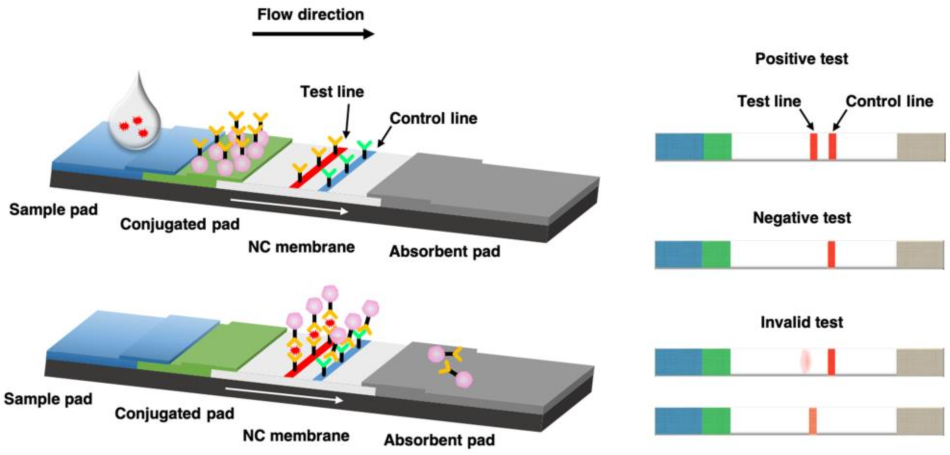
Rapid Diagnostic Test Prediction via Convolutional Neural Network

Daniel Leon, MS Applied Math, BS Chemical Engineering

BE BOUNDLESS

Introduction – Rapid Diagnostic Devices

> Rapid diagnostic devices (RDTs) have a broad application set.



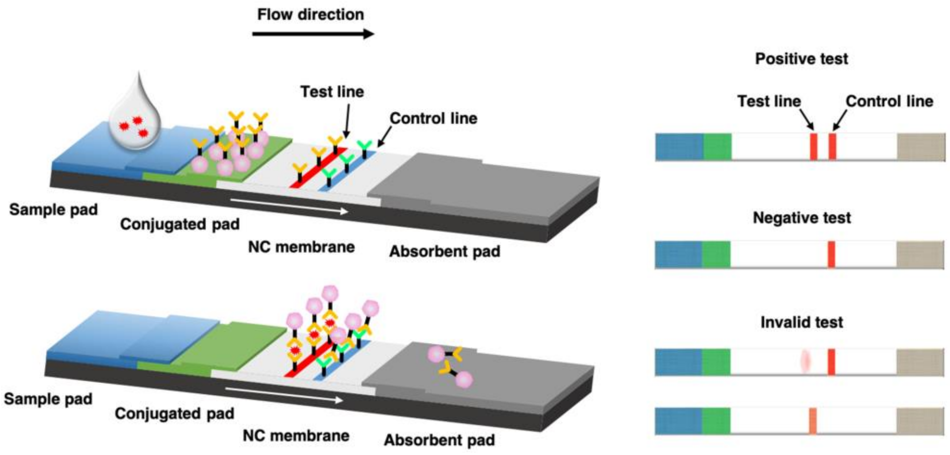
[1]

UNIVERSITY of WASHINGTON

[1]Wesley Wei-Wen Hsiao et al. “Recent Advances in Novel Lateral Flow Technologies for Detection of COVID-19”. In: Biosensors 11.9 (2021). ISSN: 2079-6374. DOI: 10.3390/bios11090295. URL: https://www.mdpi.com/2079-6374/11/9/295.

Objective

> The need for quick and accurate interpretation of RDTs could be advanced via machine learning techniques.



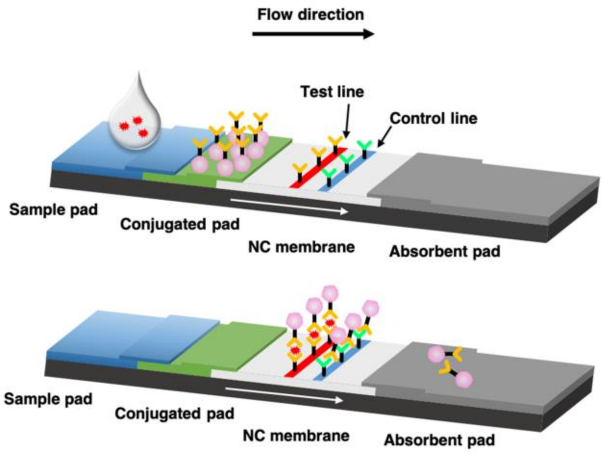
[1]

