the feature vectors from three scanning planes were concatenated together and fed into a fully connected layer to produce the final classification. The model was trained end-to-end. In the withheld test set, our model achieved an AUC of **0.911**, which is comparable to a trained radiologist. The source code is deposited at https://github.com/jessicaxuwang/MRNet.

We experimented with various architectures for image feature extraction and sequence learning. As shown in numerous other studies, we showed that pretraining the models on ImageNet is important for good performance. Counter intuitively, simple models like AlexNet and VGG11 perform better than more complex models such as ResNet34 [4], VGG16 [2], and Densenet121 [5]. The complex models probably produced suboptimal results because of the limited training data size. We further showed that a modified WaveNet [3] like architecture performed better than other sequence learning methods such as LSTM [6] and multi-headed attention used in transformer [7] models. We hope this work will stimulate further model development for automated knee MRI diagnosis.

Dataset Description



(A)

(B)

Each knee MRI exam acquires a sequence of 2D images from three planes -- axial, coronal and sagittal. Each plane scan contains variable number of images (slices) ranging from 17 to 61. Only three slices from each plane were shown above (A). The distribution of number of slices for the exams in the training set were shown in (B).

Task: given a knee MRI exam, predict the three target knee pathologies: **Abnormal**, **ACL tear** and **Meniscal tear**.

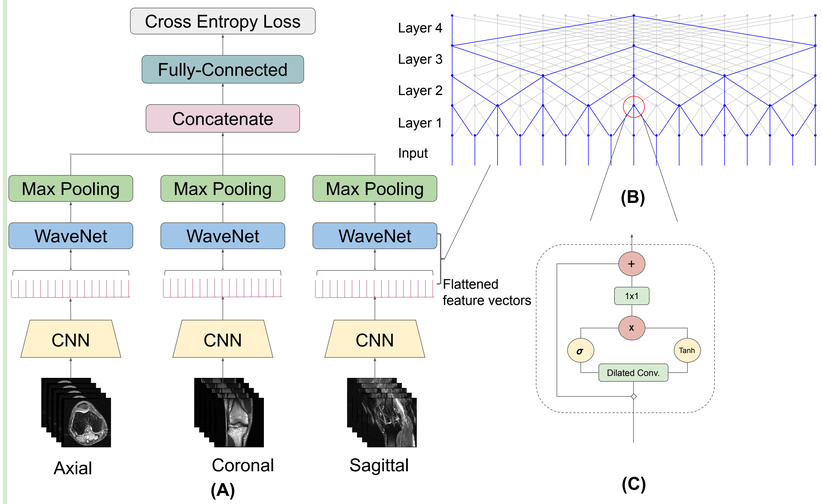
Label	Count	Percent
Abnormal	1,104	80.60%
ACL Tears	329	23.30%
Meniscal Tears	508	37.10%

Statistics of knee pathology target in the training set

Split	# of Exams	# of Patients
Train	1,130	1,088
Validation	120	111
Test	120	113

Train, validation and test split of the dataset. The splitting was performed by the competition sponsor using stratified sampling. Test set was withheld from competitors

Model Architecture



(A) Overall model architecture. Each sequences of images for sagittal, coronal and axial planes was fed into three different convolutional neural networks (CNN). The CNN transformed each image into a feature vector. The sequence of feature vectors from each scan were then fed into the WaveNet module [3], the output was combined into a single vector through max pooling. The resulting vectors for each scanning planes were concatenated together and fed into a fully-connected layer to generate a vector of size 3, one each for 'abnormal', 'ACL tear' and meniscal tear respectively. A sigmoid function was applied to the output of the fully-connected layer and a binary cross entropy loss function was used as the training objective.

(B) Wavenet [3] architecture. A slightly modified WaveNet layer was used in this study; instead of using a kernel size of 2, we uses a kernel size of 3. Also neither autoregression nor causal convolution was used. Each successive layer used exponentially growing dilation size to increase the receptive field size.

(C) Architecture of each node in the WaveNet layer as described in [3]. The dilated convolution, gated convolution and residual connections were combined to produce the output.

