CS230

Learning Histology Slice Processing Steps

Fang-Yu Lin[*]
Department of Electrical Engineering
Stanford University
fangyuln@stanford.edu

Abstract

We want to explore the possibility of learning histology slices processing in order to accelerate the process of utilizing them in future tasks, such as image registration and cancer prediction. Normally, professionals have to spend a few minutes to process one slice from each patient so that the slices can be automatically processed by computers since the histology slices are often rotated and flipped when capture. We would like to purpose an end-to-end algorithm to avoid this time consuming process. In this project, we explore different pre-trained model to do orientation classification and got our best result from ResNet50 with an accuracy of about 80% on validation set. Although the order of slices, patients variety will have an impact on our prediction results, the model we have now can already accelerate the processing step. We have to figure out how could we generalize our model in the future.

1 Introduction

The purpose of this project is to eliminate the manual process when we are utilizing histology data. We are applying deep learning to histology slice processing. Histology slice is a type of invasive medical imaging with very high-resolution which can clearly indicate the existence of cancer cells. If we want to diagnose cancer from non-invasive data, such as MRI, with the deep learning approach, it would be beneficial to add histology information to it. To fuse the MRI and histology data, we need to do image registration first. However, the raw data of histology slices are often rotated with large angles and flipped. Computers can't apply image transformation automatically as a doctor can. If we want to rotate an image to a correct orientation manually, it might take a few minutes for just one slice. As a result, it is necessary to learn how to make the histology slices usable for further research tasks, such as the image registration and diagnosis mentioned above. By applying correct rotation and horizontal/vertical flip automatically with deep learning, we can make use of the histology data in the image registration task efficiently. Thus, an end-to-end medical imaging data fusion pipeline will be constructed through this project. Due to the time limit nature of this project, we will focus more on image orientation correction. To be specific, the input to our algorithm is RGB images with TIFF format. We then use different types of neural network to output a predicted orientation.

In contrast to consumer photography, there is no specific medical imaging orientation classification yet, and that is our main task – to explore the possibility of automate the histology processing steps.

2 Related work

Past studies of image orientation with neural networks can be divided into two categories: 1) image orientation classification and 2) image skewing regression. For image orientation classification tasks,

[*]Link to repository: https://github.com/lfangin/CS230

Joshi(6) applied a pre-trained VGG-16 model(10) and added one layer to its output and classified them into 4 class of multiples of 90 degrees and got a result better than the state-of-the-art image orientation algorithms then. Also, Senn(9) wrote a github repo called "RotNet" to discuss this topic with consumer photos from large dataset using ResNet50(5). Both of them use transfer learning to overcome the issue of not enough data. Because we have a much more smaller dataset, we will also apply transfer learning to improve our results. Fischer(4) used the AlexNet(7) pre-trained model and regression output to identify the rotation angle. Fischer has done experiments on rotation angles within 30, 45, and 180 degrees. The author made a conclusion that the result of 180-degree experiment was unsatisfactory. Therefore, image skewing might not be a good choice for us since we are trying to deal with large rotation angles.

Base on previous research, there are two challenges in this project. First, there isn't a lot of data to train the model. Since our dataset is from the hospital and has to be processed by professionals, there are a relatively small amount of data ready to use. Second, as we plan to use a pre-trained model, the features of consumer/natural photos may not apply to medical imaging because medical imaging is very different from natural photos. The color of our histology slice images are monotonous and they got similar shape for every slices.

3 Data

3.1 Dataset

The dataset is from the Laboratory for Integrative Personalized Medicine (PiMed), Department of Radiology. It contains 38 cases of prostate cancer data, each of them with 5 - 10 histology slices. A total of 258 slices is now in the dataset. Each of the images has a resolution of roughly 5k x 5k pixels. All of them are annotated with their orientation, so we can manipulate the dataset to generate images with different orientations. Table1 has an example image of the input data. Since we have a small amount of dataset, we will only divide our data set to training and validation set, with a ratio of 80-20%. The data from the same patient might be very alike. To avoid the problem of testing on training set, we will apply per-patient training, which means that the histology slices from one patient will be assigned to either training or validation set. This leads to a training set of 31 patients' and testing set of 7 patients' histology slices.

