

Recurrent Deep Residual Network for Lung Cancer Classification

Beibei Du and Cole Springate-Combs
Contributions from Hongyan Wang

Outline

- Introduction
 - Problem statement
 - Solutions we proposed
- Related work
- Approaches
 - Using LightGBM on top of pre-trained model features
 - Using LSTM on top of pre-trained model features
- Experiments and Results
 - Datasets
 - Evaluation metrics
 - Results
- Conclusion

Introduction - Problem

Cancer or not Cancer?



Related Work and Motivation

Transfer learning:

transfer learning from AlexNet and GoogleLeNet pre-trained on imagenet [1]

3D CNN:

Success with nodule detection and identification [2]

Simple mean:

Reasonable results with mean of slices, transfer learning [3]

RNN:

RNN applied to traditional CNN on fixed amount of slices [4]

Contribution

Use transfer learning from 2D network to “2.5D” data

Add RNN LSTM features:

- allow the model to learn 3D information

- deal with the varying number of slices per patient

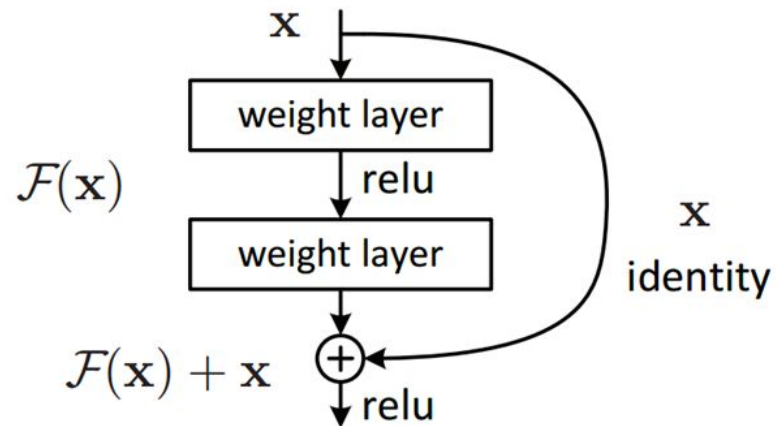
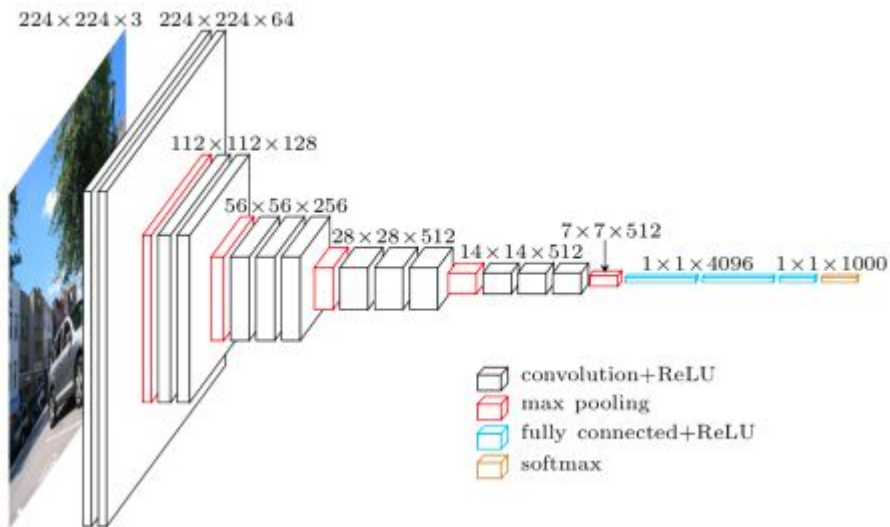
Use whole images, train one network

