CS230

Histopathologic Cancer Detection of Lymph Node Patches for Breast Cancer

Jad Aboudiab
jadab@stanford.edu

Gowtham Gundu
gowthamg@stanford.edu

Abstract

Manual pathology classification is a time consuming task that requires deep domain expertise by medical board certified pathologists. It can also be an expensive task to have pathologists spend time on classifying even the simplest positive cases, but also the extra time required to spot the difficult cases. Assistive image classification for histopathologic scans that achieves human like or better quality has become an area of deep interest within the medical industry, in the last several years. This work explores various deep neural network based techniques to come up with a robust way to build a high recall assistive system to detect metastatic cancer by analyzing small hematoxylin and eosin (H&E) stained patch images of lymph node sections. We use a publicly available Kaggle dataset[1] to train, evaluate, and compare various techniques.

1 Introduction

Manual analysis of metastatic lymph node staging by a single medical professional, tends to be a very time consuming task with increased complexity around rare forms. Studies have shown that about 25% of classifications would be changed upon second pathologic review [1]. Under time constraints detection sensitivity of small metastases on individual slides can be as low as 38% [2]. Prior experiments that provided pathologists with and without the assistance of metastatic detecting DNNs, rendered positive results that made the task easier and halved average slide review time[2]. Narrowing down to regions of interest not only reduces the time but also increases the accuracy of labeling by pathologists.

This project explores various ways to train and tune deep neural networks (DNNs) to come up with a robust way to build these networks that can act as assistive systems with high recall on capturing positive regions. While reaching human-level performance or better is a valuable and ambitious goal, having a process to build DNNs that can identify regions of interest with extremely high recall is also very valuable as these systems become more mature and widely accepted.

2 Related work

Several promising studies have applied deep neural networks to histopathological cancer detection in the last few years. Starting with Camelyon16 [3] challenge, where several participants including the winners have trained various CNN models, ensemble techniques, later followed by others like Google Brain [4], that were able to achieve recall above 92% on same Camelyon16 dataset. Most

[1] https://www.kaggle.com/c/histopathologic-cancer-detection/data
[2] https://insights.ovid.com/crossref?an=00000478-201812000-00007

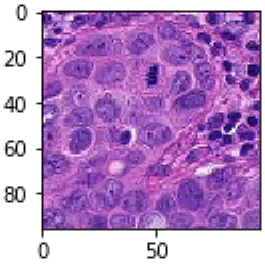
of these studies used Inception or VGG16/19 kind-of architectures at the core and experimented with various input image sizes, positive/negative example distribution, data augmentation, combining with hand built model outputs etc.

3 Dataset and Features

This work uses the publicly available Kaggle dataset that is very similar to the original PCam dataset [5], with difference that duplicates are removed. The dataset has 220K labeled images each of which is of size 96x96, with the center 32x32 region containing at least one pixel of cancer cells when image is labeled positive. The slides are stained with H&E, a common procedure performed by pathologists. All details are described in table 3.a and a positive example in 3.b.

Training Instances	220,025 (60% negatives)
Size	96x96 (32x32 has target content)
Channels	3 (8 bits per channel)

Table 3.a Dataset details



3.b Positive instance

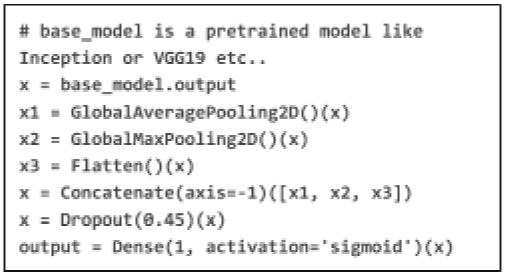
4 Our work

Our work explored several DNN models, starting with a simple DNN that has few fully connected hidden layers with an output layer that is sigmoid since we are dealing with binary classification problem. The simple model produced a low accuracy of 68% on test set and 69% on train set, clearly highlighting the need for more deeper models.

