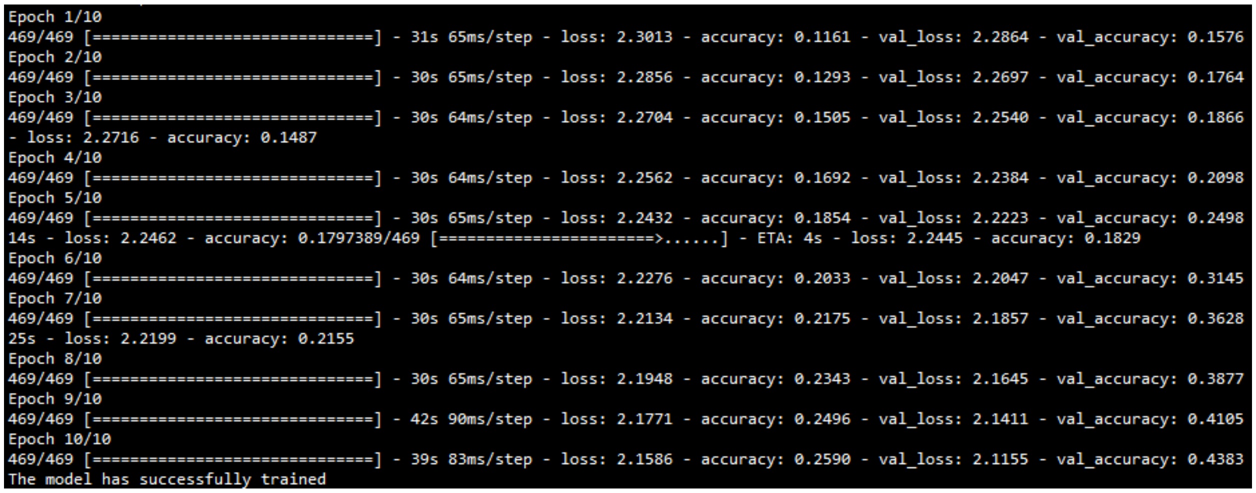
Kevin Rivera
ELE 547
Assignment 4

ReLU vs. Sigmoid

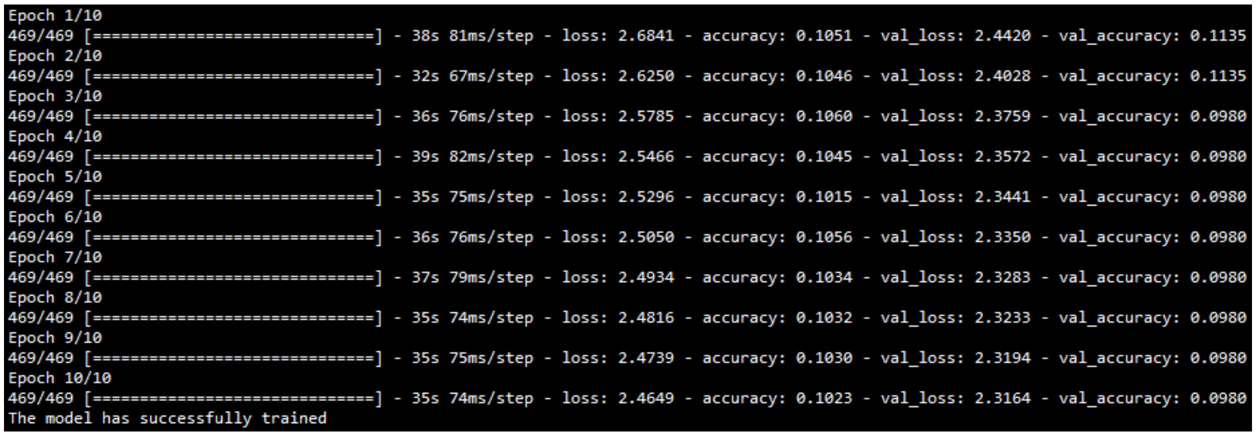
The rectified linear unit (ReLU) activation function produced a model that was ~44% accurate after when 10 epochs were executed (Fig. 1). In order to increase the accuracy to around 90%, 50 epochs were required and this took about 40 minutes to execute. The Sigmoid activation function did not generate very useful results and produced a model with an accuracy of roughly 10% after 10 epochs (Fig. 2). When the number of epochs were increased, the accuracy did not improve at all. This behavior shown by the sigmoid activation function seems to be due to the vanishing gradient problem that can be encountered when the model contains too many hidden layers. In an attempt to alleviate this issue, I reduced the number of hidden layers but the accuracy did not seem to improve at all unfortunately.