“High mean” layer

Introduction - solution

- CNN + LightGBM
- CNN + LSTM

Two pre-trained model: Vgg19 and ResNet50

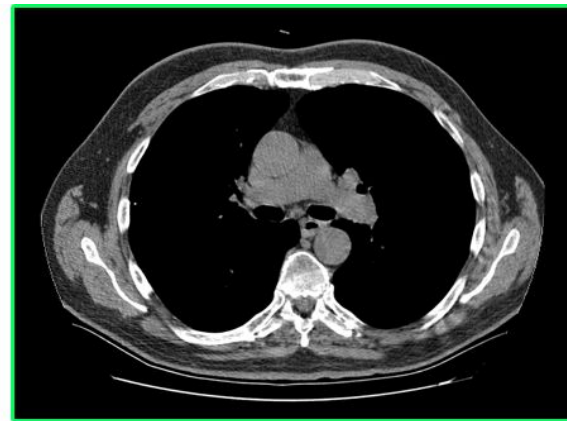
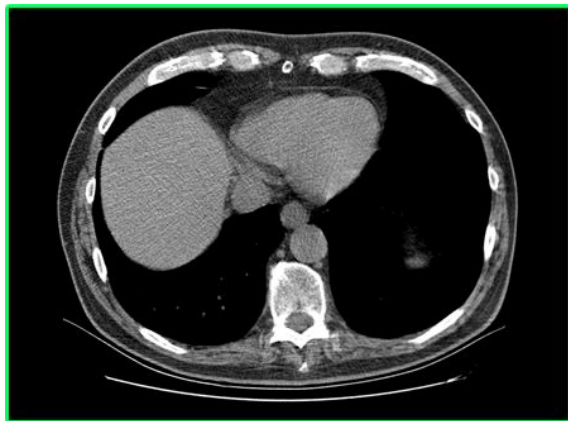


Dataset

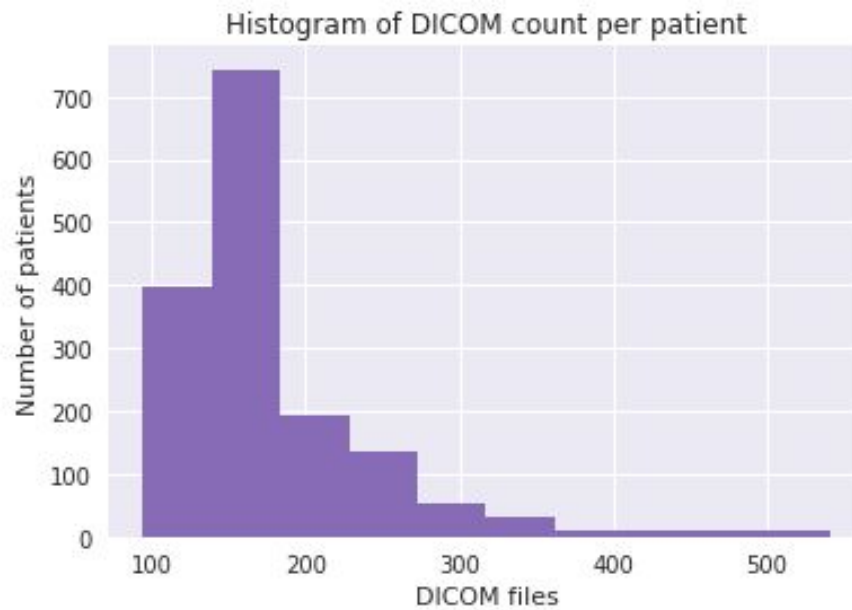
285380 Chest CT Scans

Dimension: Slices of 512 x 512 (b/w)

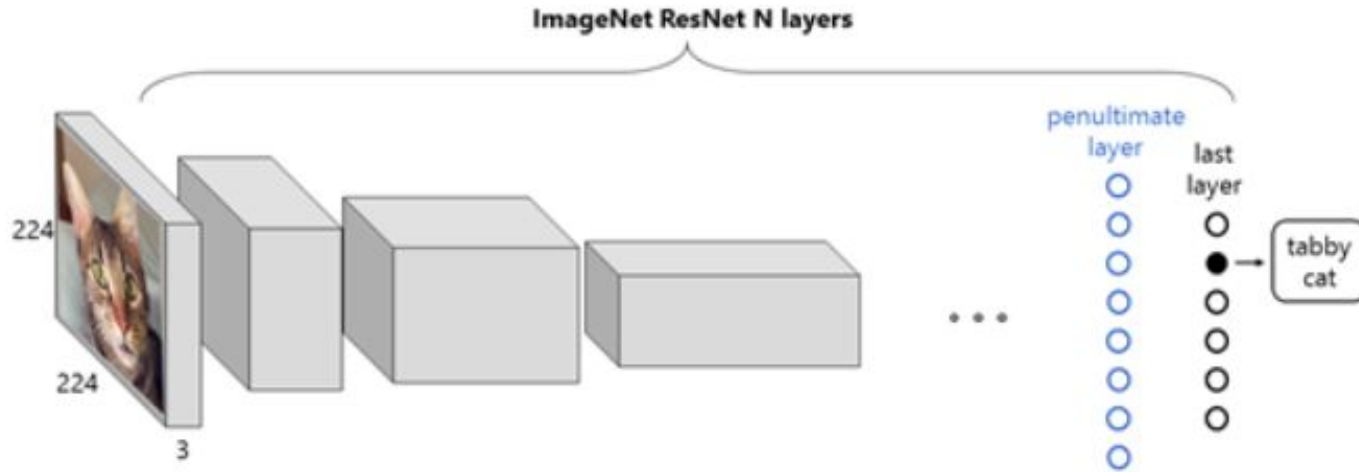
1397 patients, each with a varying number of slices



