Breast cancer IDC image prediction

By Breanna Parker

Link for github code: https://github.com/breannaparker1991/breast_cancer

Business problem:

For hospitals, detecting cancer can be time consuming and can cost a lot of money if it's all done manually. This cost will get pushed to the patients which can make the test almost inaccessible to many people who can't afford it, causing them to not recognize cancer in the early stages. By automating the process so that code can look through images to id cancer cells, hospitals will be able to run more tests, and they will be cheaper, meaning doctors are more likely to order them. With an increased ability to run more cancer testing, more people can also catch their cancer in the early stages, which will allow for earlier treatments.

Background/History:

The images consist of over 5000, 50 x 50 pixel RGB images of stained breast histopathology samples. Being able to detect cancerous cells can be difficult due to mishandling of specimens or improper staining and can be laborious work. Right now, after a sample is taken from the patient, it is slided up very thinly, placed on a slide, and stained. This will created multiple slides and these are all going to be viewed by a pathologist (or similar role) at a hospital to ID if the patient has cancer or not. This is a very laborious process, and if we can run the slide images through a computer system, this will save our pathologist a lot of time.

Data Explanation (Data prep/Data dictionary/etc):

There are two folders in this project. One has over 5000 images and the other has the corresponding labels. 1 = positive for IDC and 0 = negative for IDC. The data was then split into a train/test split and reshaped. There wasn't any missing data, names that needed to be changed, or anything that needed to be adjusted for this data preparation.

Methods:

I ran six different models to see which one is the most accurate. Those include keras classifier, decision tree classifier, random forest classifier, svc, logistic regression, and kneighbors classifier. After running each method, I tested how good that they would be for the project by running the accuracy score, precision score, the f1 score, the recall score, and the confusion matrix. I wanted to make sure that we looked at all aspects of the model to make sure it would be the best fit for this project.

```
#total number of images
print('Total number of images: {}'.format(len(X)))
```

```
Total number of images: 5547
```

```
#total number of negative IDC images
print('Number of negative IDC Images: {}'.format(np.sum(Y==0)))
```

```
Number of negative IDC Images: 2759
```

```
#total number of positive IDC images
print('Number of positive IDC Images: {}'.format(np.sum(Y==1)))
```

```
Number of positive IDC Images: 2788
```

```
#shape of the images
print('Image shape (Width, Height, Channels): {}'.format(X[0].shape))
```

```
Image shape (Width, Height, Channels): (50, 50, 3)
```

```
#train/test split
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
```

:
```
#logisitic regression accuracy
lscore = lr.score(x_test_r,y_test)
print("logistic Regression accuracy", lscore)
```

```
logistic Regression accuracy 0.6711711711711712
```

:
```
#logistic regression classifying and fittnig
knn = KNeighborsClassifier()
knn.fit(x_train_r,y_train)
```

:
```
▾ KNeighborsClassifier
KNeighborsClassifier()
```

:
```
#logisitic regression accuracy
kscore = knn.score(x_test_r,y_test)
print("KNeighbors accuracy", kscore)
```

```
KNeighbors accuracy 0.727027027027027
```

In [147]:
```
#running the accuracy of the decision tree
dscore = dtc.score(x_test_r,y_test)
print("Decision Tree Score: ", dscore)
```

```
Decision Tree Score:  0.681981981981982
```

In [149]:
```
#classifying random forest tree and fitting it
rfc= RandomForestClassifier(n_estimators = 100, random_state=42) #n_estimator = DT
rfc.fit(x_train_r,y_train)
```

Out[149]:
```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

In [150]:
```
#accuracy of random forest
rscore=rfc.score(x_test_r,y_test)
print("Random Forest Score: ", rscore)
```

```
Random Forest Score:  0.7855855855855856
```

In [151]:
```
#SVC classiying and fitting
svc = SVC(random_state=42)
svc.fit(x_train_r,y_train)
```

Out[151]:
```
SVC
SVC(random_state=42)
```

In [152]:
```
#accuracy of SVC
sscore = svc.score(x_test_r,y_test)
print ("SVM Accuracy:", sscore)
```

