1. Introduction and Context

Lung cancer is the second most common cancer and the leading cause of cancer death worldwide. More people die of lung cancer than colon, breast, and prostate cancers combined. Despite this fact, lung cancer receives the least amount of federal research funding compared to other cancers. The chance any given man will develop lung cancer in his lifetime is roughly 1 in 15. For women, the chance is 1 in 17. This includes both smokers and nonsmokers, though smokers have the greatest risk.

Machine learning and transfer learning have led major breakthroughs in many fields, including medicine. Computer vision models require suitable image datasets, which can be difficult to find and standardize in medicine. The LC25000 image dataset provides 25,000 de-identified, HIPAA compliant, and validated histopathological images for both lung and colon tissue. Given lung cancer is underfunded and under-researched, we think transfer learning in the field of deep learning could make a difference in identifying cancerous lung tissue.

2. Problem Statement

Given histopathological images of lung benign tissue, lung adenocarcinoma, and lung squamous cell carcinoma, can we build and deploy a computer vision model utilizing transfer learning that classifies new histopathological images of lung tissue more accurately than a domain expert?

3. Domain Research

Prior to exploring our dataset, we've engaged in some basic research regarding lung cancer. Domain knowledge will help us better understand our data, make more informed decisions while moving through the data science process, and better interpret our results.

First, we gathered some general statistics on lung cancer. As previously stated, lung cancer is the second most common form of cancer and the leading cause of cancer death worldwide. It is estimated that nearly 237,000 Americans will be diagnosed with lung cancer this year. It is the leading cancer killer of men and women, and the leading cancer killer of every ethnic group. Anyone can get lung cancer, and an estimated 20% were never smokers. It is also the only cancer where victims are routinely blamed as responsible for the condition.

Unfortunately, lung cancer is a racist disease. Despite a lower incidence rate in black Americans, their five year survival rate of 20.7% is lower than non-hispanic white individuals at 22.9%. Black Americans are also under-represented in clinical trials. The percentages for clinical trials participation are roughly 83% for caucasians and only 6% for black Americans. Lung cancer is also a sexist disease. Women who have never smoked are more than twice as likely to get lung cancer as men who never smoked. The five year survival rate among women is only 27%, and 171 women will die from the disease every day.

There are a few risk factors for lung cancer worth mentioning. The first, and number one risk factor for lung cancer is smoking. In the USA, cigarette smoking is linked to 80% to 90% of lung

cancer deaths. People who smoke are 15 to 30 times more likely to develop lung cancer. Interestingly, people who quit smoking have a lower risk than smokers, but still a higher risk than if they had never smoked. Unsurprisingly, secondhand smoke poses a similar risk. Radon, an odorless and tasteless gas, is another risk factor for lung cancer. Radon causes 21,000 lung cancer deaths each year, and one out of every fifteen homes in the USA has high radon levels. Other substances like asbestos, exhaust, and general pollution can also cause this cancer. Finally, like with most diseases, a personal or family history of lung cancer can leave someone at higher risk of developing lung cancer.

Lung cancer begins in the lungs and is most likely to spread first to the lymph nodes before other organs in the body. Lung cancers are typically grouped into two main types, small cell and non-small cell. Small cell lung cancer comprises roughly 10% to 15% of lung cancers. It is the most aggressive of all types and strongly related to cigarette smoking. Non-small cell lung cancer (NSCLC) is the most common lung cancer, accounting for roughly 85% of all cases. NSCLC has three main types, two of which are in our image data set.

Lung adenocarcinoma is the most common primary lung cancer in the United States and comprise up to 40% of cases, falling under the umbrella of non-small cell lung cancer (NSCLC.) Adenocarcinoma of the lung typically starts in the mucosal glands. Lung adenocarcinoma is typically found in the lung periphery, and can even be found in scars or areas of chronic inflammation. Though smokers can develop this type of cancer, it is more often found in non-smokers. Frighteningly, it can look like pneumonia on a chest X-ray, but also tends to have the best prognosis.

Lung squamous cell carcinoma (SCC) is also a type of non-small cell lung cancer (NSCLC.) This cancer tends to develop in the central part of the lung or main airway (left or right bronchi.) Lung squamous cell carcinoma tends to grow more slowly than other forms of cancer, namely adenocarcinoma and small cell lung cancer. SCC accounts for 25% to 30% of all lung cancer cases and shows up more in smokers.

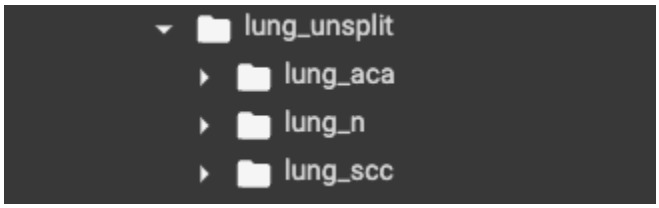
4. Data Source and Procurement

The data we are using was procured through Kaggle. The data is originally the result of work supported with resources and the use of facilities at the James A. Haley Veterans' Hospital. The data was transformed by Andrew A. Borkowski, MD, Marilyn M. Bui, MD, PhD, L. Brannon Thomas, MD, PhD, Catherine P. Wilson, MT,, Lauren A. DeLand, RN, and Stephen M. Mastorides, MD. The LC25000 dataset contains 25,000 histopathological images with 5 classes. All images are 768 x 768 pixels in size and are in jpeg file format. The images were generated from an original sample of HIPAA compliant and validated sources, consisting of 750 total images of lung tissue (250 benign lung tissue, 250 lung adenocarcinomas, and 250 lung squamous cell carcinomas) and 500 total images of colon tissue (250 benign colon tissue and 250 colon adenocarcinomas) and augmented to 25,000 using the Augmentor package.
There are five classes in the dataset, each with 5,000 images, being lung benign tissue, lung adenocarcinoma, lung squamous cell carcinoma, colon adenocarcinoma, and colon benign

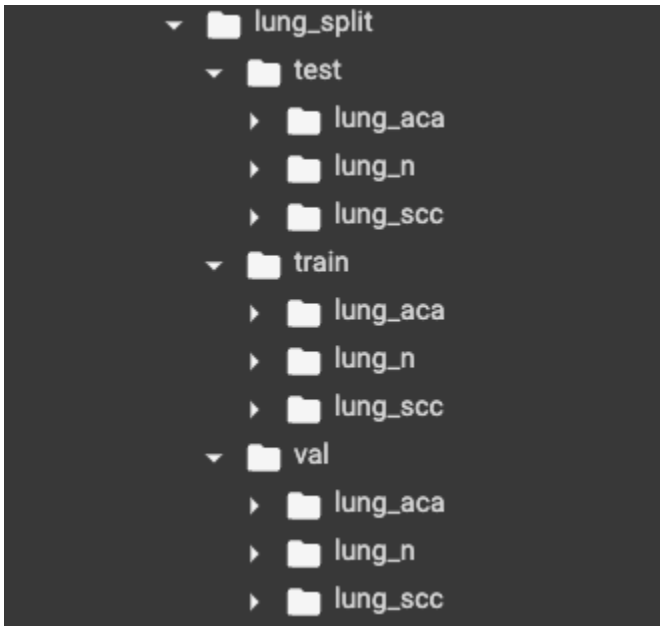
tissue. For our purposes, we have chosen to work only with the lung tissue images in this dataset.

5. **What EDA was performed?**

Prior to beginning exploratory data analysis, we set up our environment. Computer vision models often require a GPU for training, so we chose to use Colab's free GPU instance for this project. We first create a new Jupyter notebook in Colab for this project. We install and import libraries not already available in Colab and check our GPU instance is up and running. We then download the data to Google Drive, mount the drive in Colab, and remove the colon tissue images from our dataset, as we only wish to classify lung tissue. This leaves us with 15,000 images total and 5,000 of each lung benign tissue, lung adenocarcinoma, and lung squamous cell carcinoma. Because all of our classes are of the same size, we can conclude this dataset is balanced.



We use 'splitfolders' to split our image dataset into training, validation, and test sets to a named output folder. 70% of images will become training data, 20% of our images will become validation data, and 10% is set aside as test data because histopathological images of lung tissue are difficult to acquire otherwise. We view our google drive directory to find the split has taken place as desired. Because we are creating a computer vision model and the data is already relatively clean, we think it makes sense to implement a basic model to see what the model might be predicting incorrectly.



We've chosen FastAI as our deep learning library to help facilitate this. FastAI 2.0 is built on top of Pytorch and provides both quick, state of the art results in addition to being highly adjustable and customizable. We first create variables for our transformations and batch size. Our transformations are minimal, and may only flip the image or rotate it a maximum of 10 degrees. Our batch size is set to be the most the GPU can handle. In this case, 64.