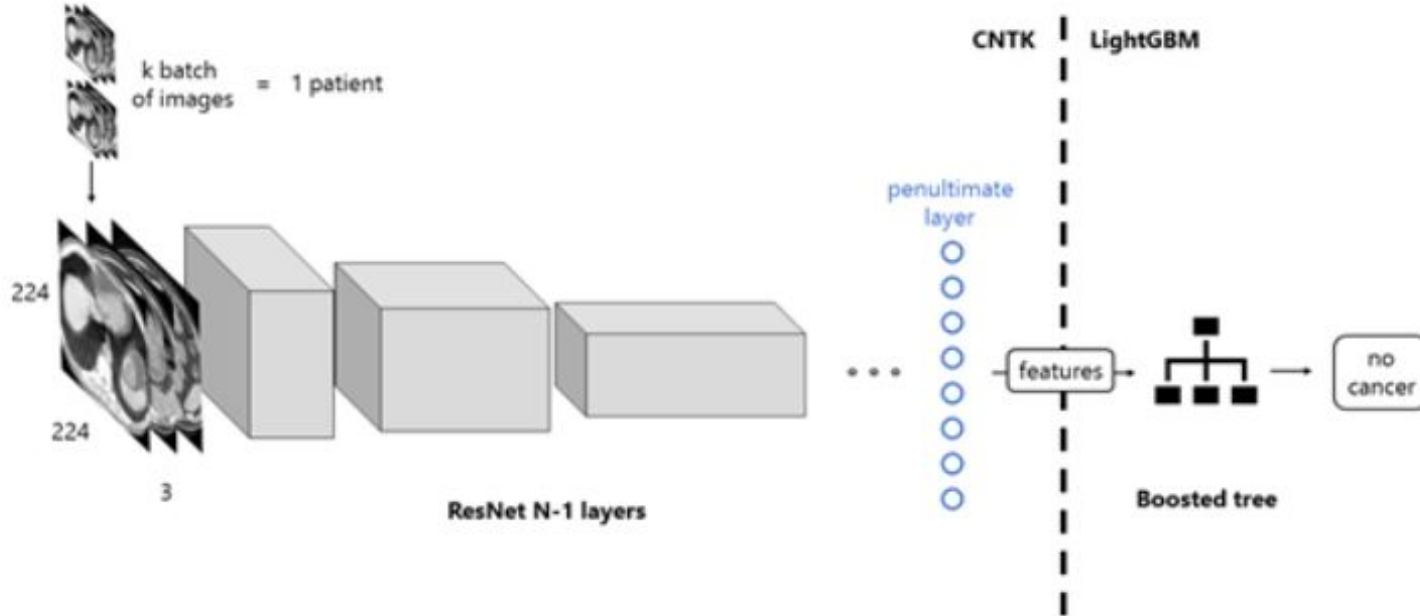
Dataset



Approaches - Basic CNN Architecture



Approaches - Extract Features



Approaches - Train classifiers

- **LightGBM**
 - Mean features
 - Label every image, use all features
 - Combine two pre-trained model features
- **LSTM**


Approaches - Train LightGBM (ResNet50)

<u>ResNet50</u>	Approach 1	Approach 2
Train Data Shape	(1397, 2048)	(81942, 2048)
Train Time (seconds)	577.780	5299.740
Stopping Rounds	3164	5000
Final Train Log Loss	0.435	0.414

Approaches - Train LightGBM (VGG19)