3.2 Preprocessing and Augmentation

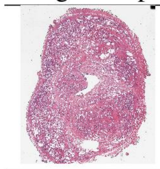
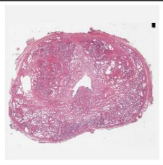
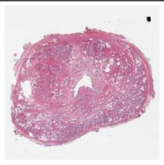
Since we want to use pre-trained model's weights on ImageNet, the size of images after preprocessing will be $244 \times 244 \times 3$. To fit this size without changing the shape, we first need to pad the original image to make it a square. Also, the prostates in original images are sometimes very close to the edges, therefore, we add a default value of 300 pixels around all four edges in every images. Rather than applying white or black RGB values for padding, we use a gray color of (R,G,B) = (242,242,242) because it fits our backgroud area most. Second, before we down-sample the images, we will apply a Gaussian blurring with a kernel size of 101×101 and $\sigma = 5$ to prevent aliasing. We then apply down-sampling so that the resolution can fit our pre-trained model. Table1 has an example after our preprocessing step.

Because of the relatively small amount of data, we will duplicate one histology slice image into 4 different rotation angles (0, 90, 180, 270 degree) and the mirror of them as our data augmentation strategy. The number of duplicated images will depend on the number of classes in our task. Also, we rotate images by a small angle of 5 degree so that it is not perfectly matching a class; for example, if you put an image rotated by 95 degrees into the dataset, it should be classified as 90 degrees in a 4 classes classification task. After data augmentation, one original slice will be duplicated as 24 versions of itself. A total amount of 4944 images in training set and 1248 images in validation set. Table1 has an example after our augmentation step. All of our preprocessing and augmentation steps are done by OpenCV(2).

4 Methods

Since we are using a relatively small dataset, transfer learning is a good starting point. Previous research showed that both models of VGG-16(6) and ResNet50(9) pre-trained on ImageNet perform

Table 1: Dataset Example Images

Original Input	Preprocessing	Augmentation
		

well on consumer/natural photo orientation classification. We will test these two pre-trained models first, and then try different method to improve them.

The loss function of our classification task will be Cross-entropy

$$-\sum_{c=1}^{M} = y_{o,c}log(p_{o,c})$$

where M is the number of classes, which is 4 in this case; y is the binary indicator if class label c is equal to our classification observation o here; p is the predicted probability observation o of class c. For this supervised learning task, the evaluation metric of our results is accuracy. All of the implementation are done with Keras(3) with Tensorflow backend(1).

4.1 ResNet50 and Classification

ResNet50 is commonly used architect that can train a deep neural network very efficient. We use a ResNet50 model pre-trained on ImageNet and add one output layer onto it. In addition, we freeze all weights in ResNet50 in order to accelerate the first experiment. After the preliminary result is obtained, we then pick two strategy to improve our model: regularization and reduce frozen layers. We will show the result of L2 regularization (L2(λ)) and Dropout (Dropout(rate)) for regularization. For different number of frozen layers, we will show the result of freezing all layers(FA) and freeze to the Activation 9 layer(F9). We choose Adam optimizer with a learning rate of 0.01, batch size of 128 for training.

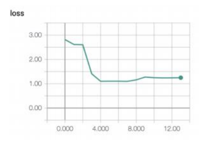
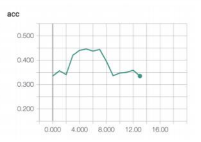
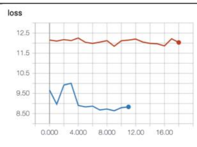
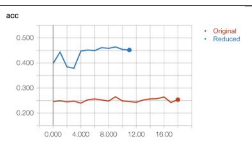
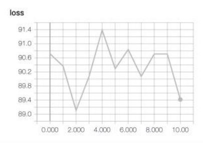
4.2 VGG-16 and Classification

We test the same architect as Jushi(6) described. We trained on a VGG-16 pre-trained model with the exact same structure but only with 4 nodes at output. After reducing the two densed layers before output and testing on different hyperparameters, no comparable result is obtained. The unsuccessful loss and accuracy plot can be found at Table2.

4.3 Resnet50 and Regression

Although previouse research suggested that large rotation angle tasks are not highly applicable for regression tasks. We still test a ResNet50 pre-trained model with regression output for comparison usage as Senn(9) suggested. However, by the time of deadline, no comparable result is obtained. The unsuccessful loss plot can be seen in Table2.