While the ReLU and Sigmoid activation models produced models with very different accuracy values, they both seemed to compile at a similar rate (~30 seconds/epoch) which is unexpected since the Sigmoid activation model should have converged faster.



```
Epoch 1/10
469/469 [==============================] - 31s 65ms/step - loss: 2.3013 - accuracy: 0.1161 - val_loss: 2.2864 - val_accuracy: 0.1576
Epoch 2/10
469/469 [==============================] - 30s 65ms/step - loss: 2.2856 - accuracy: 0.1293 - val_loss: 2.2697 - val_accuracy: 0.1764
Epoch 3/10
469/469 [==============================] - 30s 64ms/step - loss: 2.2704 - accuracy: 0.1505 - val_loss: 2.2540 - val_accuracy: 0.1866
- loss: 2.2716 - accuracy: 0.1487
Epoch 4/10
469/469 [==============================] - 30s 64ms/step - loss: 2.2562 - accuracy: 0.1692 - val_loss: 2.2384 - val_accuracy: 0.2098
Epoch 5/10
469/469 [==============================] - 30s 65ms/step - loss: 2.2432 - accuracy: 0.1854 - val_loss: 2.2223 - val_accuracy: 0.2498
14s - loss: 2.2462 - accuracy: 0.1797389/469 [=======================>......] - ETA: 4s - loss: 2.2445 - accuracy: 0.1829
Epoch 6/10
469/469 [==============================] - 30s 64ms/step - loss: 2.2276 - accuracy: 0.2033 - val_loss: 2.2047 - val_accuracy: 0.3145
Epoch 7/10
469/469 [==============================] - 30s 65ms/step - loss: 2.2134 - accuracy: 0.2175 - val_loss: 2.1857 - val_accuracy: 0.3628
25s - loss: 2.2199 - accuracy: 0.2155
Epoch 8/10
469/469 [==============================] - 30s 65ms/step - loss: 2.1948 - accuracy: 0.2343 - val_loss: 2.1645 - val_accuracy: 0.3877
Epoch 9/10
469/469 [==============================] - 42s 90ms/step - loss: 2.1771 - accuracy: 0.2496 - val_loss: 2.1411 - val_accuracy: 0.4105
Epoch 10/10
469/469 [==============================] - 39s 83ms/step - loss: 2.1586 - accuracy: 0.2590 - val_loss: 2.1155 - val_accuracy: 0.4383
The model has successfully trained
```

Fig. 1: Result of training model using a ReLU activation function. Accuracy increases after each successive epoch.



```
Epoch 1/10
469/469 [==============================] - 38s 81ms/step - loss: 2.6841 - accuracy: 0.1051 - val_loss: 2.4420 - val_accuracy: 0.1135
Epoch 2/10
469/469 [==============================] - 32s 67ms/step - loss: 2.6250 - accuracy: 0.1046 - val_loss: 2.4028 - val_accuracy: 0.1135
Epoch 3/10
469/469 [==============================] - 36s 76ms/step - loss: 2.5785 - accuracy: 0.1060 - val_loss: 2.3759 - val_accuracy: 0.0980
Epoch 4/10
469/469 [==============================] - 39s 82ms/step - loss: 2.5466 - accuracy: 0.1045 - val_loss: 2.3572 - val_accuracy: 0.0980
Epoch 5/10
469/469 [==============================] - 35s 75ms/step - loss: 2.5296 - accuracy: 0.1015 - val_loss: 2.3441 - val_accuracy: 0.0980
Epoch 6/10
469/469 [==============================] - 36s 76ms/step - loss: 2.5050 - accuracy: 0.1056 - val_loss: 2.3350 - val_accuracy: 0.0980
Epoch 7/10
469/469 [==============================] - 37s 79ms/step - loss: 2.4934 - accuracy: 0.1034 - val_loss: 2.3283 - val_accuracy: 0.0980
Epoch 8/10
469/469 [==============================] - 35s 74ms/step - loss: 2.4816 - accuracy: 0.1032 - val_loss: 2.3233 - val_accuracy: 0.0980
Epoch 9/10
469/469 [==============================] - 35s 75ms/step - loss: 2.4739 - accuracy: 0.1030 - val_loss: 2.3194 - val_accuracy: 0.0980
Epoch 10/10
469/469 [==============================] - 35s 74ms/step - loss: 2.4649 - accuracy: 0.1023 - val_loss: 2.3164 - val_accuracy: 0.0980
The model has successfully trained
```