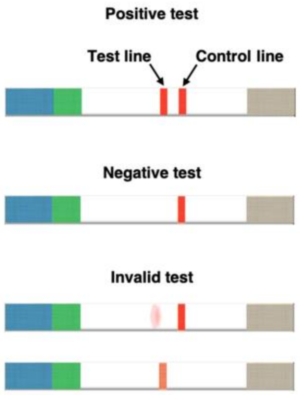
UNIVERSITY of WASHINGTON

[1]Wesley Wei-Wen Hsiao et al. "Recent Advances in Novel Lateral Flow Technologies for Detection of COVID-19". In: Biosensors 11.9 (2021). ISSN: 2079-6374. DOI: 10.3390/bios11090295. URL: https://www.mdpi.com/2079-6374/11/9/295.

Data Collection

> Smartphones offer a practical method for image collection, but user conditions will vary in terms of model of phone, lighting conditions, orientation of images, etc.





[1]

UNIVERSITY of WASHINGTON

[1]Wesley Wei-Wen Hsiao et al. "Recent Advances in Novel Lateral Flow Technologies for Detection of COVID-19". In: Biosensors 11.9 (2021). ISSN: 2079-6374. DOI: 10.3390/bios11090295. URL: https://www.mdpi.com/2079-6374/11/9/295.

The Dataset

> For this project, I had access to a dataset of over 2,000 rapid COVID-19 RDTs

> The images were taken of RDT’s with a series of tests using serial dilution (10 levels) of target antigen (Ag)

Concentrations (ng/mL):

- 10
- 5
- 2.5
- …
- 0.0195
- 0
- Invalid (no control line)

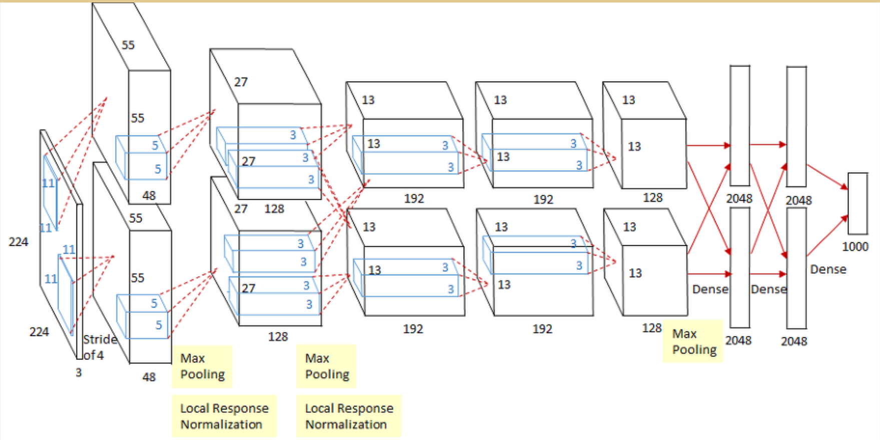
The Dataset – Preprocessing

> Preprocessing was needed, first to crop out the relevant region, then to enhance the contrast

WASHINGTON

Convolutional Neural Network Model

- Using the PyTorch framework, I set up and trained from scratch a convolutional neural network that was based on the now famous AlexNet architecture



[2]

[2]Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: Advances in Neural Information Processing Systems. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf, Krizhevsky et al.

Convolutional Neural Network Model

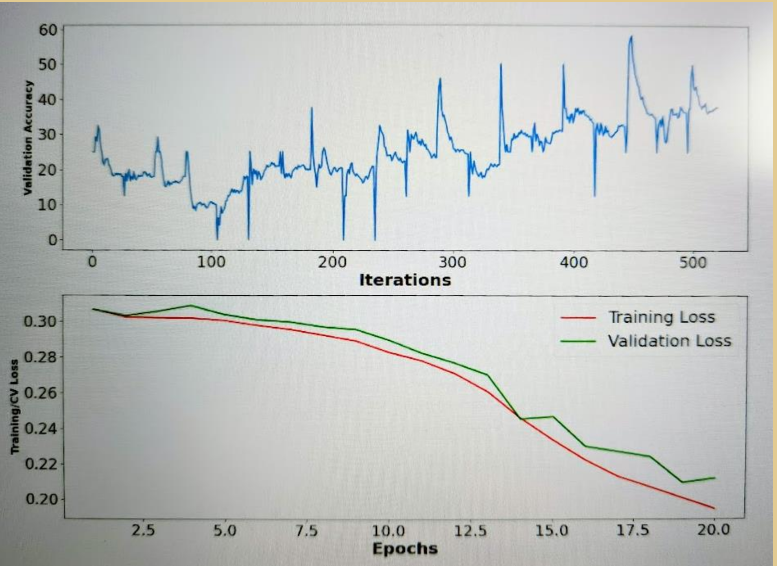
- This system allowed enough versatility to begin to tackle this problem, while still being small enough to easily train on my home GPU
  - Optimizer: stochastic gradient decent with momentum
  - Activation functions: Leaky ReLU ($\alpha = 0.05$)
  - Regularization: 15% randomized weight dropout