Table 2: Failure Method

ResNet/classification(all trainable)		VGG-16/classification		ResNet50/regression
				

5 Experiments and Results

We will seperate this part into two, first, the experiments of different models that we tried and their performance and second, analysis of the best model we obtained.

5.1 Experiments

As we discussed in previous section, we tried different models based on its number of frozen layers(FA and F9) and different regularization methods(Dropout and L2). Note that other hyperparameters that we are not discussing here, such as, learning rate and batch size are the optimal selection based on a lot more preliminary experiments that are not shown here due to limited space.

As we can see from Table3 and Figure2, the validation accuracy of F9 are higher than FA's. Note that these result is obtained by an additional rotation during training. That is, although we already have every rotation angle for every slices, we still add one more random rotation to it. Base on the results in Table3, we also tried to train with only random selection and without the random rotation with F9 models. Similar results are shown. Because the accuracy without another random rotation is higher, we will evaluate this model later in this section.

Table 3: Training and Validation Accuracy (w additional random rotation)

Model	FA	FA+L2(0.1)	FA+L2(0.2)	F9	F9+Dropout(0.9)	F9+L2(0.1)	F9+L2(0.1)+Dropout(0.9)
Training	0.9711	0.8966	0.8691	0.9986	0.8701	0.9464	0.9335
Validation	0.6402	0.6282	0.5921	0.7556	0.7284	0.6490	0.5777

Table 4: Training and Validation Accuracy (w/o additional random rotation)

Model	F9	F9+Dropout(0.9)	F9+L2(0.1)
Training	0.9593	0.9027	0.9351
Validation	0.7989	0.6691	0.7075

5.2 Evaluation and Results

The general testing We did the evaluation implementation with skikit-learn(8). Table6 shows the confusion matrix and Table7 shows the classification report with Precision, Recall and F1 score. Also, since each patients' histology slices are with high consistency, we also provide Table5 to show the predicting accuracy among each patients. Never the less, since histology slices are generally harder to recognize orientations in the first and last slices, we also draw Figure1 to provide a view of how that effect our result. A quality result are provided in Table8, where we illustrate our input data with three different quality of results, which we will discuss in the next section.

Table 5: Per Patient Accuracy

Patient #	32	33	34	35	36	37	38
Accuracy	0.417	0.912	0.979	0.859	0.903	0.646	0.923

Table 6: Confusion Matrix Table

True/Predicted	0	90	180	270
0	210	13	42	22
90	3	240	10	38
180	5	6	227	50
270	5	3	10	268

6 Discussion

From Table3, 4 and Figure2, we can see that our model suffers from overfitting. Our small dataset can lead to a overfitting result like this. However, the regularization method we are using here does not improve out validaiton set accuracy.

Table 7: Classification Report

True Angle	Precision	Recall	F1-Score
0	0.94	0.73	0.82
90	0.92	0.82	0.87
180	0.79	0.79	0.79
270	0.71	0.94	0.81
Ave	0.84	0.82	0.82

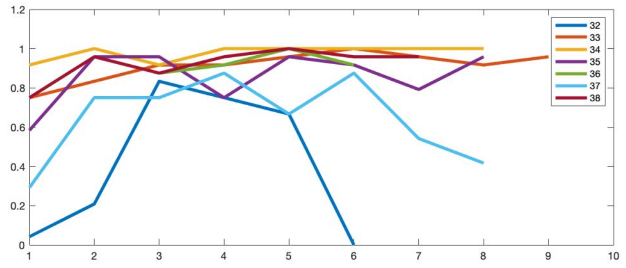


Figure 1: Accuracy vs. # of slice (per patient)

For Table6 and 7, we can see that the precision and recall have an average above 0.8 and that could support our claim that this model can accelerate histology slice processing. No specific patter of error could be found here.

Table5,8 and Figure1 can be view together as error analysis. We can see that the data quality for patient 32 is very bad and can only be predicted with an accuracy of 0.41. The histology slice is broken and it could be the reason of inaccurate results. In contrast, from the first line of Table8, the shape of prostate is very clear and complete. The third row shows that not all labels are accurate and when we only have four classes, a slice we 45 degree of ratation can have b low quality of prediction.