Fig. 2: Result of training model using a sigmoid activation function. Accuracy does not increase after each successive epoch.

PC vs. Raspberry Pi

The graphical user interface portion of the program ran relatively smoothly on my PC and using the ReLU model derived after 50 epochs and the accuracy of the predictions was very high (Fig. 3). Running the same graphical user interface program on the Raspberry Pi 4 was much more difficult due to setup issues. The final solution to these issues was to install Tensorflow v.2.3.1 using "pip3 install https://github.com/lhelontra/tensorflow-on-arm/releases/download/v2.3.0/tensorflow-2.3.0-cp37-none-linux_armv7l.whl" and this allowed me to utilize the default version of Keras that is installed using "pip3 install keras" (v.2.3.1) rather than a downgraded version. When the program was run on the Raspberry Pi 4 using the same model that was generated on the PC, the results were similar in terms of accuracy but took slightly longer to generate predictions since it is not as powerful in terms of its processing capability. I also had an HD screen hat installed on the Raspberry Pi 4 which caused heat to build up slightly and this also increased the prediction times the longer the Raspberry Pi 4 was left on but only by a few fractions of a millisecond at most.

For both the PC and Raspberry Pi 4 the Sigmoid function predicted the handwritten number to be "0" with >80% statistical confidence which was somewhat entertaining but useless.

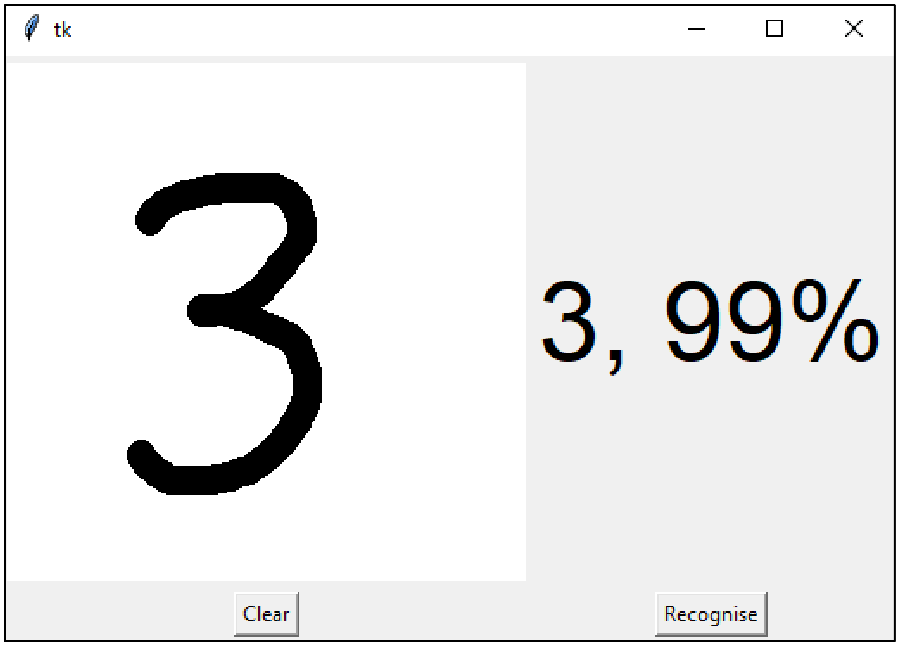


Fig. 3: Graphical user interface showing a prediction based on a hand-drawn number. This model utilized a ReLU activation function run for 50 epochs to achieve a high accuracy. This is a PC result.