# Multi-label Classification of Image Data

Jay Abi-Saad - 260801368
Julien Phillips - 260804197
Rayan Osseiran - 260803963

December 13, 2020
McGill University
COMP 551

**Abstract.** The purpose of this project was to implement a deep neural network in-order to perform multi-label classification on a modified version of the MNIST dataset. The dataset is made up of sample images which contain 1-5 handwritten digits that can each be classified into 11 classes. Our multi-label classification approach was to split the input image into individual digit images, which we then feed into a deep neural network trained on single handwritten digit recognition. We performed grid search to obtain the best three models, and then conducted 5-fold cross validation to determine the best of the three. The final accuracy obtained on the test data was 99.392%.

## *I. INTRODUCTION*

This project used a modified version of the MNIST digits dataset where each image contains 1-5 handwritten digits. The task was to implement a deep neural network capable of performing multi-label classification on the dataset. In order to accomplish this, the approach taken was to split each image in the dataset into sub-images, each containing a single digit. Our neural network was then trained on the individual digit images and their corresponding labels from the training set. Neural network selection was done based on the results of the parameter tuning process, which conducted a grid search that evaluates the best overall combination of activation function, optimizer, learning rate and batch size. The best three models from the parameter tuning process were tested further and verified by doing 5-fold cross validation. The final neural network consists of three 2D convolution layers with a filter size of 32, and an ELU activation function. This is followed by two dense layers, one with an ELU activation function, and the final with a sigmoid activation function, that ensures output is between 0 and 1. In order to construct the predictions for the test set, each image in the test set was split into its corresponding digits and fed to the neural network to make a prediction on each digit image. The default prediction, composed of only 10's (the non-digit class), was overwritten starting at the front with the neural networks predictions, which resulted in the final prediction label of the test image. This approach yielded results of 99.392% accuracy on the test set.

## *II. DATASETS*

We used a modified version of the MNIST dataset. The data was composed of sample images that each contained 1-5 handwritten digits. The labels set was composed of 5-digit lists that mapped to the 5 possible digits in the corresponding sample image. The dataset contains 11 classes, one for each possible digit, and if the number of digits in an image is less than 5, a blank digit is labeled as 10. As previously mentioned, we implemented a single-label deep neural network that classifies single digits. As a result, data preprocessing was required in-order to split and enhance images into single-digit images. In-order to split the input images, we initially set out to use built-in OpenCV methods that automatically detect digits and draw bounding boxes around them. However, we found that this approach wasn't accurate and didn't leverage the fact that the digits in the input images are aligned along the middle horizontal axis of the image. Moreover, the OpenCV approach was less optimizable than a custom implementation. For these reasons, we decided to implement our own split image method. Before starting our sliding window operation, we apply a kernel blur which has a smoothing effect on the edges of the digits. Then, to increase the definition of each digit, we apply binary window thresholding which sets pixels with intensity greater than a given upper threshold to 255 (white), and lower than a given lower threshold to 0 (black). It is to be noted that these image manipulation techniques are only used for digit detection in the sliding window loop. Once we discover the coordinates and dimensions of a given digit, we extract it from the original unaltered image rather than the smoothed and thresholded image. Our approach is simple and fast, as it only consists of sliding a window from left-to-right along the middle horizontal axis of the image. The window has a dimension of 10 by 16 pixels, which was deduced to be the average size of a digit by trial and error. To ensure that we don't detect duplicate digits, our technique pays special attention to the pixel intensity values of the left side of the sliding window. We only extract a digit from the window if at any given iteration, the pixels on the left side of the current window are not all black, while the pixels on the left side of the previous window were all black.