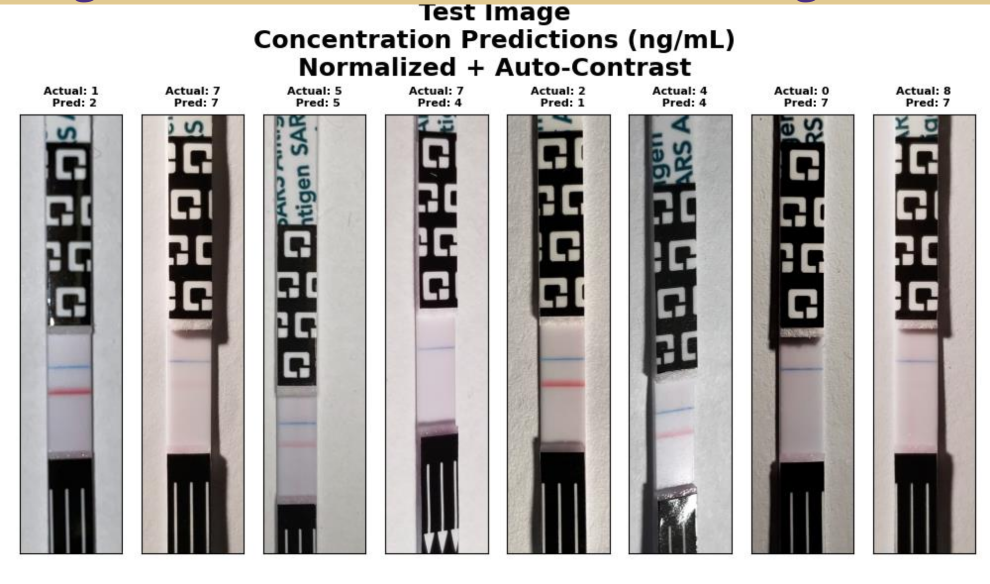
Classification Problem

- The problem was initially formatted as a classification problem, with the final layer of the network predicting 1 of 12 classes
  - Used cross-entropy loss function
    - $l_{CrossEntropy} = -\Sigma_{c=1}^{C} w_c \log \frac{\exp(x_{n,c})}{\sum_{i=1}^{C} \exp(x_{n,i})} y_{n,c}$

Classification Problem

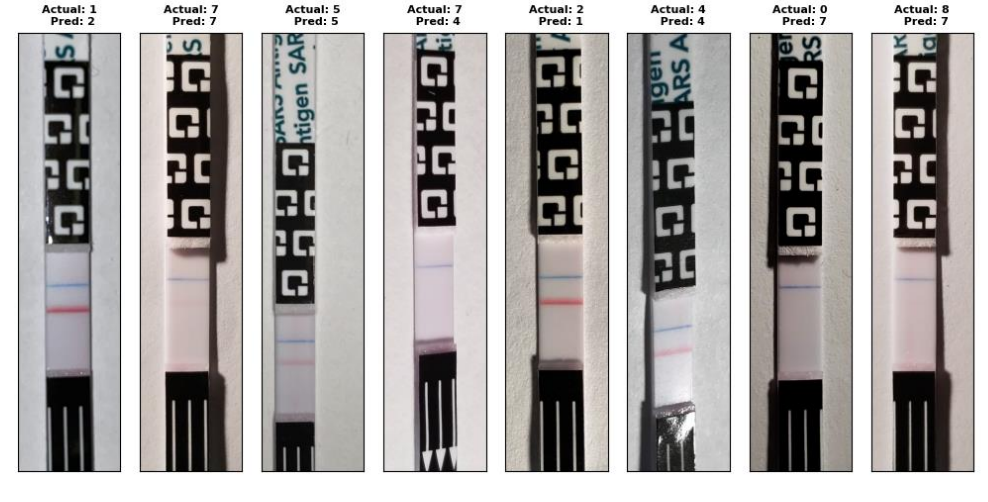
- After 20 epochs, I was able to achieve 35% accuracy.
  - Way better than randomly guessing 1 of 12 classes (8.3% for random guess)