4.1 Transfer learning

Our work explored transfer learning on various standard network architectures, starting with simple ones like LeNet that has just 2 convolution layers followed by 3 fully connected layers. This produced quality improvements over the previous simpler hand built networks that didn't have convolution layers. Then we explored transfer learning on more complex pretrained models like VGG16, VGG19, InceptionV3 and Resnet. More or less, at the end, all of these models were able to achieve similar results with hyperparameter tuning. Final results are described in Section 5.

One of the key factors that impacted the quality of the models is the way the new fully connected (FC) layers are added to the pretrained networks. Out of the various possibilities, the clear winner is the approach where most amount of information is passed from base model to the FC layers. For example, just doing a max pooling or average pooling from the output of the base model and feeding to the first newly added FC layer, underperforms when compared to concatenating both max and average pooled outputs to next layer. Even stronger results were achieved by flattening and concatenating at the end. See code snippet in Figure 4.1.b.



```
# base_model is a pretrained model like
Inception or VGG19 etc..
x = base_model.output
x1 = GlobalAveragePooling2D()(x)
x2 = GlobalMaxPooling2D()(x)
x3 = Flatten()(x)
x = Concatenate(axis=-1)([x1, x2, x3])
x = Dropout(0.45)(x)
output = Dense(1, activation='sigmoid')(x)
```

Figure 4.1.b Connecting base model output to newly added FC layers

4.2 Model Architecture and Training

After building upon several model architectures, we chose the VGG19 model as the base model for our final candidate, with Imagenet preloaded weights. Our final model has 28 layers of which

17 are trainable layers with about 20,030,017 trainable parameters. The model architecture is shown in Figure 4.2.a.

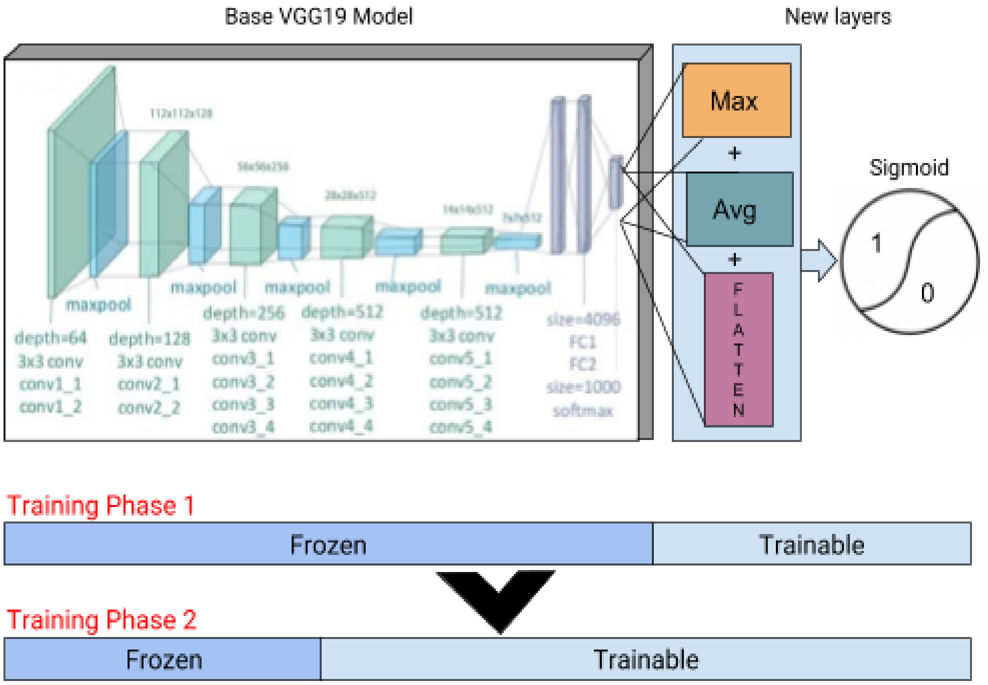


Figure 4.2.a Simple illustration of the network architecture of base VGG19 model [6] with newly added layers for final candidate. Actual inputs for our model were 96x96x3. Bottom of the image also demonstrated how the training was done in multiple phases.