## *III. RESULTS*

The initial model structure chosen was based on an existing model found online that was capable of performing multi-label image classification on a modified version on the MNIST fashion dataset, where four clothing items appeared in the dataset [1]. This neural network was chosen for two reasons; the first being its successful classification using an image dataset, the second being its potential for adaptability to support training directly on the modified MNIST digits dataset rather than breaking the images into its corresponding digits. Given that the initial approach of splitting the dataset images worked with high accuracy, it was not necessary to attempt the alternative. The model itself was extensively modified. Three 2D convolution layers were employed instead of initial five in-order to reduce model complexity and chances of overfitting the training data. Moreover, following the convolution layers, a flatten layer was added in-order to linearize data prior to the dense layer. An additional dense layer was added in-order to improve classification performance, as well as a dropout for regularization with a probability of 0.5. These modifications improved validation accuracy by up to 2% in our testing. In all testing, our selected loss function was categorical cross entropy. Moreover, the number of epochs was set to 50 in all tests, with early-stopping implemented in-order to prevent overfitting. The patience for early-stop was set to 5 and is based on the validation loss. Further optimization of the model was done during the hyperparameter tuning process, which yielded the three best performing models based on activation function, optimizer, learning rate and batch size. Optimization was done using Talos, a Tensorflow specific library that performs grid search on a provided set of parameters. The primary convenience of Talos is that it provides built-in visualization features for the results of the grid search. The results of the optimization process are shown in Table A.1 in the Appendix. The top three models from the parameter grid search were then further validated using 5-fold cross validation to select the final model. The results of the 5-fold cross validation were that the best model parameters are an ELU activation function, stochastic gradient descent optimizer, learning rate of 0.01 and batch size of 64. The ELU activation function is applied to the convolution layers, as well as the first dense layer. Moreover, final training occurred on a test-validation split of 7.5%. The primary reasoning for using a small number is that 20% of the dataset is already being used for testing. Indeed, there are 14,0000 elements in the test set and 56,000 elements in the training set. As a result, this leaves us with an overall data split of 26% for testing and validation and 74% for training when factoring in our validation samples. With that being said, Figures A.1. highlights the change in validation and training accuracy vs the number of epochs during training. Figure A.2. does the same but compares it to the loss. For Figure A.1, as expected, we notice that the accuracy for both training and validation increases logarithmically throughout the epochs, and seems to plateau at a certain point. For Figure A.2, as expected, the loss for both training and validation decreases logarithmically until it plateaus.

Using the optimal model discussed above, we were able to achieve 99.392% accuracy on the test data. This accuracy is lower than the validation accuracy of the model, and one possible explanation for this is the actual split image process. Whilst unlikely, it is possible that the algorithm fails to detect all digits in an image, or possibly even overestimates it. This does not cause issues in-terms of mapping predictions to their corresponding labels, but may result in incorrect predictions. As an example, we may only output [3,5,10,10,10] in an image instead of [3,5,6,10,10] because the six was not correctly extracted from the split image function. When determining validation and training accuracy, the above prediction would be partially correct as all digits are correct aside from the six. In the test set, we expect the entire sequence to be correct, so this would be an incorrect prediction from a test perspective.

## *IV. DISCUSSION & CONCLUSION*

Our approach of breaking down the images into their subsequent digits ended up being a very successful approach for this project, yielding a high accuracy on the test set. For future reference, it would be better to improve the performance of the split image function in-order to reduce data loss. Whilst this is not necessarily problematic from a training perspective, it can lead to increased error with test sets, particularly because the task at hand is to identify an entire sequence in an image, and not just a single digit. Our grid search for tuning parameters was extremely helpful in finding the best parameters for the model, and cross-validation helped us validate which model was best of the top three. One area of improvement for the future could be regarding the patience for early stop. We observed a variety of different behaviours and accuracy depending on what the patience was set to. It represents a delicate balance, where we do not want to overfit the data, but at the same time do not want to train a model with sub-optimal performance. Figures A.1. and A.2. can also help indicate that we possibly should have stopped sooner. One potential approach here could be to treat it as a hyperparameter and to perform grid search, as well as cross-validation in-order to determine an optimal value.