Model Training Procedure

- **Implementation**
  - We implemented the model using Pytorch [8].
- **Data augmentation**
  - We used albumentation [9] to augment the input images. Specifically, each stack of images is resized to 160 x 160, randomly flipped, shifted ±8%, rotated ±25 degrees and scaled to 0.88 to 1.12. We augmented the entire stack of images simultaneously instead of each individual image independently because it is important to keep the registration of the entire stack images. When testing, no augmentation is applied other than scaled the image to 160 x 160. We chose 160 x 160 as input size rather than the original 256 x 256 to reduce the memory requirement and increase the speed of training.
- **Training**
  - The model was optimized using Adam [10] optimizer with a initial learning rate 1e-5, beta1 = 0.9 and beta2 = 0.8. The learning rate was decayed using a stepwise schedule of 10, 25, 30, 35, 40 and 45 with a rate of 0.1. The models were trained a total 50 epoches. We used batch size of 1 for all of our experiments. Because different exams have different number of images, to use a batch size more than one, we need to pad each series. Each batch has an average 95 images, larger batch size requires significant more memory. We also compared with other optimizers such as SGD with momentum, RMSprop [11] and AMSGrad [12], and Adam produced better validation result.
- **Early stopping**
  - We used the validation set loss to monitor the training progress. We experimented with many different hyperparameters, the validation loss always increases in later training epochs. We used the best validation AUC checkpoint from an early epoch for final submission.

Comparing CNN Architectures

Model	Pretrained	Avg. AUC	AUC Abnormal	AUC ACL tear	AUC Meniscal Tear
VGG11	Yes	**0.912**	**0.962**	**0.931**	**0.843**
AlexNet	Yes	0.898	0.952	0.902	0.840
VGG16	Yes	0.888	0.939	0.917	0.807
ResNet34	Yes	0.878	0.936	0.908	0.790
DenseNet121	Yes	0.868	0.903	0.914	0.787
VGG11	No	**0.851**	0.892	**0.920**	0.742
VGG16	No	**0.851**	0.902	0.870	**0.780**
ResNet34	No	0.850	0.899	0.875	0.775
AlexNet	No	0.845	**0.909**	0.882	0.742
DenseNet121	No	0.691	0.713	0.714	0.646

We evaluated the performance of different CNN architectures for extracting features from images. We used either a pretrained network on ImageNet or randomly a initialized network with the same architecture. The table above shows the performance of various network architectures on the validation set. "Avg. AUC" is the average AUC of the three knee pathology targets. It is clear that pretrained network performed better than the randomly initialized network. It is somewhat surprising that simpler networks like VGG11 and AlexNet performs better than more complex network such as ResNet34 and DenseNet121. This probably due to the fact that the training set is too small to take full advantage of the more complex network architectures.

Comparing Sequence Learning Modules

	Avg. AUC	AUC Abnormal	AUC ACL tear	AUC Meniscal Tear
WaveNet + Maxpool	**0.912**	**0.962**	0.931	**0.843**
LSTM + Maxpool	0.900	0.947	**0.944**	0.810
Transformer + Maxpool	0.896	0.951	0.941	0.795
Maxpool	0.877	0.902	0.914	0.814
AvgPool	0.865	0.852	0.923	0.820

We compared different sequence learning modules to integrate the sequence of vectors generated by the CNN feature extractor (VGG11). Specifically, we compared max pool, average pool, WaveNet + Maxpool, LSTM (bidirectional LSTM) + Maxpool and Transformer (4 attention heads) + Maxpool. WaveNet performed best among all the modules tested.

Which Plane is Most Informative?

	Avg. AUC	AUC Abnormal	AUC ACL tear	AUC Meniscal Tear
Coronal	**0.893**	0.945	0.896	**0.838**
Axial	0.880	0.950	**0.904**	0.787
Sagittal	0.870	**0.952**	0.893	0.766

Each exam contains three scans for the axial, coronal and sagittal planes. We trained models with only one of the planes as input and examined its validation performance. Coronal plane produced the best average AUC. Overall the the predictive power are quite similar for the three planes.

References

1. MRNet. https://stanfordmlgroup.github.io/competitions/mrnet/
2. Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. CoRR, abs/1409.1556.
3. Oord, A.V., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A.W., & Kavukcuoglu, K. (2016). WaveNet: A Generative Model for Raw Audio. SSW.
4. He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770-778.
5. Huang, G., Liu, Z., & Weinberger, K.Q. (2016). Densely Connected Convolutional Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2261-2269.
6. Hochreiter, Sepp and Jürgen Schmidhuber. "Long Short-Term Memory." Neural Computation 9 (1997): 1735-1780.
7. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., & Polosukhin, I. (2017). Attention is All you Need. NIPS.
8. Pytorch. https://pytorch.org
9. Albumentation. https://github.com/albumentations-team/albumentations
10. Kingma, D.P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. CoRR, abs/1412.6980.
11. Hinton, G.E. Srivastava, N., & Swersky, K. (2012) Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. page 14.
12. Tran, P.T., & Phong, L.T. (2019). On the Convergence Proof of AMSGrad and a New Version. IEEE Access, 7, 61706-61716.