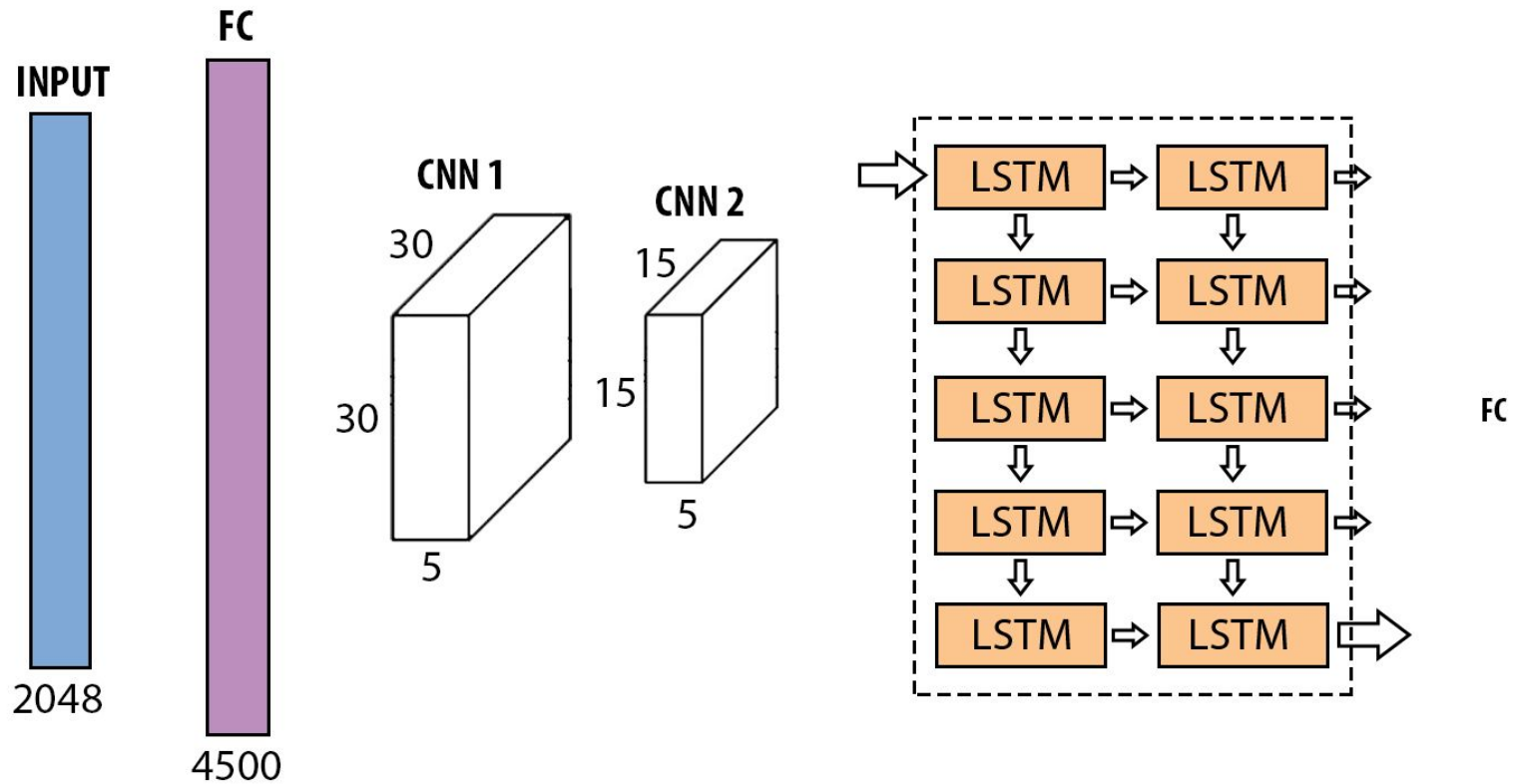
<u>VGG19</u>	Approach 1	Approach 2
Train Data Shape	(1397, 25088)	(81942, 25088)
Train Time (seconds)	6419.99	46979.59
Stopping Rounds	2890	5000
Final Train Log Loss	0.4337	0.4026

Approaches - Train LightGBM (combine two)

Combine Two	
Train Data Shape	(1397, 27316)
Train Time (seconds)	6574.270 
Stopping Rounds	3553
Final Train Log Loss	0.436