## *V. STATEMENT OF CONTRIBUTIONS*

**Julien:** Worked on neural network model, parameter tuning, and contributed to everything else.
**Jay:** Worked on splitting images, neural network models, and contributed to everything else.
**Rayan:** Worked on neural network model, parameter tuning, and contributed to everything else.

## *VI. REFERENCES*

[1] Franky, “Multi-Label Classification and Class Activation Map on Fashion-MNIST,” 06-Aug-2018. [Online]. Available: https://towardsdatascience.com/multi-label-classification-and-class-activation-map-on-fashion-mnist-1454f09f5925.

## *VII. APPENDIX*

| | start | end | duration | round_ epochs | loss | categorical_ accuracy | val_loss | val_categorical _accuracy | activation | batch _size | epochs | lr | optimizer |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 12/11/20-173616 | 12/11/20-174451 | 514.584861 | 41 | 0.04971416 | 0.985594 | 0.03862765 | 0.988285 | relu | 32 | 50 | 0.001 | sgd |
| 1 | 12/11/20-174451 | 12/11/20-174702 | 130.919217 | 9 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 32 | 50 | 0.001 | adam |
| 2 | 12/11/20-174702 | 12/11/20-174912 | 130.034247 | 9 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 32 | 50 | 0.001 | rmsprop |
| 3 | 12/11/20-174912 | 12/11/20-175148 | 155.60377 | 13 | 9.259553 | 0.111565 | 10.00862 | 0.113069 | relu | 32 | 50 | 0.1 | sgd |
| 4 | 12/11/20-175148 | 12/11/20-175315 | 86.614528 | 6 | 10.87373 | 0.098292 | 11.65245 | 0.092619 | relu | 32 | 50 | 0.1 | adam |
| 5 | 12/11/20-175315 | 12/11/20-175441 | 86.034225 | 6 | 10.10175 | 0.111878 | 9.958107 | 0.113069 | relu | 32 | 50 | 0.1 | rmsprop |
| 6 | 12/11/20-175441 | 12/11/20-180429 | 587.717207 | 48 | 0.004474448 | 0.998577 | 0.0129947 | 0.997347 | relu | 32 | 50 | 0.01 | sgd |
| 7 | 12/11/20-180429 | 12/11/20-180558 | 89.091797 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 32 | 50 | 0.01 | adam |
| 8 | 12/11/20-180558 | 12/11/20-180727 | 88.90236 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 32 | 50 | 0.01 | rmsprop |
| 9 | 12/11/20-180727 | 12/11/20-181306 | 338.667486 | 50 | 0.06491539 | 0.98157 | 0.04762641 | 0.985214 | relu | 64 | 50 | 0.001 | sgd |
| 10 | 12/11/20-181306 | 12/11/20-181449 | 103.075534 | 13 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 64 | 50 | 0.001 | adam |
| 11 | 12/11/20-181449 | 12/11/20-181623 | 94.28153 | 12 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 64 | 50 | 0.001 | rmsprop |
| 12 | 12/11/20-181624 | 12/11/20-181824 | 120.149054 | 18 | 0.0157836 | 0.995417 | 0.02602914 | 0.992309 | relu | 64 | 50 | 0.1 | sgd |
| 13 | 12/11/20-181824 | 12/11/20-181912 | 47.90252 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 64 | 50 | 0.1 | adam |