We trained the model in 2 phases, with the first phase min-batch trained on just the newly added (last 5) layers, using Adam optimizer. The second phase is trained on more layers (last 14), but with Stochastic Gradient Descent and a smaller learning rate to further reduce the loss. The second phase almost achieves an accuracy of 99.9% on training set after few epochs, though it starts overfitting after 98% accuracy.

We performed several data augmentations to help the model generalize better. We used elastic deformations [7], rotations, flips, and changes to brightness. The augmentations provided good wins early on, but most of the wins were later masked by tuning and other training improvements.

4.3 Model tuning

Our work split the model tuning task into two parts: i) Hyperparameter tuning ii) Threshold tuning.

4.3.1 Hyperparameter tuning

Training of a DNN requires choosing several hyperparameters like learning rate, dropout probabilities, batch size, number of layers to freeze from pre-trained model etc.. Picking the right hyperparameters is important to build a more accurate model with fewer training steps. Our work initially explored commonly chosen defaults for these parameters, but training several models gave better intuition into choosing more optimal hyperparameters. For example, it is clear that the model was overfitting and increasing the dropout (to 0.5) helped better generalize and perform well on validation/test datasets.

Later phases of our work also explored Bayesian Hyperparameter optimization [8], which builds a surrogate probability model of objective function, that is optimized to come up with next set of parameters to try on actual objective function. AUC for ROC is used to guide the surrogate model.

Bayesian Hyperparameter optimization yielded minimal improvements, but on models that were already highly accurate.

4.3.2 Threshold tuning

One of the main goals of our work is to come up with a way to train models that can detect positive regions with highest recall possible. Unfortunately it is quite hard to have a loss function that can optimize for recall. For example, recall itself cannot be optimized, because it is not a smooth function and also the gradients are undesirable. Hence our work focussed on tuning the threshold used to convert probability to class membership. For sigmoid activation function, 0.5 is chosen by default, but is something that can be tuned.

Our tuning procedure involved plotting precision, recall and f1-score of positives and negatives on validation dataset, then picking a threshold with best trade-off. The chosen threshold can be verified on test data. Figure 4.3.2.a shows a plot for one of our best models. Based on that plot, picking a threshold of 0.12 yields a recall of almost 98% on positives, with minimal trade-off on precision from 95% to 92%. These numbers also hold on the test dataset, demonstrating the strength of this approach.

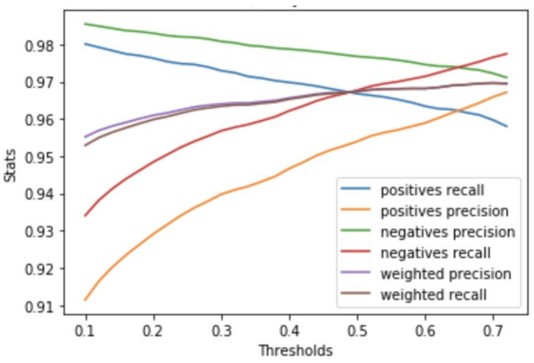


Figure 4.3.2.a. Precision/Recall stats plotted over probabilities, computed on validation dataset.

5 Results

In the end, our work yielded promising results with best performing models having an accuracy of 97% (see Table 5.a) and further we were able to tune them to get to 98% recall on detecting positive cancer regions (see Table 5.b).