We then create our DataLoader. The DataLoader class will help us explore our image data further. DataLoader are extensions of Pytorch's DataLoader class but with more functionality and flexibility. DataLoader will help at all steps of exploring, cleaning, and preparing our data before, during, and after building our computer vision model. In fact, a FastAI model cannot be built without one. It's worth noting, we will be resizing these images to 224 x 224 as the CNNs we plan to utilize for transfer learning were trained on this resolution. We pass the DataLoader the path to our split dataset, define the train and validation subsets, pass our transformations, batch size, and new image size. We choose to squish the image, rather than crop or zoom, so no image data is lost.

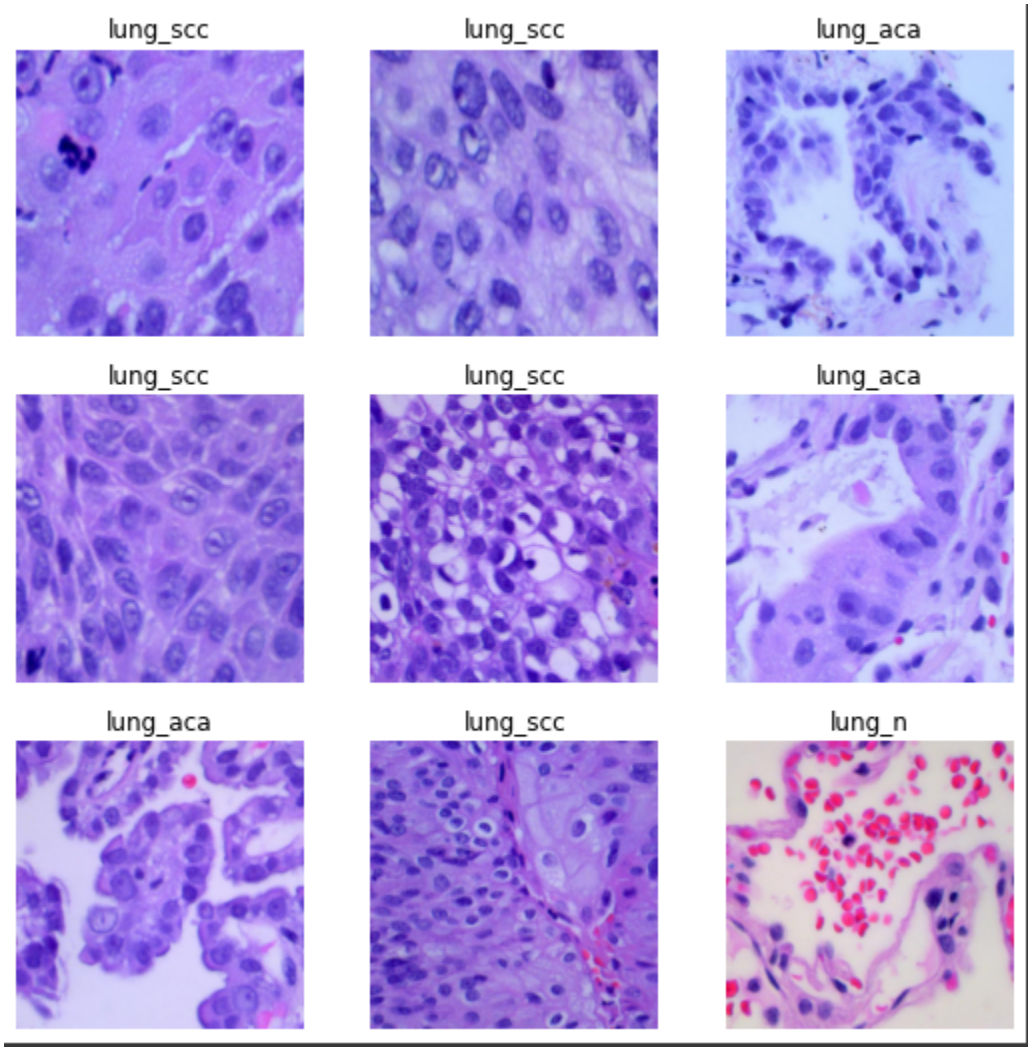
At this point, we find all three of our desired classes are present; lung adenocarcinoma, lung benign tissue, and lung squamous cell carcinoma. We check the number of images in our training and validation sets to be sure the DataLoader class was created correctly. We started with 15,000 images. 70% to train would be 10,500 images. 20% to validation would be 3,000 images. The remaining 10%, or 1,500 images, are set aside to test with our deployed model. Taking the length of the train and validation sets in our DataLoader, we find these numbers to hold true. Our data is now ready to be explored.

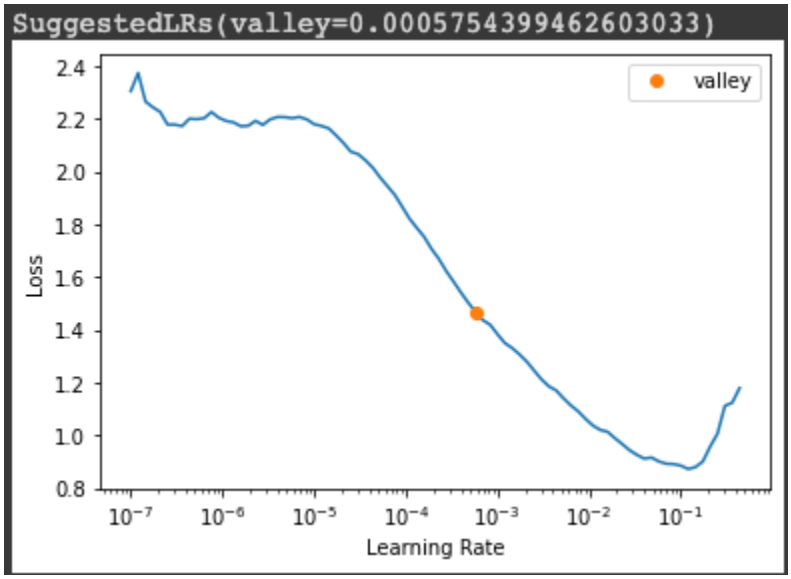
```
[ ] dls.vocab

    ['lung_aca', 'lung_n', 'lung_scc']
```

```
[ ] print(len(dls.train_ds), len(dls.valid_ds))

    10500 3000
```

We show a batch of our image dataset, specifying the batch size. This allows us to see the images and their labels. We note they appear to be resized correctly and labeled correctly. We are now ready to create our first computer vision model. We create our learner, which is a class that handles the basic training loop, by passing it our DataLoader, the pretrained model we wish to use, and a metric. In this case, we choose Restnet50 as our model and accuracy as our metric. We use FastAI's learning rate finder to visualize our learning rate plotted against loss, and pick one that looks appropriate. We then utilize transfer learning to train the last two layers

of our pretrained model, using the learning rate we picked.