| 14 | 12/11/20-181912 | 12/11/20-182007 | 55.54264 | 7 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 64 | 50 | 0.1 | rmsprop |
| 15 | 12/11/20-182008 | 12/11/20-182339 | 211.689758 | 31 | 0.01637505 | 0.994768 | 0.02005122 | 0.994545 | relu | 64 | 50 | 0.01 | sgd |
| 16 | 12/11/20-182339 | 12/11/20-182429 | 49.352919 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 64 | 50 | 0.01 | adam |
| 17 | 12/11/20-182429 | 12/11/20-182518 | 48.66592 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 64 | 50 | 0.01 | rmsprop |
| 18 | 12/11/20-182518 | 12/11/20-182904 | 226.235889 | 50 | 0.1041958 | 0.970883 | 0.0612982 | 0.980862 | relu | 128 | 50 | 0.001 | sgd |
| 19 | 12/11/20-182904 | 12/11/20-183052 | 108.368313 | 22 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 128 | 50 | 0.001 | adam |
| 20 | 12/11/20-183053 | 12/11/20-183152 | 59.007658 | 12 | 0.02875056 | 0.992555 | 0.03628676 | 0.992995 | relu | 128 | 50 | 0.001 | rmsprop |
| 21 | 12/11/20-183152 | 12/11/20-183335 | 103.589802 | 23 | 0.01455078 | 0.995536 | 0.02294076 | 0.99377 | relu | 128 | 50 | 0.1 | sgd |
| 22 | 12/11/20-183335 | 12/11/20-183405 | 29.872701 | 6 | 8.937178 | 0.103032 | 8.942907 | 0.100399 | relu | 128 | 50 | 0.1 | adam |
| 23 | 12/11/20-183405 | 12/11/20-183436 | 30.408551 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 128 | 50 | 0.1 | rmsprop |
| 24 | 12/11/20-183436 | 12/11/20-183820 | 223.719149 | 50 | 0.01830611 | 0.994358 | 0.01766898 | 0.994157 | relu | 128 | 50 | 0.01 | sgd |
| 25 | 12/11/20-183820 | 12/11/20-183854 | 34.613295 | 7 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 128 | 50 | 0.01 | adam |
| 26 | 12/11/20-183855 | 12/11/20-183930 | 35.151332 | 7 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | relu | 128 | 50 | 0.01 | rmsprop |
| 27 | 12/11/20-183930 | 12/11/20-184923 | 593.210091 | 49 | 0.02751503 | 0.991877 | 0.02705035 | 0.991802 | elu | 32 | 50 | 0.001 | sgd |
| 28 | 12/11/20-184923 | 12/11/20-185216 | 172.926219 | 12 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 32 | 50 | 0.001 | adam |
| 29 | 12/11/20-185216 | 12/11/20-185454 | 157.496901 | 11 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 32 | 50 | 0.001 | rmsprop |
| 30 | 12/11/20-185454 | 12/11/20-185706 | 132.501903 | 11 | 0.0744894 | 0.983984 | 0.06589504 | 0.984827 | elu | 32 | 50 | 0.1 | sgd |
| 31 | 12/11/20-185707 | 12/11/20-185902 | 115.131343 | 8 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 32 | 50 | 0.1 | adam |
| 32 | 12/11/20-185902 | 12/11/20-190223 | 201.06367 | 14 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 32 | 50 | 0.1 | rmsprop |
| 33 | 12/11/20-190223 | 12/11/20-190908 | 404.961273 | 34 | 0.004431519 | 0.998696 | 0.01224777 | 0.997019 | elu | 32 | 50 | 0.01 | sgd |