Model	Parameters	Train accuracy	Validation accuracy	Test accuracy
VGG19	~143M	98.5	97.2	97.1
VGG16	~138M	98.5	96.3	96
Resnet50 [9]	~26M	99.6	96.7	96.8
InceptionV3 [10]	~23M	90	89.7	89.8
MobilenetV2	~3.5M	98.9	96.7	96.5
Lenet	~60K	77	77.6	77.7
MLP[3]	700K	69	68	68.2

Table 5.a Different models trained on dataset of size 220K, with 80/10/10 split for train/validation/test. Each image of size 96x96 pixels, with 3 channels.

label	precision	recall	f1-score	num_instances
0	0.98	0.94	0.96	13179
1	0.92	**0.98**	0.95	8824
avg/total	0.96	0.96	0.96	22003

Table 5.b Tuned model has Recall of 98% on positives, with 92% precision, on test set.

To verify the actual predictions, we generated saliency and occlusion maps on a random sample of

[3] Multi layer perceptron model with no convolutions

true positives and true negatives. We consulted a Board Certified Anatomic and Clinical Pathologist, Aida K. Rechdouni M.D, who reviewed these samples and graded the quality of the predictions. Dr. Rechdouni first graded the 10 samples herself, without seeing the model predictions, and then analyzed the predictions providing a summary of each one. For 5 out of 7 true positive samples, the model picked the exact regions in which Dr. Rechdouni labeled as metastatic cancer. For the remaining 2 positives, the model still predicted accurately, though it didn't comprehensively identify all the metastatic cancer within the slide. For the 3 out of 3 negative samples, the model predicted accurately though also didn't comprehensively identify all the benign cells. In total, the model agreed with Dr. Rechdouni on all 10 random examples.

In Figure 5.c, we present one of the true positives analyzed by Dr. Rechdouni. This particular slide has a combination of benign and metastatic cells. Large cells, with enlarged nuclei and pink ample cytoplasm were heavily present in the upper region of the photo. The model was highly activated around these large cells which are much larger than the surrounding lymphocytes, indicating positive metastatic cancer.

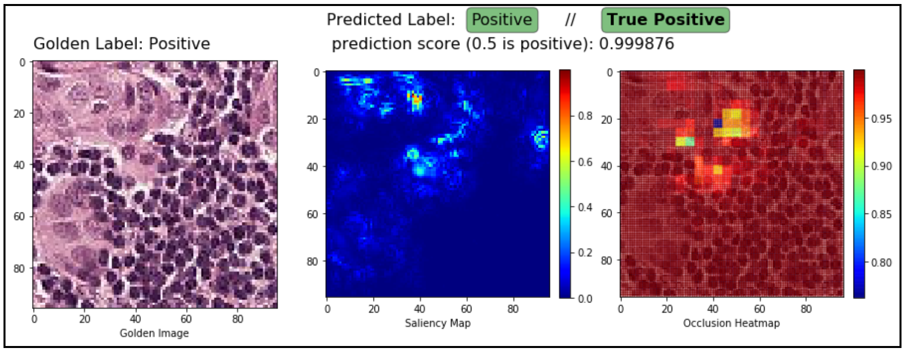


Figure 5.c A true positive random sample analyzed by Dr. Rechdouni.

In Figure 5.d we present a true negative also analyzed by Dr. Rechdouni. This slide showed lots of normal shaped lymphocytes without any indication of metastatic cancerous cells. The model found the left and bottom portions of the image most interesting for it to make decision. Dr. Rechdouni notes that any region of this photo would have been correct, given that all the cells are benign lymphocytes.

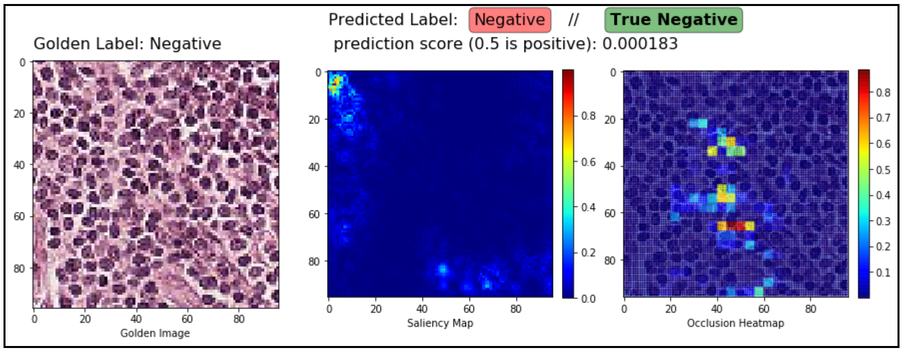


Figure 5.d A true negative random sample analyzed by Dr. Rechdouni.

