Noah Mezher

Wenjin Zhou

Final Project – Defined

Computing for Health and Medicine

Project Description

For this final project, I opted to follow the tutorial on how to use the MNIST dataset on a Convolutional Neural Network to train it. The MNIST dataset is not a clinical dataset, however it still offers valuable insight as to how to train a neural network. The MNIST Database (abbreviated from the Modified National Institute of Standards and Technology) is a repository containing handwritten digits, most used for training image processing systems. The Convolutional Neural Network (CNN) that I trained using the tutorial from k-d-w.org[1] uses this dataset to train survival analysis. Survival analysis is used in clinical cases to predict the time to death based on some events. The goal of this network is to increase the survival rate of patients with the goal of patient longevity.

In order to replicate what would happen in a real clinical study, some masking is used on the data in question. Patients may drop out of studies due to unforeseen events without any resolution as to why that case failed. This network employs "censoring" on MNIST data to account for this, with some records being censored and other records being uncensored. If a patient experiences an event while being recorded under the study, that data is "uncensored". On the other hand, if a patient drops out of the study or loses contact with the conductors of the study, that data is "censored".

Since MNIST is a dataset of digit images, the risk data is generated by randomly assigning a risk score and a risk group to each digit. Some digits may perform better in each group than others based on their risk score. There are four risk groups, and each risk score is independent of each other. The higher the risk score, the more likely that digit will encounter an event. The full database of handwritten digits can be found and downloaded from Yann LeCun's website[2].

Related Work

In a study first published March 12, 2018, Researchers Mobadersany et. al.[3] developed a convolutional neural network to predict cancer patient survival rates. Their method uses microscopic images of brain tissue as input to the neural network, which simultaneously diagnoses the patient and trains itself based on its diagnosis success. Their research found that the neural network surpasses the accuracy and objectivity of a human doctor's diagnosis and presents promising insight as to how cancer is diagnosed in the future.

While the above study innovative in its approach with neural networks and cancer diagnoses, there is a substantial amount of research done to back up their method. In a 2008 study[4], researchers found that computer processing had advanced to the point where they were able to generate quantifiable parameters to help in neuroblastic prognoses. Their network was trained in a similar fashion, where images of cancer cells were used as input with manual changes made to make the

[1] https://k-d-w.org/blog/2019/07/survival-analysis-for-deep-learning/

[2] http://yann.lecun.com/exdb/mnist/

[3] https://www.pnas.org/content/115/13/E2970

[4] https://www.archivesofpathology.org/doi/full/10.1043/1543-2165(2008)132%5B903%3ACGOND%5D2.0.CO%3B2

network more accurate. The researchers in that study concluded that their network produced a slide classification accuracy of 87.9% with a sample size of 36 neuroblastoma cases. Their results provided a promising addition to the use of neural networks to diagnose and treat cancer patients.

The use of convolutional neural networks and handwritten image recognition is not a concept that is foreign to most researchers in the field. Numerous studies have been conducted regarding training and accuracy while using handwritten images as input, including a study on gradient based learning dating as far back as 1998.[5] This study found that convolutional neural networks outperformed all other techniques they tried. This is because of their ability to deal with variances in 2D images, making them the optimal choice in a project such as this.

The credibility of a convolutional neural network is not unknown to the academic world, and it has been utilized for over 20 years in the field. By extending this data science approach within a powerful language such as Python, we can extract incredible processing power to train a neural network for analysis. By using a simple set of text input data such as MNIST and randomly assigning survival rates and risk groups, we can generate meaningful data that mirrors what occurs in the healthcare industry every year. These methods give researchers useful data that will help shape how we diagnose cancer patients now and in the future.

Methods

By following the tutorial in the jupyter notebook found on k-w-d.org, I was able to generate neural network data based on the MNIST dataset. As described previously, each digit ranging from 0-9 was assigned one of four risk groups (ranging from 0 to 3) and a risk score. The higher the risk score of a particular digit, the more at risk that digit is within its group to experience an "event" indicating the end of that piece of data's life. If an event was witnessed, that piece of data is uncensored, and if data is lost, that data is flagged as censored. In all, this neural network performs three important steps: randomly generating synthetic data to assign to ten digits, training the neural network to recognize events, and predicting the survival of our input data.