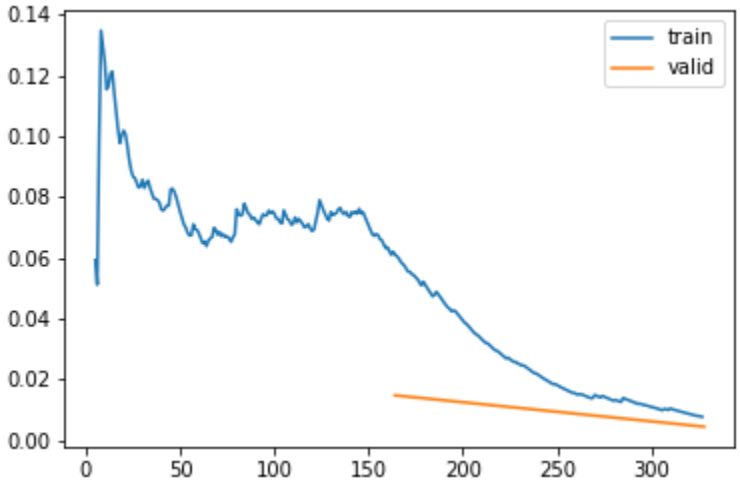




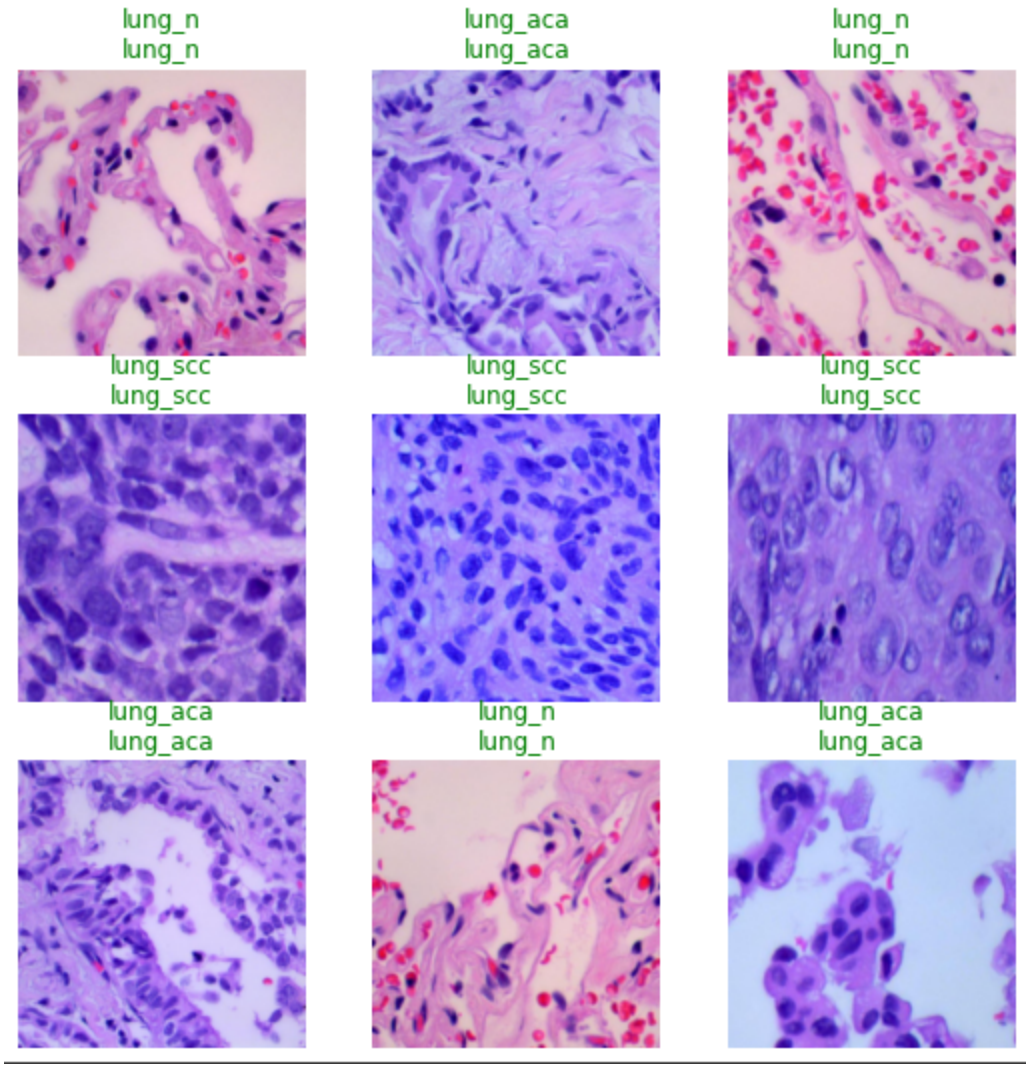
After two epochs, it seems likely our model may start to overfit. We conclude training here and visualize our loss by plotting it. After roughly 300 batches processed, our training and validation loss are at their lowest. Our model has an accuracy of 0.999; this is encouraging for a first pass. We use 'show_results' to visualize predictions and their actual labels. We can see our model is performing well. We then create a confusion matrix to visualize misclassified predictions. We can see our model perfectly predicts lung benign tissue, and only occasionally mixes up lung adenocarcinoma and lung squamous cell carcinoma.

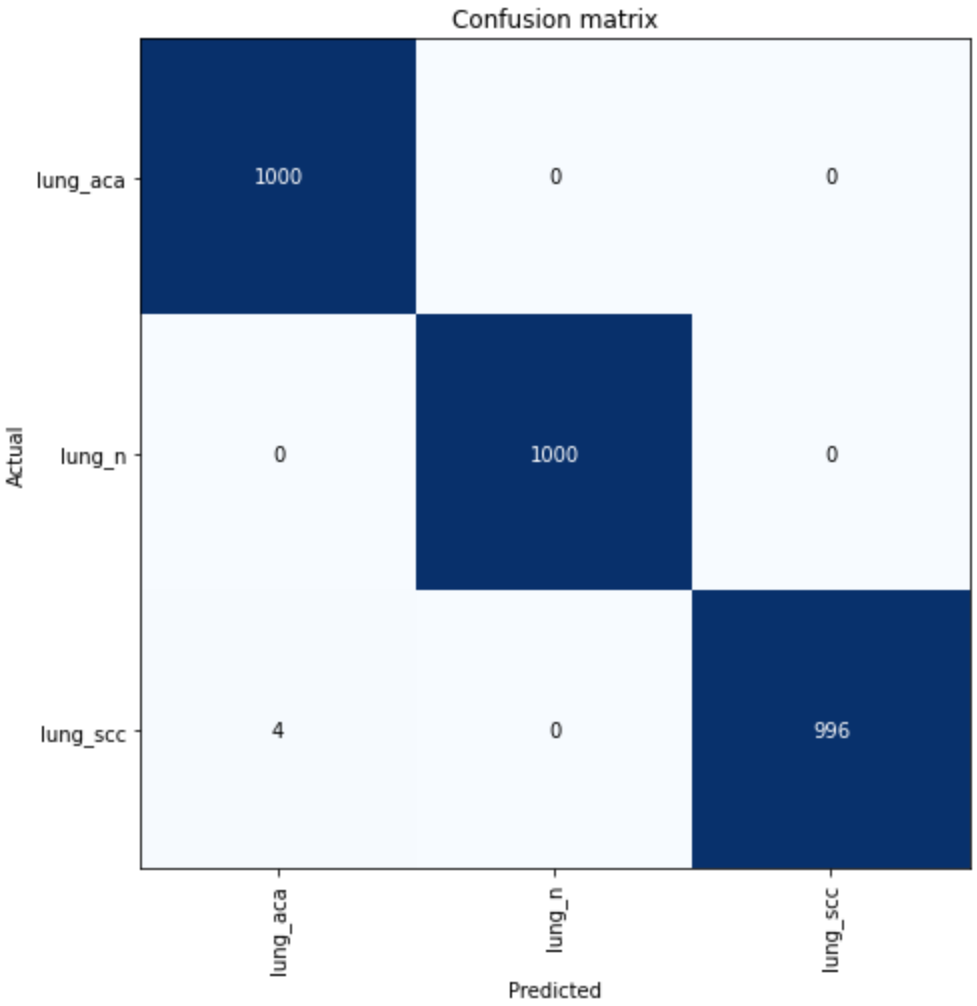
epoch	train_loss	valid_loss	accuracy	time
0	0.145546	0.065961	0.977000	15:47

epoch	train_loss	valid_loss	accuracy	time
0	0.062059	0.014836	0.993000	03:31
1	0.007783	0.004542	0.998667	03:29



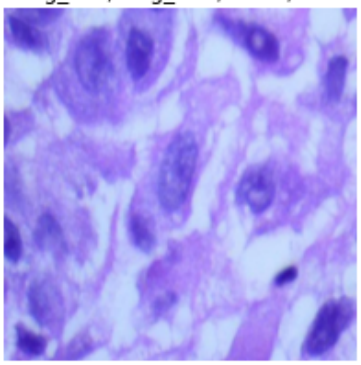
We can create and use an interpretation object to see where the model made the worst predictions and was least confident. As expected, our model experienced the greatest loss when misclassifying lung squamous cell carcinoma as lung adenocarcinoma. Alternatively, we can use 'ImageClassifierCleaner' to view the highest-loss images to allow for removal or relabeling. Upon further research, our highest-loss images do not appear to be labeled incorrectly. Therefore, we can conclude our image data has been loaded, transformed, and labeled appropriately.


lung_n
lung_n
lung_aca
lung_aca
lung_n
lung_n
lung_scc
lung_scc
lung_scc
lung_scc
lung_scc
lung_scc
lung_aca
lung_aca
lung_n
lung_n
lung_aca
lung_aca

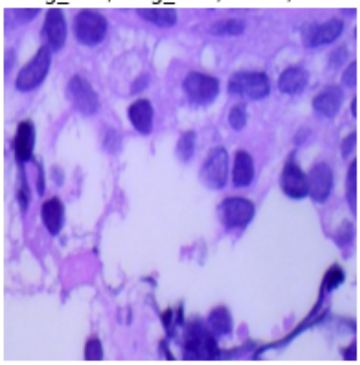

Confusion matrix
lung_aca
lung_n
lung_scc
1000
0
0
0
1000
0
4
0
996
Actual
lung_aca
lung_n
lung_scc
Predicted