| 34 | 12/11/20-190908 | 12/11/20-191035 | 87.273266 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 32 | 50 | 0.01 | adam |
| 35 | 12/11/20-191036 | 12/11/20-191201 | 85.955255 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 32 | 50 | 0.01 | rmsprop |
| 36 | 12/11/20-191202 | 12/11/20-191735 | 333.233134 | 50 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 64 | 50 | 0.001 | sgd |
| 37 | 12/11/20-191735 | 12/11/20-191931 | 115.630376 | 15 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 64 | 50 | 0.001 | adam |
| 38 | 12/11/20-191931 | 12/11/20-192150 | 139.52689 | 18 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 64 | 50 | 0.001 | rmsprop |
| 39 | 12/11/20-192150 | 12/11/20-192342 | 111.826531 | 17 | 0.01783651 | 0.994552 | 0.02207091 | 0.994455 | elu | 64 | 50 | 0.1 | sgd |
| 40 | 12/11/20-192342 | 12/11/20-192429 | 46.880621 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 64 | 50 | 0.1 | adam |
| 41 | 12/11/20-192429 | 12/11/20-192531 | 62.102137 | 8 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 64 | 50 | 0.1 | rmsprop |
| 42 | 12/11/20-192532 | 12/11/20-193053 | 321.259715 | 48 | 0.00429293 | 0.99877 | 0.01135102 | 0.997198 | elu | 64 | 50 | 0.01 | sgd |
| 43 | 12/11/20-193053 | 12/11/20-193140 | 47.442314 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 64 | 50 | 0.01 | adam |
| 44 | 12/11/20-193141 | 12/11/20-193228 | 47.532475 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 64 | 50 | 0.01 | rmsprop |
| 45 | 12/11/20-193228 | 12/11/20-193621 | 232.477292 | 50 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 128 | 50 | 0.001 | sgd |
| 46 | 12/11/20-193621 | 12/11/20-193815 | 113.708858 | 23 | 0.01187859 | 0.99684 | 0.02096457 | 0.996304 | elu | 128 | 50 | 0.001 | adam |
| 47 | 12/11/20-193815 | 12/11/20-194018 | 123.377757 | 25 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 128 | 50 | 0.001 | rmsprop |
| 48 | 12/11/20-194018 | 12/11/20-194137 | 79.209861 | 17 | 0.01109967 | 0.996669 | 0.01713703 | 0.994813 | elu | 128 | 50 | 0.1 | sgd |
| 49 | 12/11/20-194138 | 12/11/20-194208 | 30.034448 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 128 | 50 | 0.1 | adam |
| 50 | 12/11/20-194208 | 12/11/20-194238 | 29.973448 | 6 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 128 | 50 | 0.1 | rmsprop |
| 51 | 12/11/20-194238 | 12/11/20-194626 | 228.654663 | 50 | 0.009965711 | 0.997108 | 0.0109862 | 0.997228 | elu | 128 | 50 | 0.01 | sgd |
| 52 | 12/11/20-194627 | 12/11/20-194702 | 35.505917 | 7 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 128 | 50 | 0.01 | adam |
| 53 | 12/11/20-194702 | 12/11/20-194737 | 34.847946 | 7 | 1.19E-07 | 0.097398 | 1.19E-07 | 0.098134 | elu | 128 | 50 | 0.01 | rmsprop |