Classification Problem

- Classification doesn't consider the proximity of one class to another
- By altering the approach, I hypothesized that the outcomes could be improved

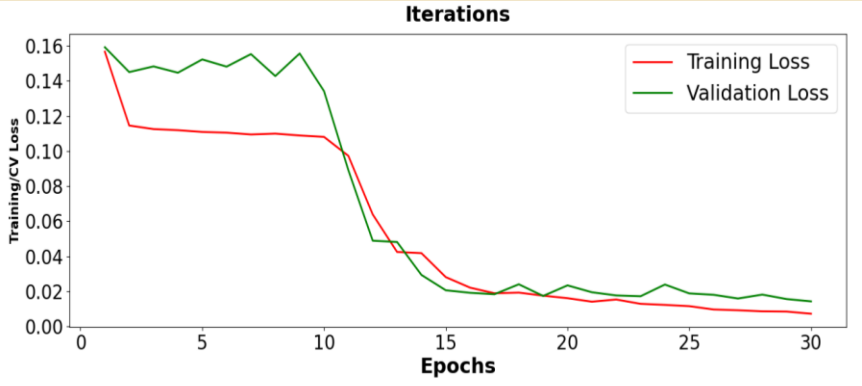


Regression Problem

- Reconfiguring the model to the regression format goes from predicting which of the 12 classes is probable, to predicting a single outcome in terms of reagent concentration
  - Used mean-squared error loss function
    - $l_{MSE} = \frac{1}{n} \Sigma_{\mathrm{i}=1}^{\mathrm{n}} (y_i - \hat{y}_i)^2$

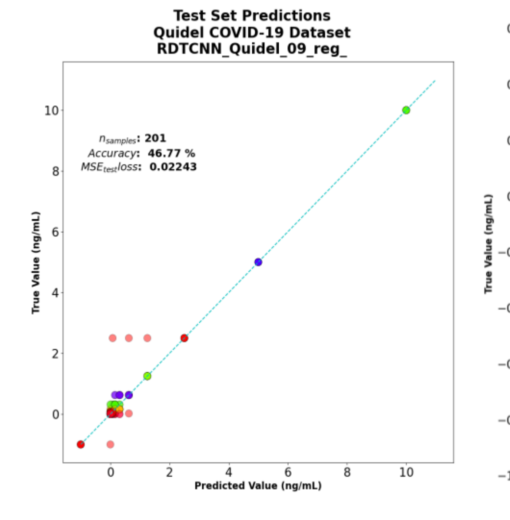
Regression Problem

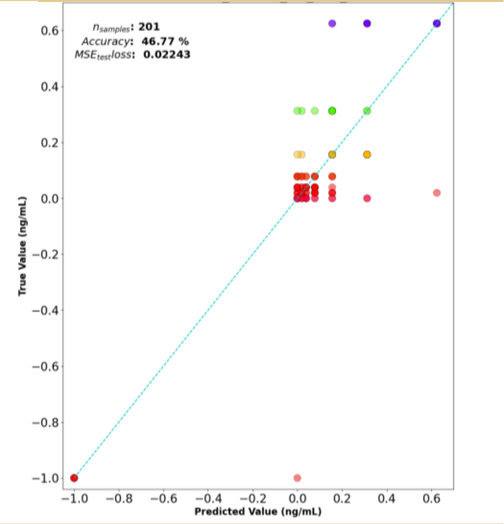
- After 30 epochs achieved 46.8% accuracy and an MSE loss of 0.022
  - ~35% of misclassified tests were 1 level off
  - ~30% of misclassified tests were 2 levels off
  - Model correctly predicted invalid tests in all but 1 case