Prediction/Actual/Loss/Probability

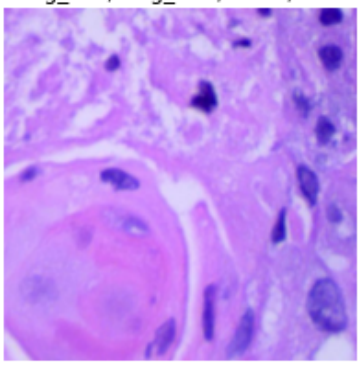
lung_aca/lung_scc / 4.09 / 0.98



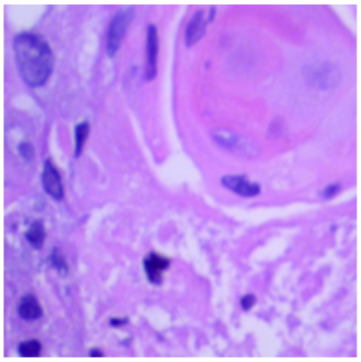
lung_aca/lung_scc / 2.60 / 0.93



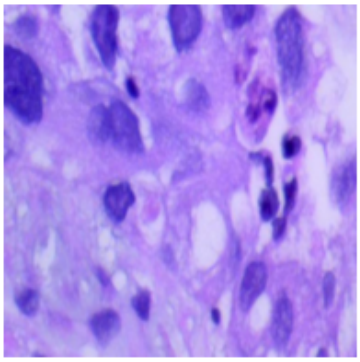
lung_aca/lung_scc / 1.95 / 0.86



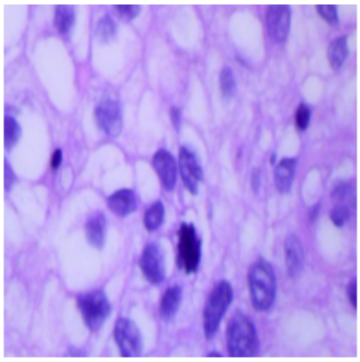
lung_aca/lung_scc / 1.36 / 0.74



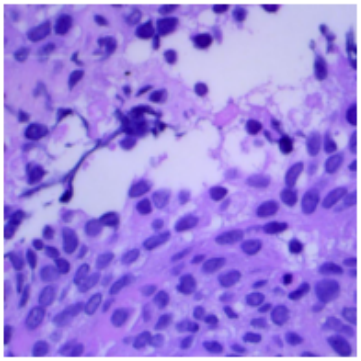
lung_aca/lung_aca / 0.59 / 0.56



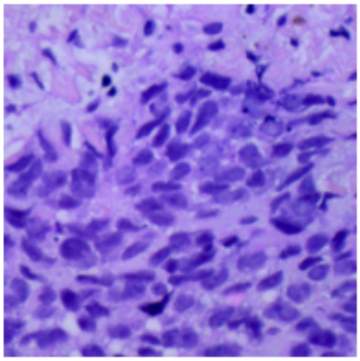
lung_scc/lung_scc / 0.32 / 0.73



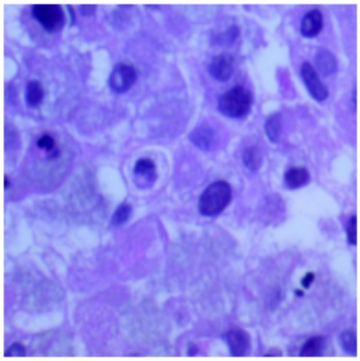
lung_scc/lung_scc / 0.18 / 0.83

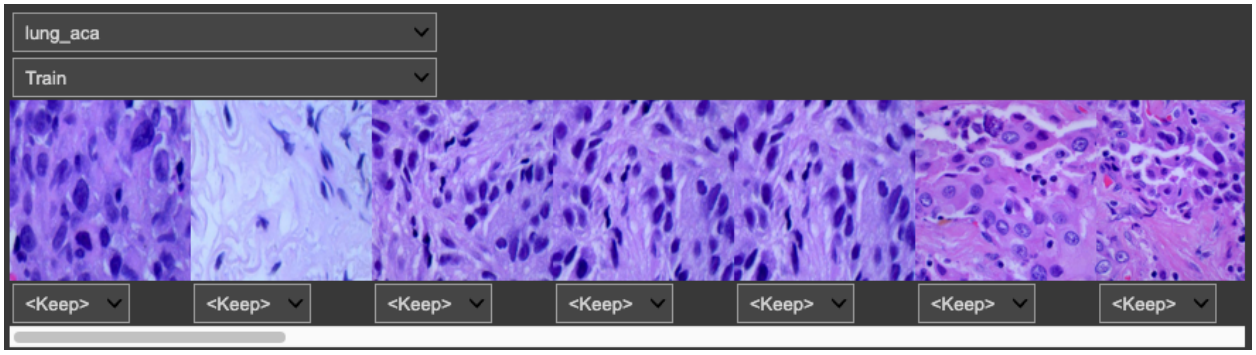


lung_scc/lung_scc / 0.15 / 0.86



lung_aca/lung_aca / 0.15 / 0.86





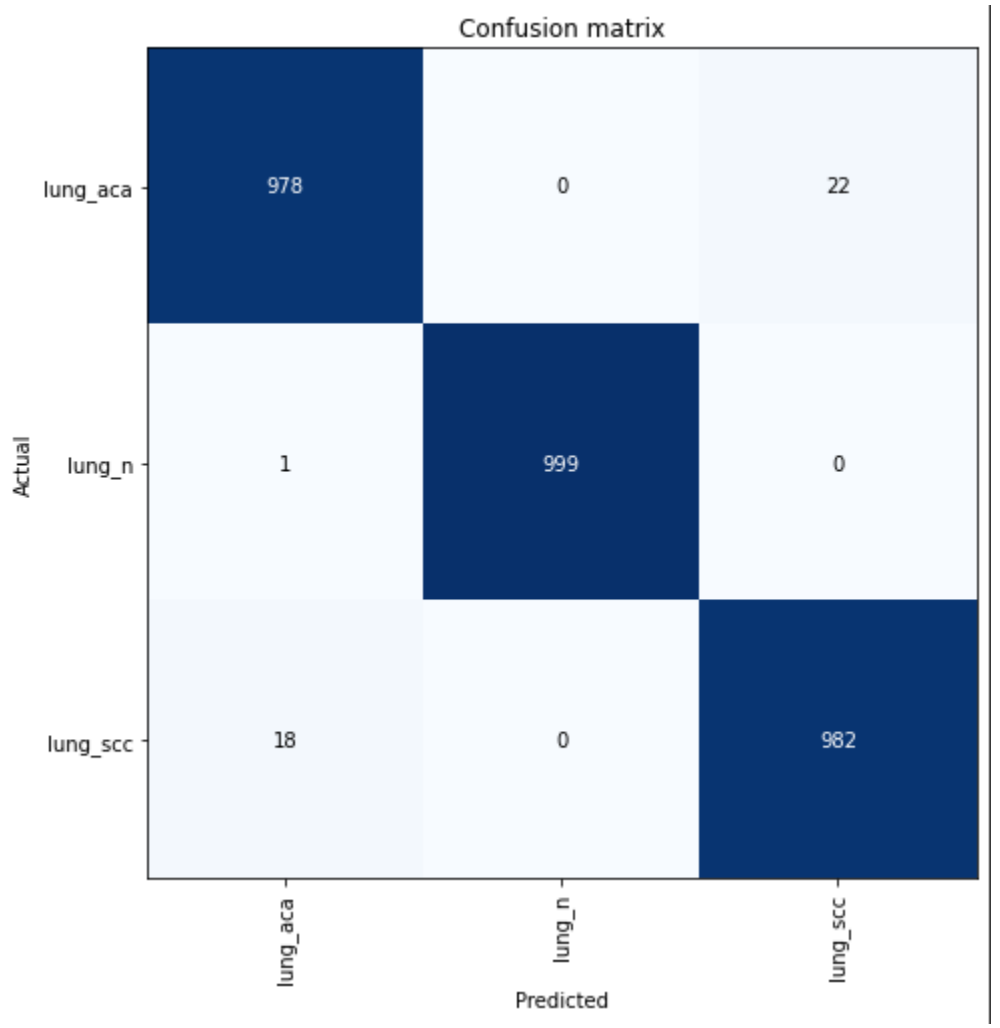
6. **Feature Exploration and Engineering**

Now that we have a good sense of our data and baseline from our Resnet50 model, we try some other models from other families that should perform similarly to our baseline. We will utilize the same transformations, batch size, and learning rate in an attempt to keep things constant. It's also worth noting, as we are utilizing deep learning we cannot and do not explore or engineer features. We instead rely on our models to determine the best features as they train.