Approaches - Prediction

- Using LightGBM
- **Using LSTM**



Max Pooling with Stride 2
Leaky Relu
Batch Normalization

Model Parameters

Optimizer: Adam

Learning rate of 0.001, momentum of 0.9

Initialization: Glorot normal (aka Xavier normal)

fan in = number of input units in the weight

fan out = number of output units

$$\mathcal{N}\left(0, \sqrt{\frac{2}{fan\ in + fan\ out}}\right)$$

Loss: LogLoss on Softmax

“High-mean” layer

Motivation:

Cancerous nodes only on some slices

RNN not working well

Max, Mean not sensitive

Get mean of slices whose mean is above the patient's mean

Experiments and Results

- Evaluation Metrics
 - Log Loss

$$\text{LogLoss} = -\frac{1}{n} \sum_{i=1}^n [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)],$$

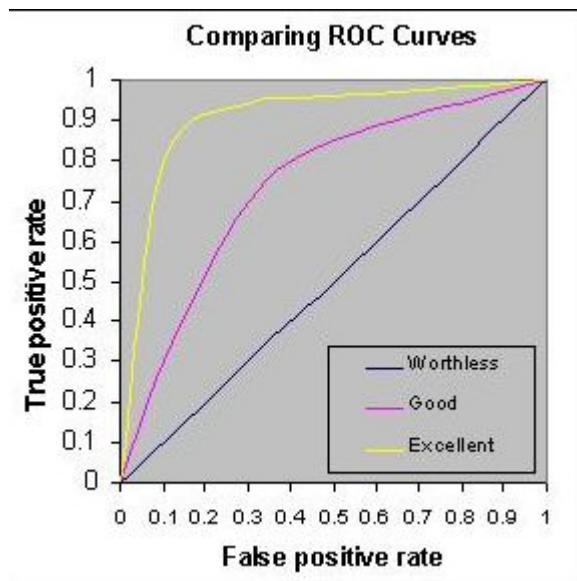
where

- n is the number of patients in the test set
- \hat{y}_i is the predicted probability of the image belonging to a patient with cancer
- y_i is 1 if the diagnosis is cancer, 0 otherwise
- $\log(\cdot)$ is the natural *base e* logarithm

Note: the actual submitted predicted probabilities are replaced with $\max(\min(p, 1 - 10^{-15}), 10^{-15})$. A smaller log loss is better.

Experiments and Results

- Evaluation metrics
 - ROC curve



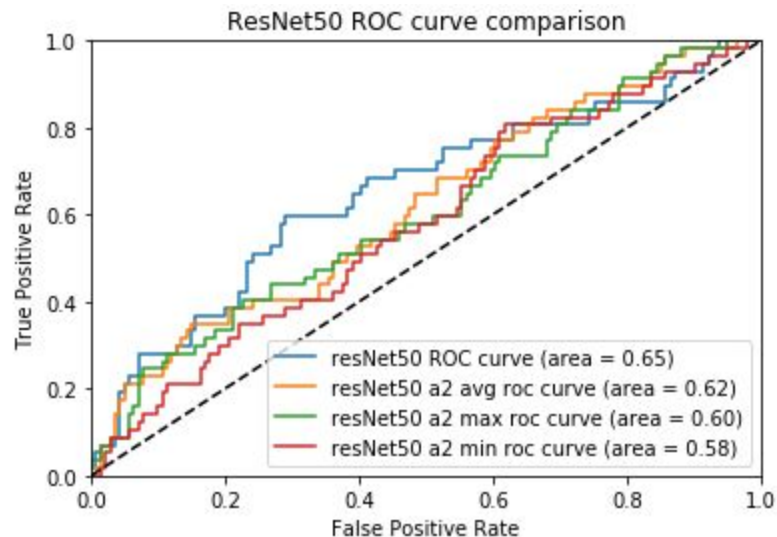
Experiments and Results

- Results - Log Loss Comparison

Model		Log Loss	
		Train dataset	Val dataset
VGG19_1		0.434	0.557
VGG19_2	Avg	0.403	0.582
	Max		0.773
	Min		0.609
<u>ResNet50_1</u>		0.435	0.576
<u>ResNet50_2</u>	Avg	0.414	0.585
	Max		0.617
	Min		0.623
Combine Two model		0.436	0.560