Results and Discussion

class_label	risk_score	risk_group
0	3.071	3
1	2.555	2
2	0.058	0
3	1.790	1
4	2.515	2
5	3.031	3
6	1.750	1
7	2.475	2
8	0.018	0
9	2.435	2

class_label	risk_score	risk_group
0	3.464	3
1	2.848	2
2	0.020	0
3	2.000	1
4	2.828	2
5	3.444	3
6	1.980	1
7	2.808	2
8	0.000	0
9	2.788	2

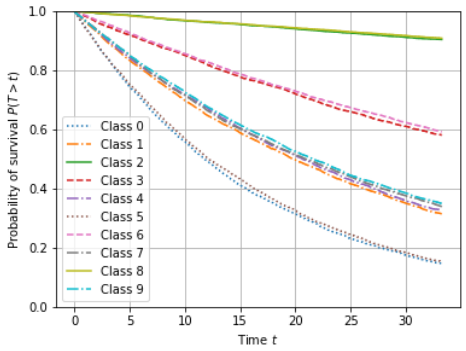
[5] https://ieeexplore.ieee.org/abstract/document/726791

After some trial and error, I was successfully able to get the jupyter notebook to train a neural network and output the probability of survival based on randomly seeded input (as shown above). I also modified the random generation of the data to see what different numbers would look like on the output graphs once the neural network is sufficiently trained. For reference, the original results produced in the article will be on the left, and my modified results will be on the right for each of the data steps.

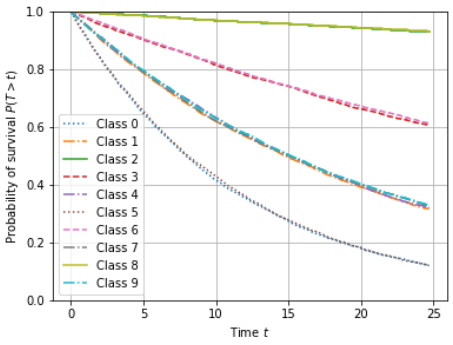
We can use the generated censored data and estimate the survival function $S(t)$ to see what the risk scores actually mean in terms of survival. We stratify the training data by class label, and estimate the corresponding survival function using the non-parametric Kaplan-Meier estimator.

```
[ ]  styles = ('-', '--', '-.', ':')

     plt.figure(figsize=(6, 4.5))
     for row in risk_score_assignment.itertuples():
         mask = y_train == row.Index
         coord_x, coord_y = kaplan_meier_estimator(event_train[mask], time_train[mask])
         ls = styles[row.risk_group]
         plt.step(coord_x, coord_y, where="post", label=f"Class {row.Index}", linestyle=ls)
     plt.ylim(0, 1)
     plt.ylabel("Probability of survival $P(T > t)$")
     plt.xlabel("Time $t$")
     plt.grid()
     plt.legend()

     #the graph clearly shows that the higher the risk of a patient, the less
     #likely that patient is to survive over time.
```

```
<matplotlib.legend.Legend at 0x7fe2971a1828>
```

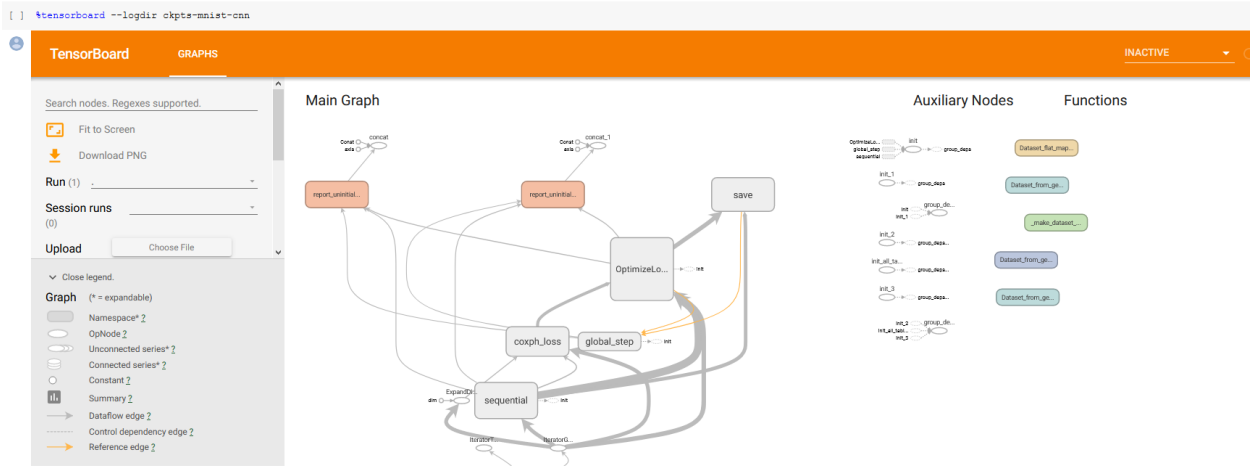



```
<matplotlib.legend.Legend at 0x7fe284523c18>
```