7. **Model Building and Tuning**

We've chosen three model types to explore that tend to do well with image classification. The first model we try is Resnet50d from the Resnetd family. Resnet50d tweaks the model by modifying Resnet50's network architecture. Utilizing our DataLoader, we create a new learner for the Resnet50d model. Because the model is coming from TIMM and not the FastAI library, we pass the model in using quotation marks. We then utilize transfer learning to train the last two layers of the model. Interestingly, Resnet50d performs worse with an accuracy of 0.986. Furthermore, the model predicts benign lung tissue as cancerous. This is problematic as we don't want to provide cancer treatment to healthy lung tissue. Because this performs significantly worse than our original Resnet50 model, we choose to not proceed with Resnet50d.

epoch	train_loss	valid_loss	accuracy	time
0	0.204972	0.084226	0.968667	03:39
epoch	train_loss	valid_loss	accuracy	time
0	0.097061	0.046851	0.982000	03:48
1	0.040652	0.035451	0.986333	03:45

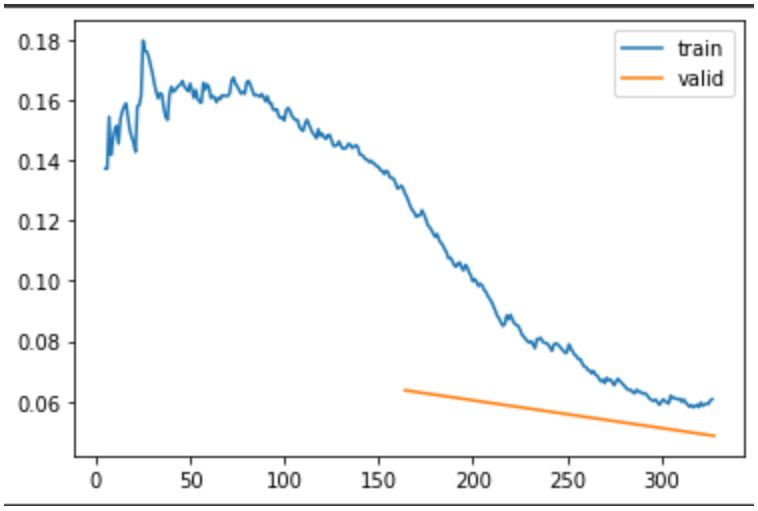


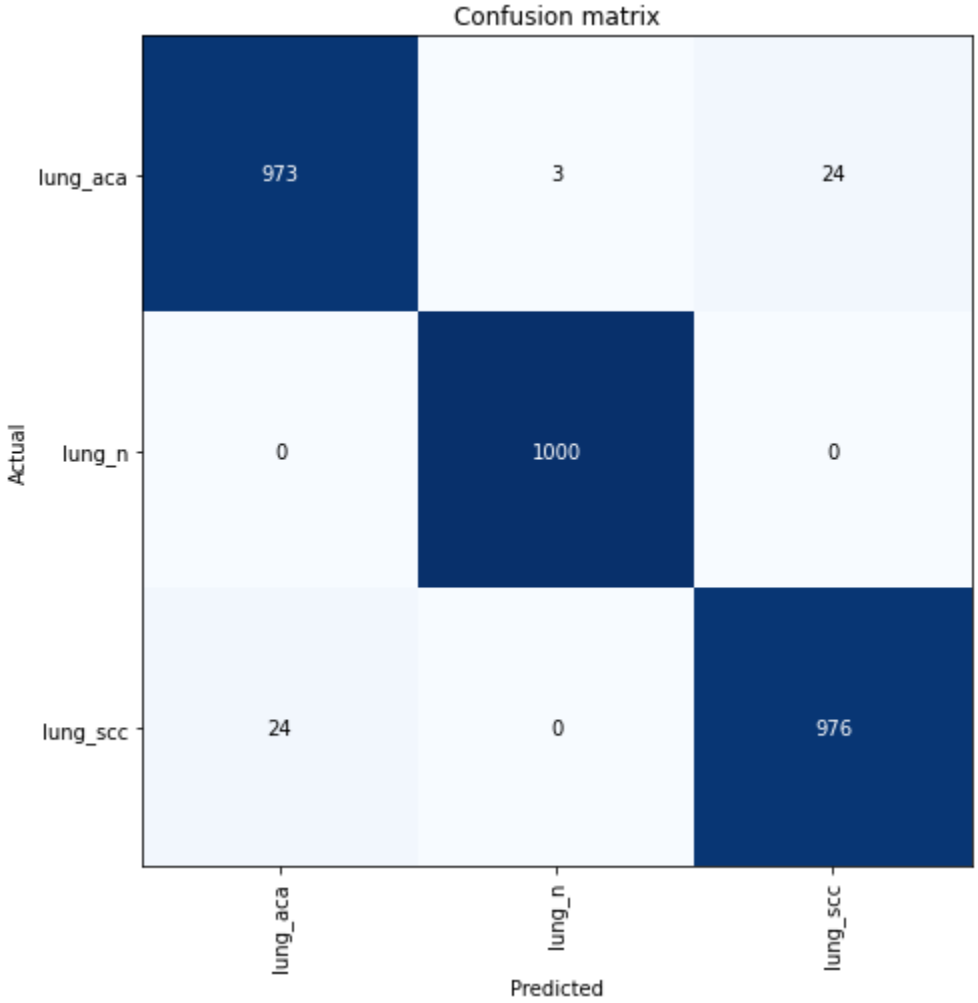
The second model we try is efficientnetv2_rw_t from the Efficientnetv2 family. Efficientnetv2 is a type of convolutional neural network with faster training speed and better parameter efficiency while being up to 6.8x smaller. It is designed to be highly efficient for visual recognition tasks. Utilizing our existing DataLoader, we create a new learner for the efficientnetv2_rw_t model. Because the model is coming from TIMM and not the FastAI library, we pass the model in using quotation marks. We then utilize transfer learning to train the last two layers of the model. Our accuracy is 0.983, and we find the model likely does not classify well enough for our purposes. This model tends to mix up lung squamous cell carcinoma and lung adenocarcinoma quite a bit.

It also predicts cancerous tissue as being benign on 3 occasions, which is problematic as we don’t want to misdiagnose a patient with lung cancer. We also find our training and validation loss bottom-out around 300 batches processed, so it is likely training this model for more epochs will not improve the model. Consequently, we disqualify efficientnetv2_rw_t.

epoch	train_loss	valid_loss	accuracy	time
0	0.297248	0.160012	0.939000	03:25

epoch	train_loss	valid_loss	accuracy	time
0	0.130778	0.063746	0.973000	03:35
1	0.060791	0.048639	0.983000	03:37



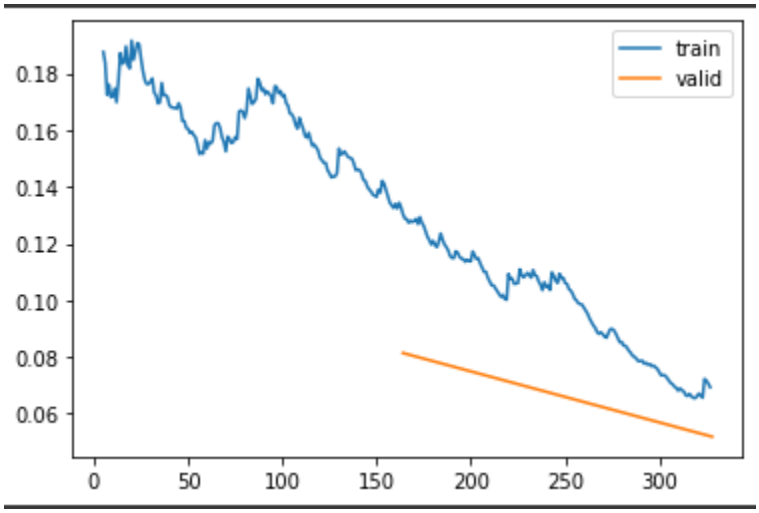


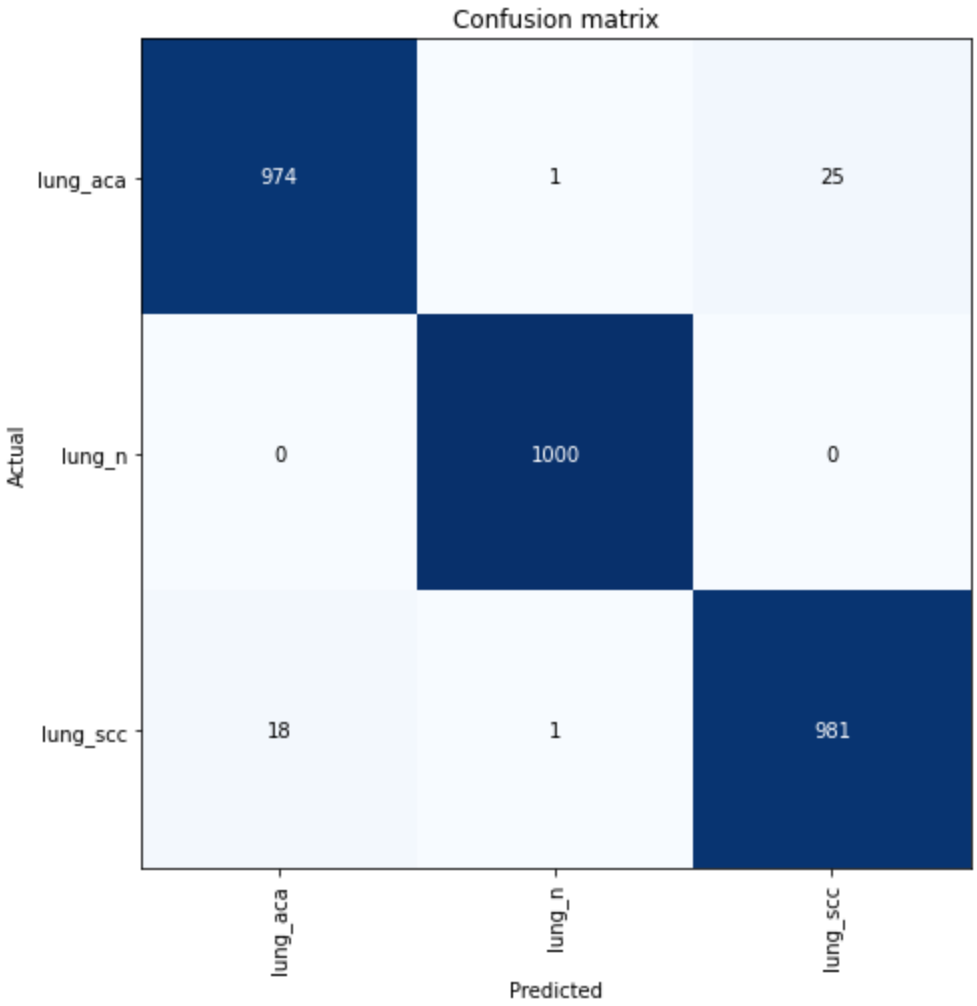
The third model we choose to try is regnetz_b16 from the family Regnetz. Regnetz is interesting because it is a network design space and not a network architecture. It is made up of different model architectures and parameters. In short, it is designed to be extremely flexible. Regnetz can be run on mobile devices or be highly accurate when tweaked for vision recognition tasks