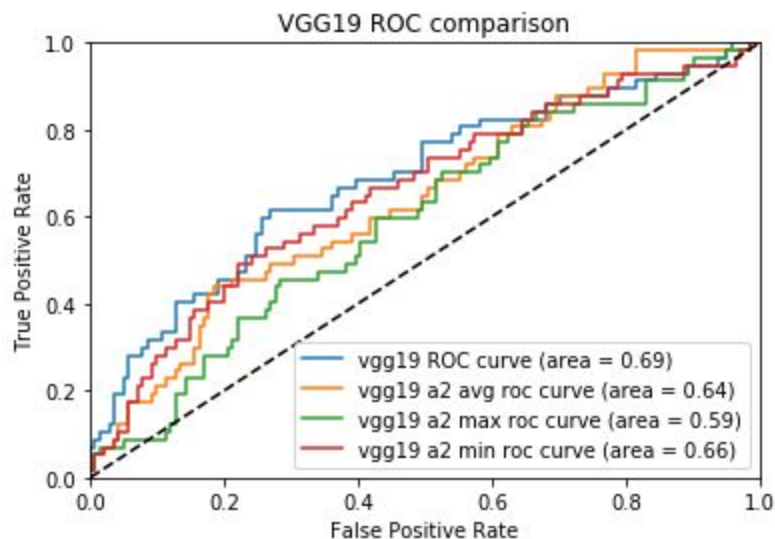
Experiments and Results

- Results - ROC curve comparison (ResNet50)



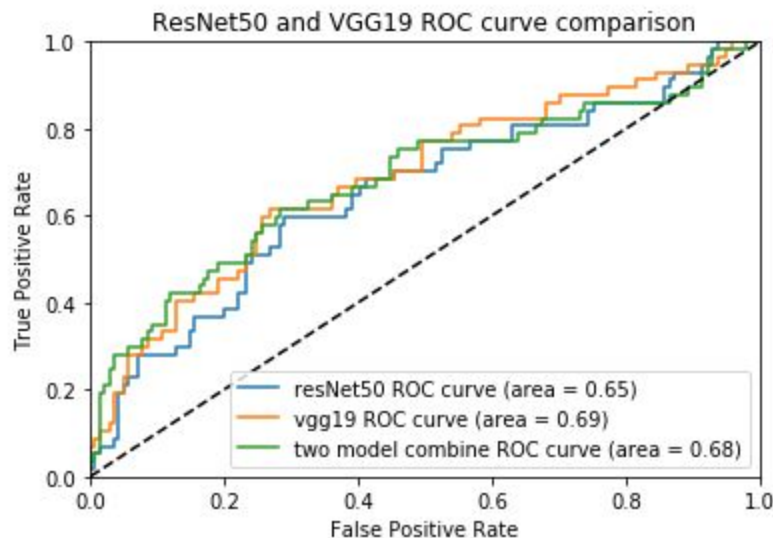
Experiments and Results

- Results - ROC curve comparison (VGG19)