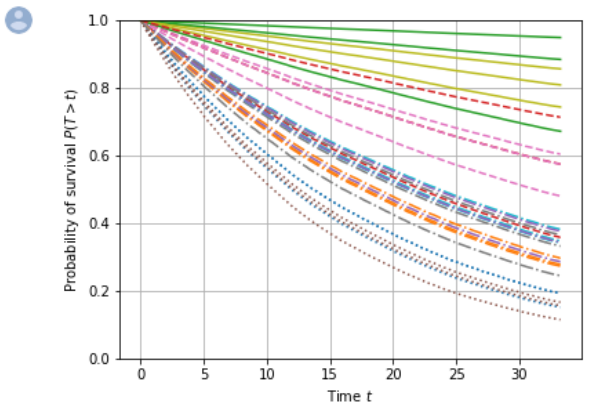
As you can see, with my adjustments the classes in each risk group follow a tighter trend outside class 2 and 8, which seem to follow more loosely after my adjustments. After my adjustments, the concordance index on test data with actual risk scores also increased from .705 out of 1 to .719 out of 1. In both cases, each data point in each risk group follows a similar path to those within its risk group, which makes sense. A cancer patient that has a similar cell structure to another cancer patient can expect to experience similar symptoms and events compared to those in a different risk group with different results. Even though we are using handwritten numerals as input, the parallels to real-world diagnoses are invaluable, and will save lives when put in the right hands.

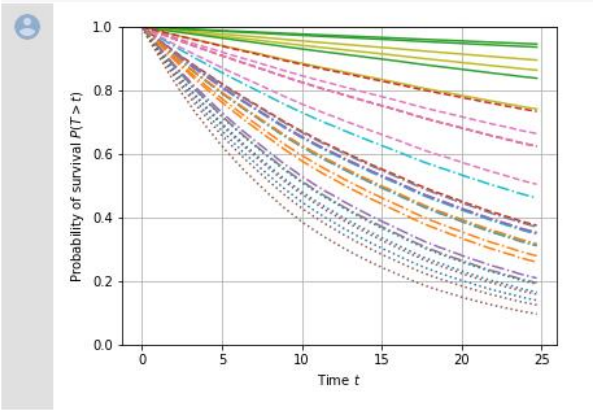
However, all of this prediction data is useless without actually comparing the prediction values to actual results. After the neural network was ran on my computer, the following tensorboards were generated that outline the steps taken by the neural network in the form of a function call flowchart.



The tensorboard generated by the initial seeded data and my modified data look fairly similar, with line thickness and function calls appearing in the same places. As the neural network trained, the concordance index on validation data approached the optimal value we were hoping for, and the discordant index decreased accordingly too. This means that over time, our network became better at making predictions and became more accurate.

```
[ ]  sample = train_test_split(x_test, y_test, event_test, time_test,
                               test_size=30, stratify=y_test, random_state=89)
     x_sample, y_sample, event_sample, time_sample = sample[1::2]

     sample_pred_fn = make_pred_fn(x_sample)
     sample_predictions = np.array([float(pred["risk_score"])
                                    for pred in estimator.predict(sample_pred_fn)])

     sample_surv_fn = breslow.get_survival_function(sample_predictions)

     plt.figure(figsize=(6, 4.5))
     for surv_fn, class_label in zip(sample_surv_fn, y_sample):
         risk_group = risk_score_assignment.loc[class_label, "risk_group"]
         plt.step(surv_fn.x, surv_fn.y, where="post",
                  color=f"C{class_label}", linestyle=styles[risk_group])

     plt.ylim(0, 1)
     plt.ylabel("Probability of survival $P(T > t)$")
     plt.xlabel("Time $t$")
     plt.grid()
```