on a GPU. Utilizing our existing DataLoader, we create a new learner for the regnetz_b16 model. Because the model is coming from TIMM and not the FastAI library, we pass the model in using quotation marks. We then utilize transfer learning to train the last two layers of the model. We find model accuracy peaks around 0.985 and training and validation loss also stall around a batch size processed of 300. This implies training our model beyond two epochs is likely not helpful. Regnetz_b16 experiences the same issues as our efficientnetv2 model in that it not only mixes up our cancerous tissue, but will occasionally predict cancerous lung tissue as being benign. This is problematic as we don't want to miss a cancer diagnosis; a patient's life could depend on identifying this deadly cancer. Consequently, we disqualify the regnetz_b16 model.

epoch	train_loss	valid_loss	accuracy	time
0	0.296972	0.159250	0.940000	03:40

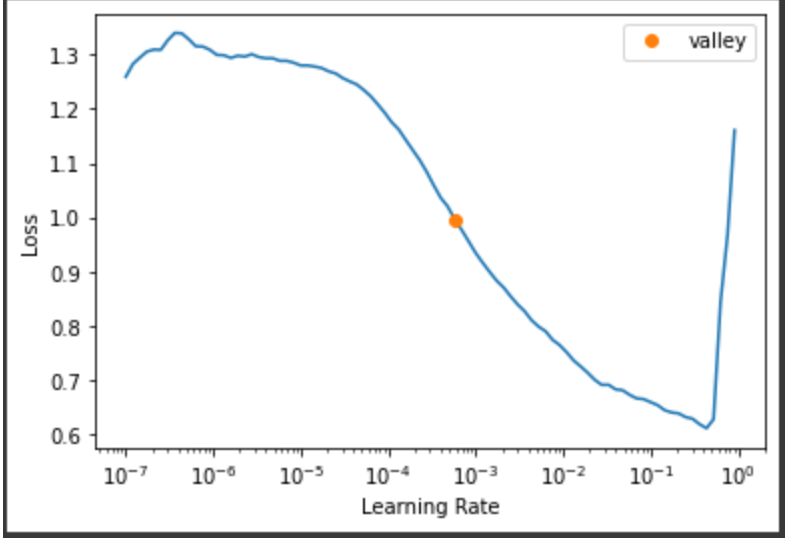
epoch	train_loss	valid_loss	accuracy	time
0	0.132705	0.081372	0.974667	03:57
1	0.069301	0.051864	0.985000	03:55



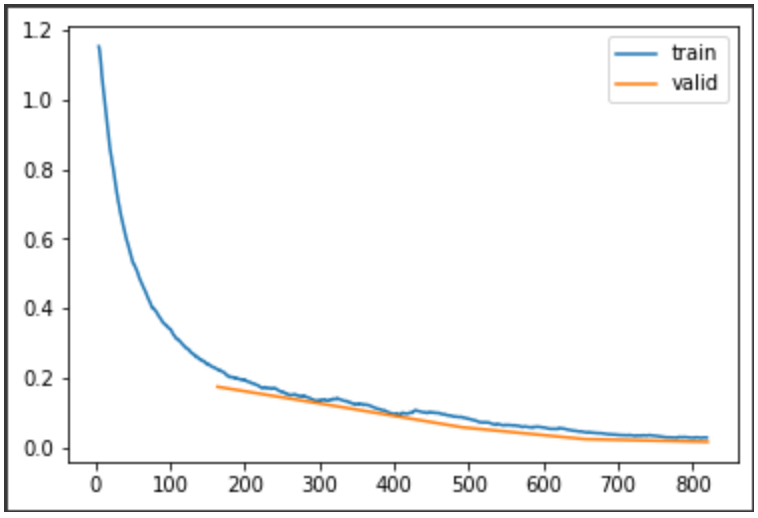


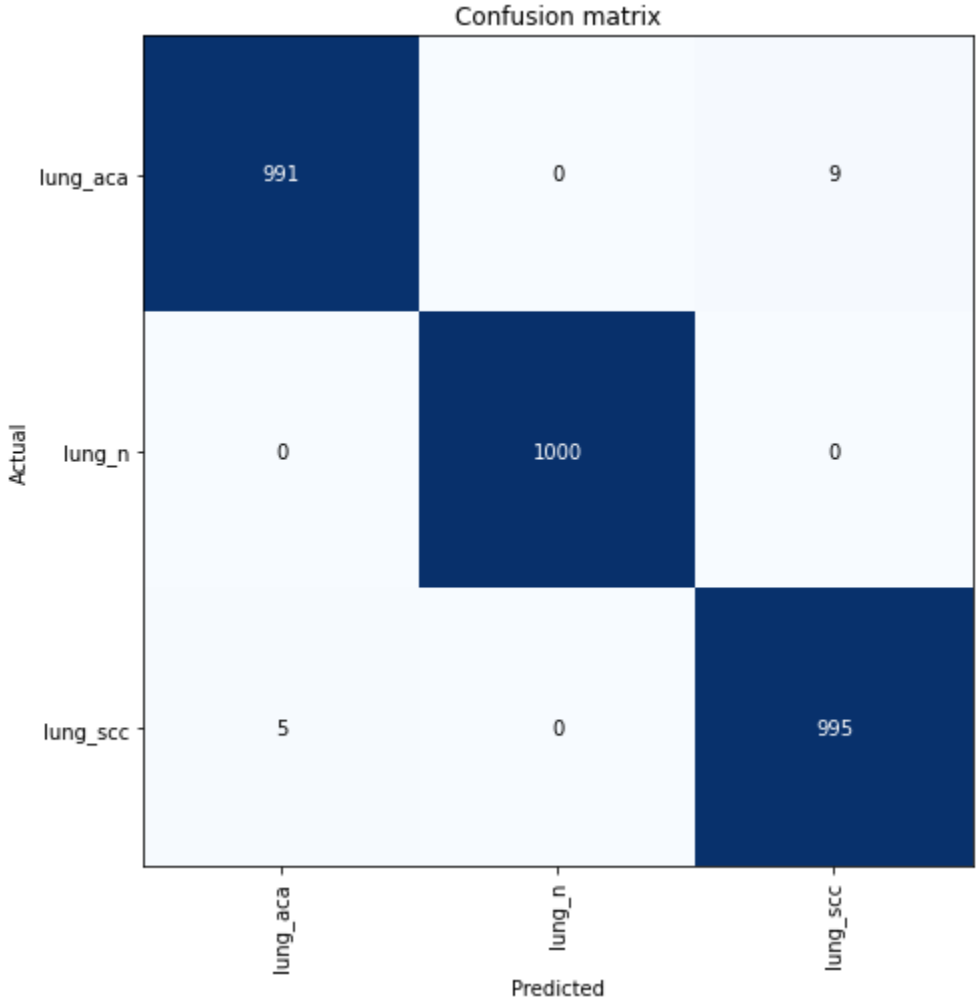
Finally, we will attempt to train a model from scratch. We've chosen to train resnet34 (denoted as xresnet34 in the FastAI library as not being pretrained) from the Resnet family as it's in the same family as our highest performing pretrained model, resnet50, but should train a little faster (which is something we want given we will be training for more epochs.) We create a new learner, using our existing DataLoader but also specifying the number of expected classes. Like before, we search for a good learning rate and fit, rather fine_tune, our model for five epochs. We find this model performs incredibly well with an accuracy of 0.995 and our training and validation loss bottom out around 800 batches processed. This model performs similarly to our resnet50 model. It classifies benign lung tissue perfectly, but seems to mix up lung adenocarcinoma and lung squamous cell carcinoma. It is also worth noting we could have changed or specified our loss function and optimizer. However, because this model performed so well we don't think it makes much sense to alter them.