**Table A.1.** Raw results of hyperparameter optimization through grid search.The optimal three results are highlighted in orange. 5-fold cross-validation was performed on these collections of parameters in-order to select the optimal model. Finally, the optimal configuration in CV was selected based on the highest average validation accuracy.

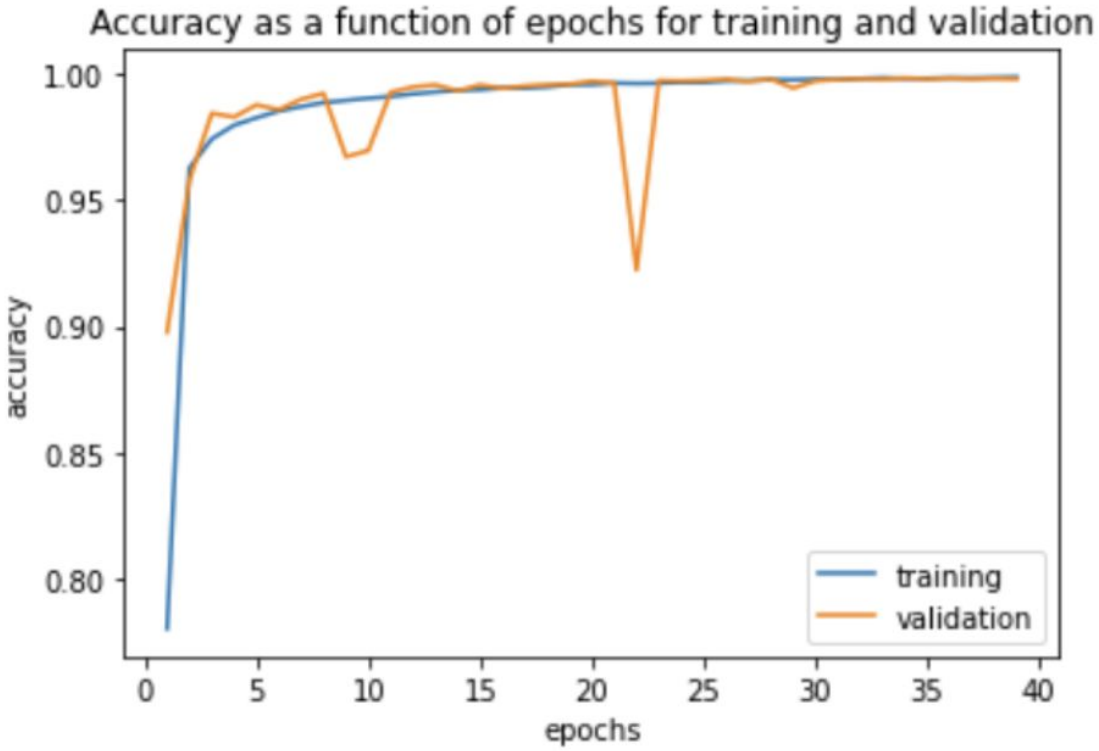


**Figure A.1.** Change in training and validation accuracy as a function of the number of epochs.

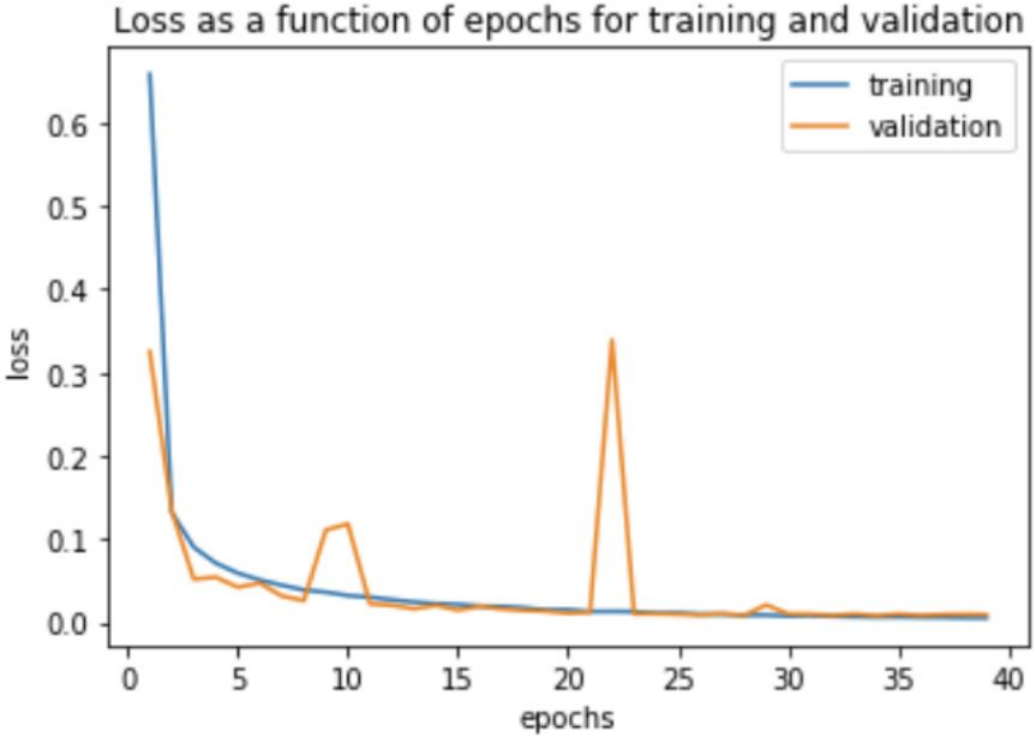


**Figure A.2.** Change in training and validation loss as a function of the number of epochs.