```
SVM Accuracy: 0.7882882882882883
```

Analysis:
The svc or random forest classifier are the two most accurate classifiers at around 78%. I attempted to run a gridsearch to see if that would help, but the code would never run. 78% is very inaccurate, especially for a hospital setting. It might be necessary to get a larger sample size for more accurate data, closer to 100k. The precision score for svc was 80%, the recall score was only 77%, and the f1 score was 79%. These were the best scores out of all of the methods ran but it still isn't good enough to be used for this project. I even ran the confusion matrix to see if there were a lot of false negatives and positives, and there was a significant number in both categories, for all models used.

Conclusion:
This sample size and models used as is, are not good enough for a hospital setting. Either a bigger sample size or different model is needed to make this more accurate. Breast cancer identification is a very complicated procedure, so it would make sense that 5000 images

wouldn't be enough. This would be a very useful computer program that should continued to be worked on until it's done correctly. Many different businesses will find this very useful.

Assumptions:
The biggest assumption that we're making with this data set is that the pathologists are correct in which images are IDC positive and which ones are negative. If this is at all inaccurate, then it will give us issues when trying to find an accurate method.

Limitations:
Only having a specific number of images and having to try each method to try and find out which one is the most accurate are all limiting to getting the best results. Also, since this is a very important project, if people have cancer, there has to be a high level of accuracy for it to be useful. The hospital doesn't want to risk false negatives or false positives. Even though all positives could be verified by a pathologist before resulting them out, having a large number of false negatives means that many people that have cancer will be told that they do not, which will open up a lot of liability to the hospital and doctors. And having a pathologist review every single result might defeat the purpose of this project. It might save time to tell the pathologist what it thinks the images are but, if the pathologist still has to review every slide, it won't save much time overall.

Challenges:
It was definitely difficult trying to figure out how to reshape the images so that we can use the train/test method. On top of that, finding a good method to use has been difficult since all methods currently used are all below 80% accurate. Also, not only do you need to train the computer to be able to identify cancerous versus non-cancerous, but if there are staining issues as well. The computer needs to be able to identify what "bad" parts of a slide might look like due to mishandling or incorrect staining.

Future Uses/Additional applications:
Being able to identify cancer cells via images is very useful. However, there are many different ways in which we can use image classification for people's faces, for forest fire detection from photos taken from a drone, or even for grocery shopping. Image classification is very important in our day to day lives and it would be very useful to find a more accurate method for detection.

Implementation Plan:
If this project was accurate, we could run a validation method at a hospital by having a pathologist to look at every sample, along with the machine to make sure that it's accurate. Then, after one hospital uses it successfully for some time, it could get rolled out to multiple hospitals. There should also be a daily quality control to make sure that the method never strays or needs adjustment.

Ethical Assessment:
All of these images had to come from patients. I don't think that these patients all gave permission for their images to be used in a medical setting. I'm sure that agreeing to tests

means that they agree for their information to be used for training purposes. However, is it really okay to be using people's test results so freely without their consent? Also, what about the women who have been wrongly told that they don't have cancer? The next time they go to a doctor with a problem, the cancer might be too far progressed for there to be any treatment. This is a very serious consequence, which is why there needs to be multiple methods in place to make sure that this almost never happens.

Questions:

1. What to do about false negatives?
    a. We need to find a method that has little or no false negatives. However, we can also run all patients in duplicate to confirm the test results. If the results don't agree, a pathologist can look at the results to confirm.
2. How is it proven that the images used are real positives and negatives?
    a. A pathologist would have had to ID these images. There should be multiple identification methods to make sure that these are all accurate.
3. How to make sure that the methodology is accurate?
    a. We run the accuracy, precision, recall score, f1 score, and a confusion matrix to check for accuracy. If any of these models were good, we would be able to see through these methods.
4. If the patient has a different cancer, will the system catch it?
    a. This will most likely be no, since the images are only looking at specific irregularities that it has been trained on. But, it might be possible to train it to look for multiple types of cancers.
5. Can there be a secondary check to catch any false results?
    a. Having a pathologist double check every result would not be a good use of time. The slides could be run through the system twice and any irregularities be checked by a pathologist. However, at that point, we need to see what the likelihood is that there are two false negatives.
6. Can this be used for other histology stained samples (i.e. cancer of the lung, etc.)?
    a. This is a capability for the machine, but there would need to be another set of training slides for this particular cancer.
7. Can a daily report be pulled for the number of positives and negatives called?
    a. This is something that can be easily be printed and looked at.
8. Can more samples be continually added to the training set to make it more accurate?
    a. Yes, and the system can be regularly validated. We can run a group of samples through it regularly to retrain it. Also, before testing begins, we can run a set of validated slides that are known positives and negatives, to make sure that the system is working accurately before each day.
9. How quickly does this run?
    a. This should run very quickly and each patient should be run within minutes.
10. What are things that can cause false results?
    a. Smudged slides, incorrectly stained slides, not properly trained model can all cause false results.

Appendix:

A;, J. A. (n.d.). *Deep Learning for Digital Pathology Image Analysis: A comprehensive tutorial with selected use cases*. Journal of pathology informatics. https://pubmed.ncbi.nlm.nih.gov/27563488/

Joseph, R. (2022, October 18). *Grid search for model tuning*. Medium. https://towardsdatascience.com/grid-search-for-model-tuning-3319b259367e

Sharma, P. (2024, March 15). *Build your first image classification model in just 10 minutes!*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2019/01/build-image-classification-model-10-minutes

Image datasets: https://www.kaggle.com/datasets/simjeg/lymphoma-subtype-classification-fl-vs-cll?resource=download , https://www.kaggle.com/code/allunia/breast-cancer