SuggestedLRs(valley=0.0005754399462603033)

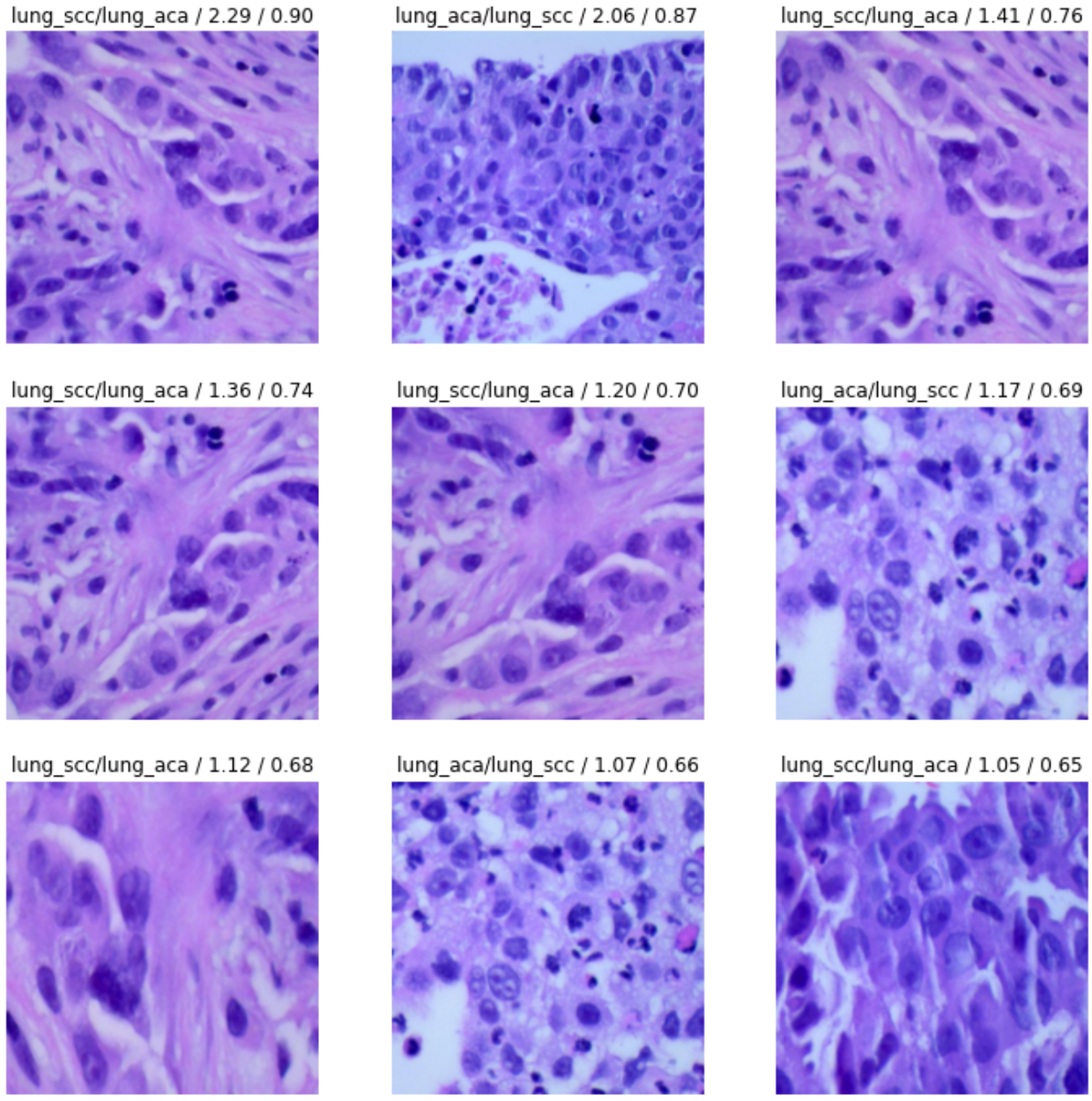


epoch	train_loss	valid_loss	accuracy	time
0	0.225664	0.174855	0.926333	20:12
1	0.139161	0.117329	0.948000	03:16
2	0.087528	0.058262	0.978333	03:26
3	0.044322	0.024319	0.991000	03:21
4	0.028557	0.016616	0.995333	03:32





Because this is so similar to the resnet50 model, we'll also plot our top losses here. Interestingly, though it misclassified more instances of lung adenocarcinoma and lung squamous cell carcinoma than the resnet50 model, our resnet34 model seemed to do so with a lower loss.



Our resnet50 model is the one that performed the best and the model we choose to export and deploy as an image classifier. However, given our resnet34 model's performance, we are led to believe the Resnet family may simply perform exceptionally well for this given problem. Future improvements may include training a resnet50 model from scratch or trying variations of a resnet101 model. In this case, 0.999 accuracy is sufficient. Furthermore, the ability to perfectly classify benign lung tissue compared to cancerous tissue is most important to us. At the end of the day, we want to correctly inform patients whether they have a small cell lung cancer or not.

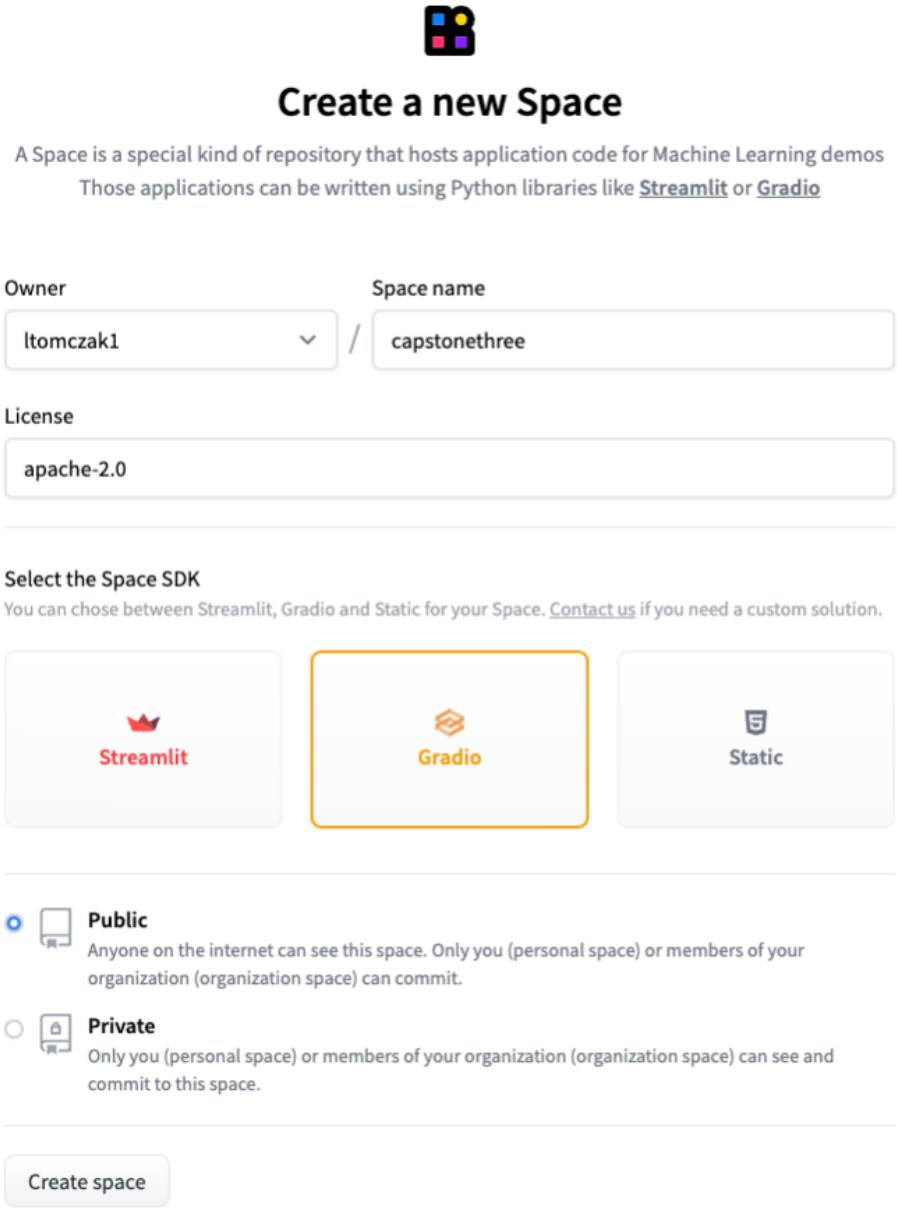
Satisfied with our model, we can export it for deployment.

```
[ ]  learn.export('lung.pkl')
```

8. **Deploying Our Model**

We will be deploying our model with Hugging Face Spaces. A Space is a special kind of repository that hosts application code for machine learning demos. These applications can be written using Python libraries (SDKs) like Stremlit or Gradio. In our case, we choose Gradio due to its simplicity and flexibility. Gradio is specifically built with machine learning models in mind. If one wants to create a UI specifically for a machine learning model, Gradio's simple setup and execution streamlines the process. Additionally, Gradio allows the user to grab an API so one can use it as a jumping-off point to built a full-fledged app.