7 Conclusion and Future Work

We have our best performing model from Resnet50 with freezing layers up to the Activation 9 layer. From previous sections, we noticed that our model can predict the rotation angle with an accuracy of about 80%. But we also noticed that our model is highly overfitting and should purpose other method for regularization. Our predicting quality depends on the original image quality a lot. To sum up, our model can accelerate the histology processing steps with a good quality of images.

For future work, we have three direction here. First, Collect more data to avoid overfitting. Second, try different models that wasn't able to try because of the limitation of time and computation resources. Also, different Cost function could be purposed after in-depth error analysis. Lastly, to complete this pathology fusion pipeline, we need to implement a model for image mirroring too.

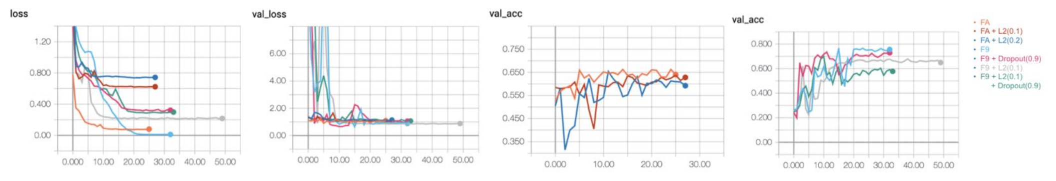
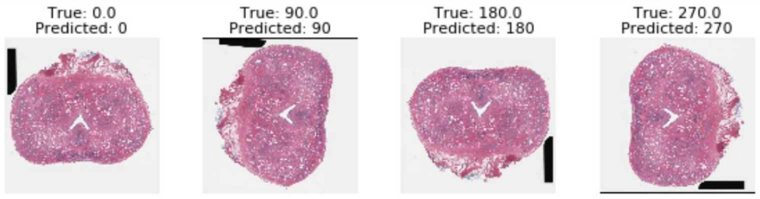
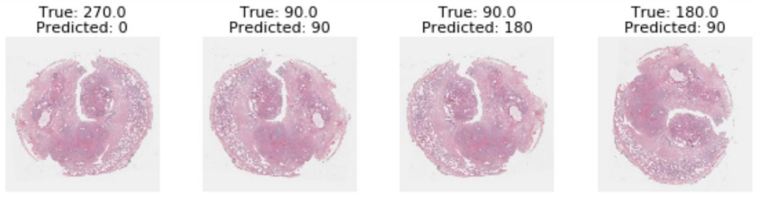
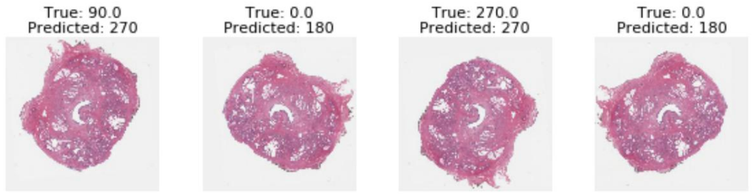


Figure 2: Training, Validation Loss and Validation Accuracy of all Model

Table 8: Example slices and classification results with different accuracy

Slice #, Patient #	Example Results
(256,38), acc=1.0	 True: 0.0 Predicted: 0 True: 90.0 Predicted: 90 True: 180.0 Predicted: 180 True: 270.0 Predicted: 270
(207,32), acc=0.04	 True: 270.0 Predicted: 0 True: 90.0 Predicted: 90 True: 90.0 Predicted: 180 True: 180.0 Predicted: 90
(244,37), acc=0.29	 True: 90.0 Predicted: 270 True: 0.0 Predicted: 180 True: 270.0 Predicted: 270 True: 0.0 Predicted: 180

8 Contributions

This is a personal project with no teammate. Also, the project is from the Laboratory for Integrative Personalized Medicine (PiMED), where my advisor Prof. Mirabela Rusu also helped me through the whole project.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[3] F. Chollet et al. Keras. `https://keras.io`, 2015.

[4] P. Fischer, A. Dosovitskiy, and T. Brox. Image orientation estimation with convolutional networks. In *German Conference on Pattern Recognition*, pages 368–378. Springer, 2015.

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[6] U. Joshi and M. Guerzhoy. Automatic photo orientation detection with convolutional neural networks. In *2017 14th Conference on Computer and Robot Vision (CRV)*, pages 103–108. IEEE, 2017.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[9] D. Saez. *RotNet*, 2017.

[10] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.