As seen above, the prediction confirmed that the images corresponding to low-risk groups decrease slowly and steadily over time without much loss, and the images in higher-risk groups decrease more drastically but flatten out over time. This corresponds to real-world data, and the data in my results appear to be more accurate than the initial data. I believe this because the solid lines (low-risk group) are more tight together on the upper part of the graph, meaning groups less at risk are more likely to experience the same event encounter rate.

Future Work

While the results of this neural network may at first appear to only indicate the rate at which handwritten digits experience events based on their risk factor, the methods that go into achieving these results can point researchers in the right direction regarding cancer diagnoses and neural networks. There has already been a great deal of research done on this subject, and neural networks are becoming faster every day as technology progresses, so finding new ways to train them and produce results such as these provide insight as to how to make other networks faster.

I would like to expand this neural network to use real clinical data and train it that way, much like how we used skin lesion data to train a similar network using Harvard data. I would also like to research further the differences between networks, as each convolutional neural network are implemented in a similar fashion, but the algorithms and data structures used within those networks can vary a great deal. It would also be valuable to see how other kinds of neural networks perform on MNIST data.

I was also only able to get this to work on Google Collaborate. From there, I downloaded the .ipynb file and upload that to GitHub, but I wasn't actually able to get it to run on my local machine due to issues I outlined in the README and the top of my .ipynb file. Because of certain Python modules not being compatible with various aspects of my installed software, I was not able to effectively run the notebook. I would like to see how quickly I can generate results using my hardware, since I have a fairly decent computer.

Conclusion

Using a convolutional neural network, I was able to use MNIST handwritten data to predict how different risk groups react to certain levels of risk. I assigned a risk factor and a risk group to digits 0 through 9 and watched how those "patients" interact with events in both censored and uncensored data. This allows for unreported or incomplete results to be accounted for and makes the data we collect more robust. In my testing, I found that this data was effective in tracking when events occur and how likely those events are to occur for different risk groups. The higher the risk, the more likely that patient is to experience an event.

Although this data is not clinical data, it still provides pertinent insight as to how neural networks are used to diagnose patients using microscopic cell imaging. This method can be used to track patients' recovery rate, death rate, and allow doctors to give more accurate prognoses with the aid of computing. This network also trains itself over time, making it more accurate as it progresses over time. With the assistance of manual data manipulation and human error checking, we can use neural networks to advance the medical field, saving lives and improving our knowledge as to how cancer affects patients.

References

Kong, Jun. “Computer-Assisted Grading of Neuroblastic Differentiation.” *An Error Occurred Setting Your User Cookie*, 2008, www.archivesofpathology.org/doi/full/10.1043/1543-2165(2008)132%5B903%3ACGOND%5D2.0.CO%3B2.

LeCun, Yann. “Gradient-Based Learning Applied to Document Recognition .” *Gradient-Based Learning Applied to Document Recognition - IEEE Journals & Magazine*, 1998, ieeexplore.ieee.org/abstract/document/726791.

LeCun, Yann. “THE MNIST DATABASE.” *MNIST Handwritten Digit Database, Yann LeCun, Corinna Cortes and Chris Burges*, yann.lecun.com/exdb/mnist/.

Mobadersany, Pooya, et al. “Predicting Cancer Outcomes from Histology and Genomics Using Convolutional Networks.” *PNAS*, National Academy of Sciences, 27 Mar. 2018, www.pnas.org/content/115/13/E2970.

Pölsterl, Sebastian. “Survival Analysis for Deep Learning.” *Sebastian Pölsterl*, Sebastian Pölsterl, 11 Oct. 2019, k-d-w.org/blog/2019/07/survival-analysis-for-deep-learning/.