We first install and import the necessary Gradio library. We then set up our Space on Hugging Face. We choose the Apache license and Gradio for our SDK as previously mentioned. We also set the space to 'Public' so anyone can use our app.

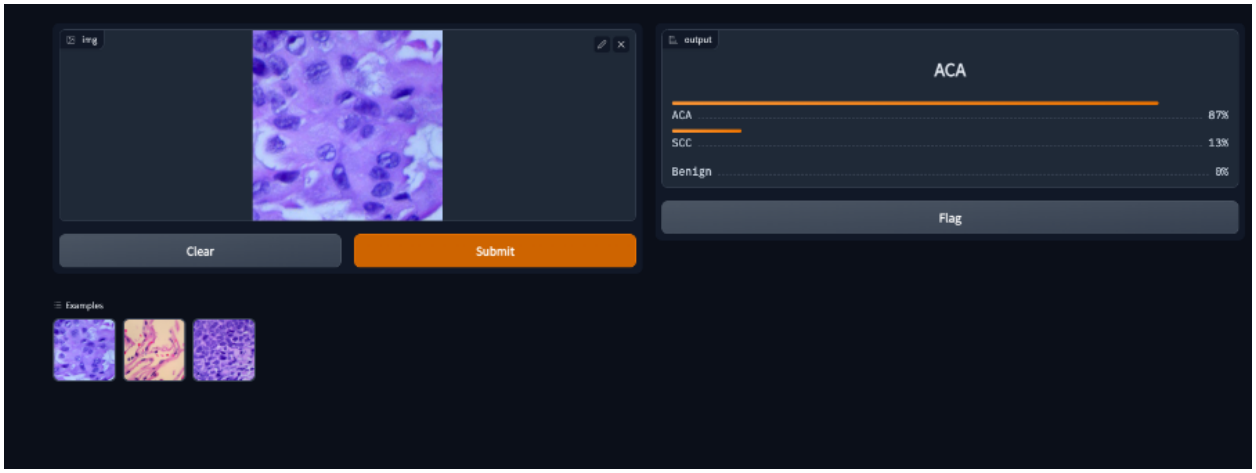


Before pushing all of our materials to Hugging Face, we first try to deploy through our notebook in Colab. We first load our saved model and create a Gradio interface by defining our categories by the order they are indexed. We must also provide a function Gradio can call. In this case, we will return the decoded prediction, the index of the predicted class, and the probabilities of all classes in order of their index labels. We return this as a dictionary of the possible categories and the probability of each one.

Now we create our actual interface. We pass in the function we created, our input (an image,) labels as output, and examples (images we will push to Hugging Face.) We launch our interface and Gradio provides a temporary link to demo our lung cancer image subclassifier. Our app is

functional! Gradio requires a Python script, so we alter the code slightly and push the script, model,m examples, and requirements file to Gradio.

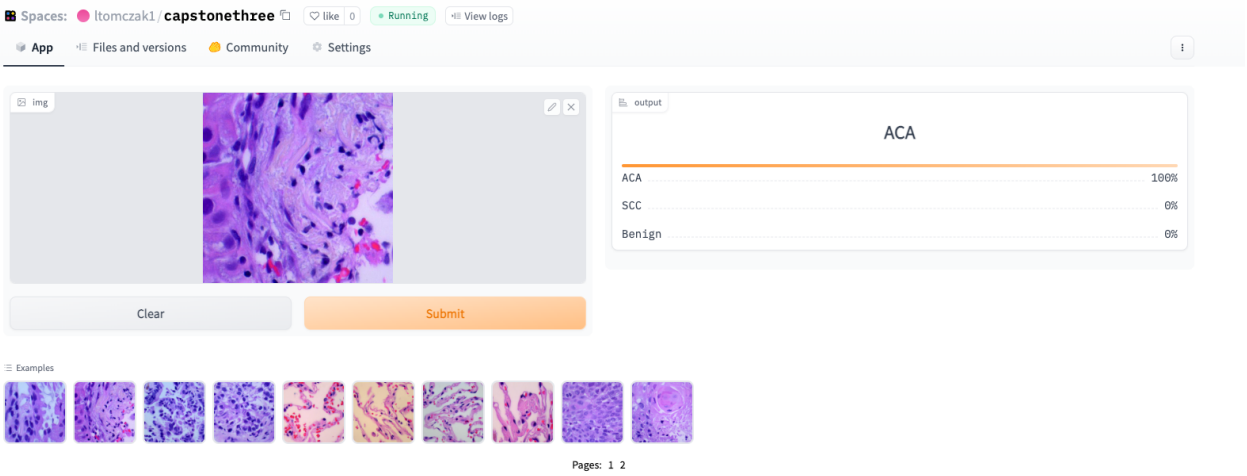
```
Colab notebook detected. To show errors in colab notebook, set `debug=True` in `launch()`
Running on public URL: https://522cc059cc272549.gradio.app

This share link expires in 72 hours. For free permanent hosting and GPU upgrades (NEW!), check out Space
(<gradio.routes.App at 0x7f815ef5dcd0>,
 'http://127.0.0.1:7861/',
 'https://522cc059cc272549.gradio.app')
```



Our model is now deployed! Images from the publicly available test set we created can be run through the image classifier. This is important, as our model has never seen these images before. Additionally, we've provided some example photos with the app for those that do not wish to drag and drop images from the test set.

main capstonethree 1 contributor History: 29 commits + Add file

ltomczak1 Update requirements.txt 01e0c69			5 minutes ago
.gitattributes	1.34 kB	initial commit	14 days ago
README.md	251 Bytes	initial commit	14 days ago
app.py	721 Bytes	Update app.py	25 minutes ago
lung.pkl pickle	103 MB LFS	Upload lung.pkl	6 days ago
lungaca.jpeg	55.1 kB	Upload 3 files	14 days ago
lungaca142.jpeg	72.7 kB	Upload 12 files	6 days ago
lungaca199.jpeg	76.4 kB	Upload 12 files	6 days ago
lungaca861.jpeg	72.4 kB	Upload 12 files	6 days ago
lungn.jpeg	57.9 kB	Upload 3 files	14 days ago
lungn159.jpeg	58.9 kB	Upload 12 files	6 days ago
lungn380.jpeg	65.3 kB	Upload 12 files	6 days ago
lungn633.jpeg	54.6 kB	Upload 12 files	6 days ago
lungscc.jpeg	70.4 kB	Upload 3 files	14 days ago
lungscc344.jpeg	80.1 kB	Upload 12 files	6 days ago
lungscc626.jpeg	61.4 kB	Upload 12 files	6 days ago
lungscc958.jpeg	65.4 kB	Upload 12 files	6 days ago
requirements.txt	36 Bytes	Update requirements.txt	5 minutes ago



[The working app, with examples, can be found here.]
[The test data set can be found here.]

9. Recommendations

Our image classifier can now classify lung tissue more accurately than a domain expert. Based on the strong ability of our image classifier to correctly classify lung benign tissue, lung adenocarcinoma, and lung squamous cell carcinoma, we can provide some recommendations for its use.

First and foremost, we recommend oncologists take biopsies of lung tissue for those suspected of having cancer or those that are precancerous. We would also suggest the images be standardized, aka magnified, scaled, and de-identified to the same specifications as this dataset. Next, the image can be passed to our classifier, which may be hosted on a secure server to protect private information. Our image classifier should correctly classify the type of lung tissue, and an oncologist can then develop a treatment plan for the patient based on the cancer identified.

Beyond correctly identifying the cancer and creating a targeted treatment plan, there are some additional benefits. The first benefit here is a trained pathologist may not be required to examine the biopsy and make a determination. This can lead to a faster turnaround time and a more accurate diagnosis; something critical with faster moving lung cancers. This could also lead to reduced patient costs, as a specialist may not be required if using this image classifier. Finally, this could lead to more data on lung cancer. With more unique image data, our model may be further improved. Given what we've learned about the types of non-small cell lung cancer, where they grow, and how quickly they grow, this lung cancer subclassification model could potentially save lives.

10. Citations

1. Borkowski AA, Bui MM, Thomas LB, Wilson CP, DeLand LA, Mastorides SM. Lung and Colon Cancer Histopathological Image Dataset (LC25000). arXiv:1912.12142v1 [eess.IV], 2019
2. https://www.cdc.gov/cancer/lung/basic_info/
3. https://www.cancer.org/cancer/lung-cancer/if-you-have-small-cell-lung-cancer-sclc.html
4. https://go2foundation.org/