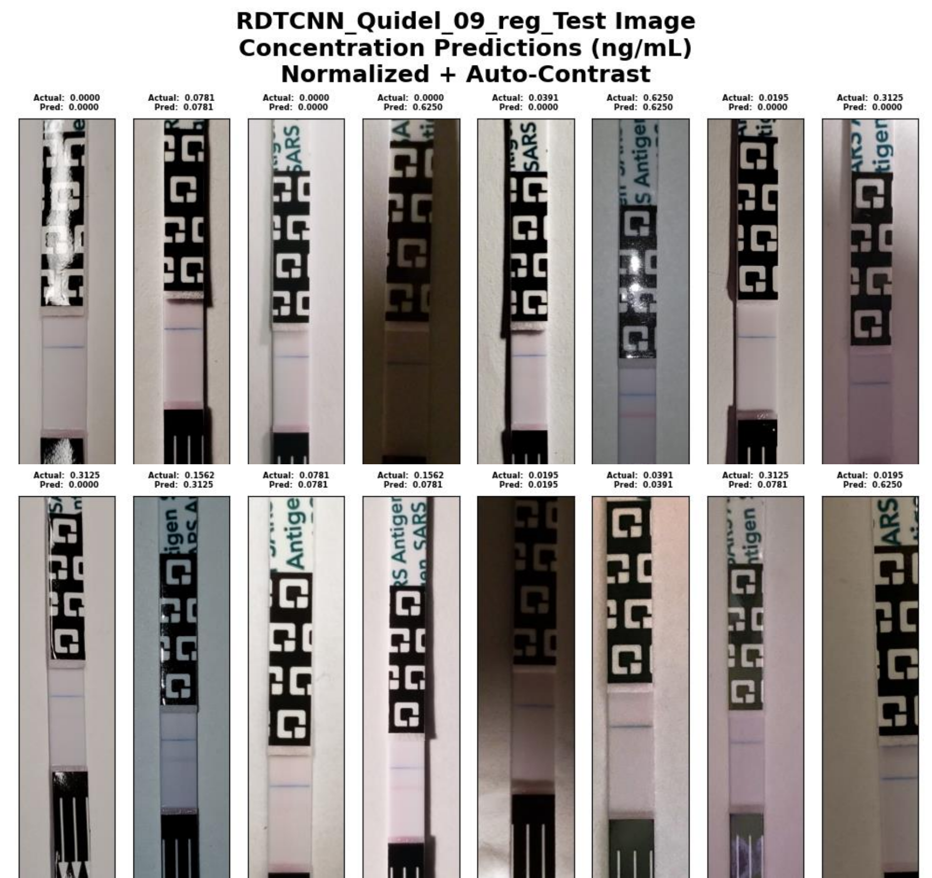


Regression Problem

> Model performed very well in the high concentration range, and moderately well in the very low concentration range

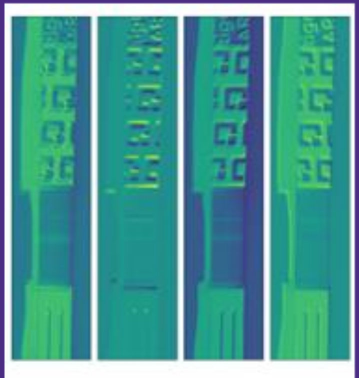


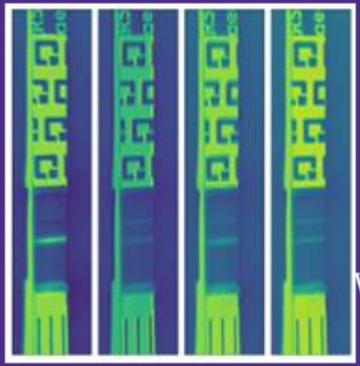
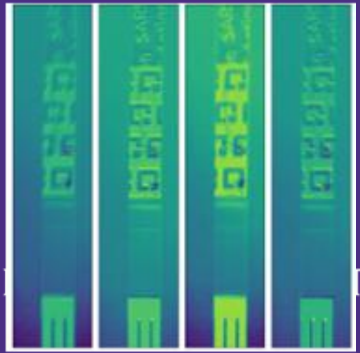




Digging into the Model

- We are looking at the first 16 filters in the first layer of the CNN.
  - This gives an idea of what the features look like to the network
  - The test lines are enhanced in this layer, which is what we want
  - The labels on the strips are also enhanced, which is not what we want



Considerations

- The dataset was constructed to simulate real-world difficulties one would encounter when having patients submit images with their own smartphone
- Practical limitations of the project constrained the size of the model that I could work with
  - Was only able to work with batches of 4 images at a time

Conclusions

- I showed that we can use CNN's to interpret smartphone images of RDT's
  - Under classification format, I achieved 35% accuracy after 20 epochs of training
  - Under the regression format, I achieved 47% accuracy and an MSE loss of 0.022
- A system similar to this could be deployed by healthcare providers to provide accurate quantification of RDT's
  - This could reduce the resources needed to provide actionable information to doctors and other healthcare providers
  - This could be implemented in low-resource settings

Next Steps

- I would start with a larger base network, such as ResNet
- Training on a cluster would allow for more sophisticated optimizers, would allow larger batch sizes, which results in faster training speeds
- More sophisticated image pre-processing would remove the unnecessary noise of RDT labels and shadows, and reduce the total image size, which could have a huge impact on training speed and model accuracy

Thank you

- dannylleon@gmail.com
- linkedin.com/in/daniel-leon86/

UNIVERSITY of WASHINGTON