6 Conclusions and Future Work

Our work prescribes a robust way to train and tune DNN models that can assist pathologists by helping them narrow down to the fewer but highly probable areas of interest, with a very little risk of missing a positive region. Even if small, because of risk involved in the false negatives, accomplishing 100% recall is ideal. Useful future work would be to explore active learning from false negatives on validation dataset to further improve the recall and verify the improvements on test set. Also there could be sampling biases that can be addressed so that these models continue to produce very high recall results on unseen data in the wild. For example, the dataset could be biased to fewer slides or fewer patients, hence understanding that distribution and adjusting the sampling of train/validation/test datasets can really help (though unfortunately this would require more detailed dataset).

7 Contributions

Jad and Gowtham trained over 150 different models, with each focussing on different architectures initially. Later Gowtham focussed on Hyperparameter tuning and Threshold tuning while Jad focussed on Data Augmentation, Heatmaps and verifying the models with inputs from a Board Certified Anatomic and Clinical Pathologist.

8 Code

Our code can be seen and downloaded from https://github.com/unjadded/cs230_histo_cancer**.** We forked and built up on an open source library for generating saliency and occlusions [11].

9 Credits

Special credits to Aida Rechdouni, M.D (Board Certified Anatomic and Clinical Pathologist) for analyzing predictions on a random sample, to help understand the correctness of the models.

10 References

[1] Vestjens JH, Pepels MJ, De boer M, et al. Relevant impact of central pathology review on nodal classification in individual breast cancer patients. Ann Oncol. 2012 https://www.ncbi.nlm.nih.gov/pubmed/22495317

[2] Diagnostic Assessment of Deep Learning Algorithms for Detection of Lymph Node Metastases in Women With Breast Cancer:https://jamanetwork.com/journals/jama/fullarticle/2665774

[3] 270 Whole-Slide-Images of lymph node sections for breast cancer: https://camelyon16.grand-challenge.org/Data/

[4] Yun Liu, Krishna Gadepalli, Mohammad Norouzi, George E. Dahl, Timo Kohlberger, Aleksey Boyko, Subhashini Venugopalan, Aleksei Timofeev, Philip Q. Nelson, Greg S. Corrado, Jason D. Hipp, Lily Peng, Martin C. Stumpe "Detecting Cancer Metastases on Gigapixel Pathology Images"

[5] https://github.com/basveeling/pcam

[6] Yufeng Zheng, Clifford Yang, Alex Merkulov "Breast cancer screening using convolutional neural network and follow-up digital mammography", Proc. SPIE 10669, Computational Imaging III, 1066905 (14 May 2018); doi: 10.1117/12.2304564; https://doi.org/10.1117/12.2304564

[7] https://www.kaggle.com/ori226/data-augmentation-with-elastic-deformations

[8] Tinu Theckel Joy, Santu Rana, Sunil Gupta, Svetha Venkatesh "Fast Hyperparameter Tuning using Bayesian Optimization with Directional Derivatives" https://arxiv.org/abs/1902.02416

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Microsoft Research "Deep Residual Learning for Image Recognition" https://arxiv.org/pdf/1512.03385.pdf

[10] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna "Rethinking the Inception Architecture for Computer Vision" https://arxiv.org/abs/1512.00567

[11] https://www.kaggle.com/blargl/simple-occlusion-and-saliency-maps