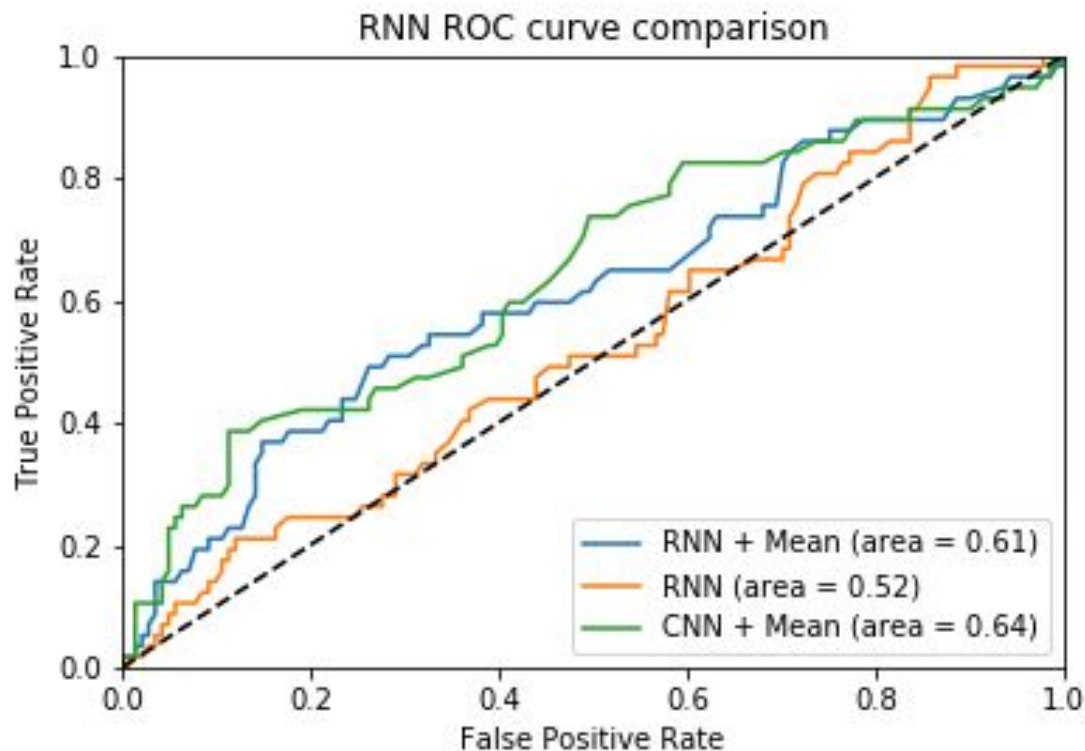


Experiments and Results

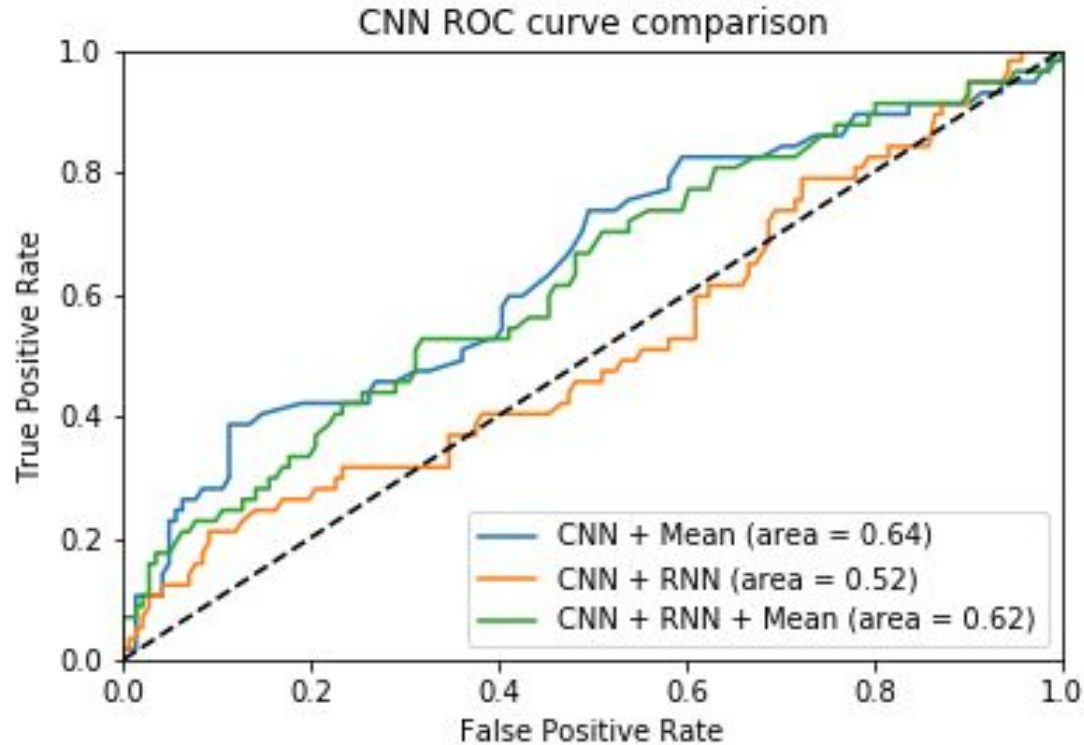
- Results - ROC curve comparison (ResNet50 and VGG19)



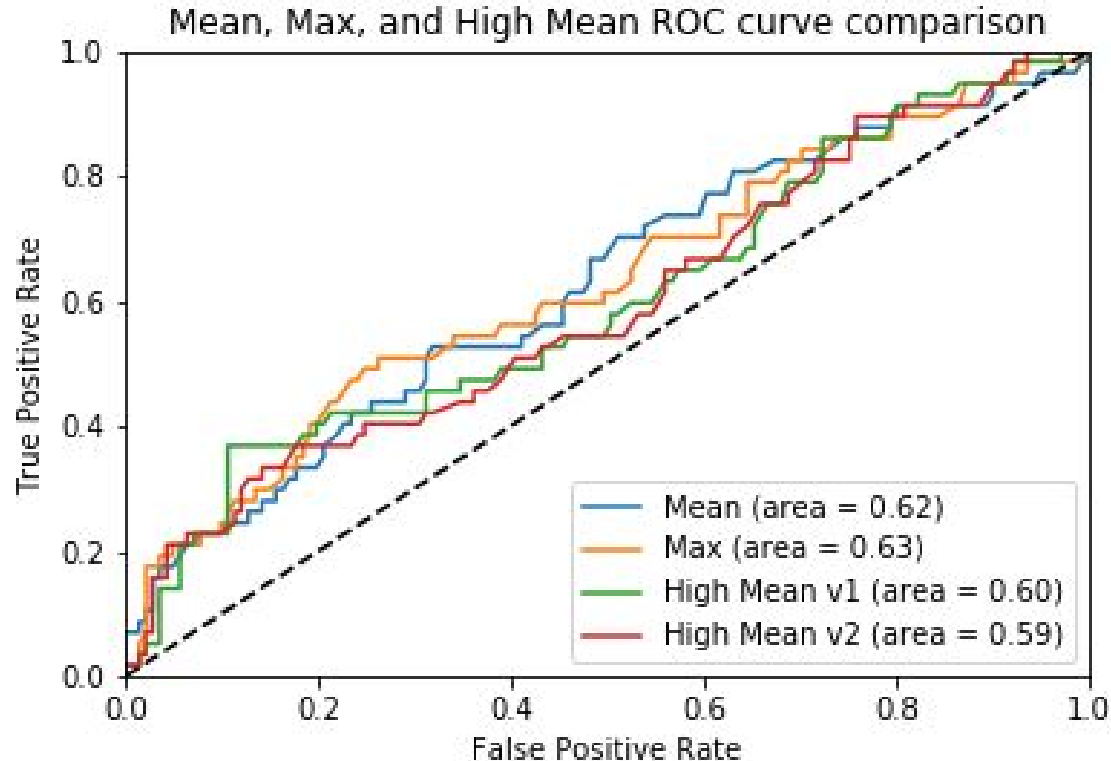
Experiments and Results - Approach 2



Experiments and Results - Approach 2



Experiments and Results - Approach 2



Model	LOSS	AUC
cnn + mean	2.09	0.64
cnn + rnn	3.06	0.52
cnn + rnn + mean	3.17	0.623
cnn + rnn + highmean v2	9.49	0.59
cnn + rnn + highmean v1	9.94	0.6
cnn + rnn + max	2.93	0.63
rnn + mean	2.77	0.61
rnn	2.9	0.53
data combo, cnn+rnn+mean	2.66	0.57

Conclusions

Transfer learning was successful

RNN performance was disappointing

Sequence too long?

Performance wall?

Easy to overfit

Future improvements

Ideas:

Up/Down-sample to fixed number of slices

Bucketing

More network after RNN (requires fixed length)

More data to help overfitting

References

- [1] Ramaswamy and Truong. "Pulmonary Nodule Classification with Convolutional Neural Networks." (2016).
- [2] Anirudh, Rushil, et al. "Lung nodule detection using 3D convolutional neural networks trained on weakly labeled data." *SPIE Medical Imaging*. International Society for Optics and Photonics, 2016.
- [3] Fierro, Miguel, Ye Xing, and Tao Wu. "Quick-Start Guide to the Data Science Bowl Lung Cancer Detection Challenge, Using Deep Learning, Microsoft Cognitive Toolkit and Azure GPU VMs." Blog post. Cortana Intelligence and Machine Learning Blog. Microsoft, 17 Feb. 2017. Web. 10 Mar. 2017.
- [4] Ypsilantis, Petros-Pavlos, and Giovanni Montana. "Recurrent Convolutional Networks for Pulmonary Nodule Detection in CT Imaging." arXiv preprint arXiv:1609.09143 (2016).