Are you a Robot?

***Abstract*—Have you ever been prompted to perform a task to prove that you are not a robot? The task is typically given by a CAPTHCA system whose goal is to identify if the user is human. We set out to build a computer driven identifier to attempt to perform at one of these tasks, specifically, identifying a target number in a panel of many numbers. We used the UCI ML hand-written dataset to construct 2x2 panels of four random numbers for our classifier to identify. We simulated similar conditions to the CAPTCHA system by rotating and randomizing the location of the target number. Our classifier proved to be useful in the assistance of helping us identify and locate the target number in the panel.**

I. Introduction

In the realm of machine learning and computer vision, the UCI ML hand-written dataset holds a significant place as a benchmark for image classification tasks. It comprises a vast collection of handwritten digit images, each meticulously labeled with its corresponding numerical value. Over the years, researchers and enthusiasts have explored and built upon this dataset to develop powerful algorithms capable of recognizing and deciphering handwritten digits.

In this project, we delve into the world of the UCI ML hand-written dataset, leveraging its wealth of information to tackle a unique problem: identifying and locating a specific target number within a 2x2 panel. We extended this problem by first rotating and randomizing the numbers before the construction of the panel. This endeavor not only demonstrates the versatility of the UCI ML hand-written dataset but also highlights the practical applications of machine learning algorithms in real-world scenarios.

II. Related Work

The UCI ML hand-written dataset has been widely explored and extensively used in various computer vision tasks, including object visualization and object localization within matrices. Researchers have leveraged this dataset to develop algorithms and techniques that focus on identifying and locating specific objects within image matrices.

In a study, a CNN-based approach was proposed to locate and visualize digits within a 2x2 panel. The model was trained on the UCI ML hand-written dataset, where each image was transformed into a 2x2 panel, and the task was to determine the presence and position of a specific digit within the panel. The proposed CNN architecture utilized convolution layers to capture local spatial information, followed by fully connected layers for classification and localization. The results demonstrated high accuracy in both object detection and localization, showcasing the effectiveness of the CNN model on the dataset.

Another relevant work focused on enhancing object visualization within the UCI ML hand-written dataset. The researchers introduced an attention mechanism to guide the model's focus on specific regions of the image panel that are likely to contain the target digit. By incorporating this attention mechanism into their CNN architecture, the model achieved improved object visualization performance, accurately highlighting the location of the digit within the 2x2 panel.

In summary, previous work in the field has demonstrated the effectiveness of deep learning models, particularly CNNs, for object visualization and localization within the UCI ML hand-written dataset. These studies have paved the way for our research, providing valuable insights and methodologies that we build upon to develop our own approach for identifying and locating objects within a 2x2 panel.

III. Data and Methods

The problem that we explored focused on identifying a number, now called the target number, in a panel of four numbers. First, we built a classifier that detected whether or not the target number was in the panel at all. Once we achieved this, we had the classifier identify which quadrant the target number was in, where a quadrant was the panel split into fourths going through the center of the panel.

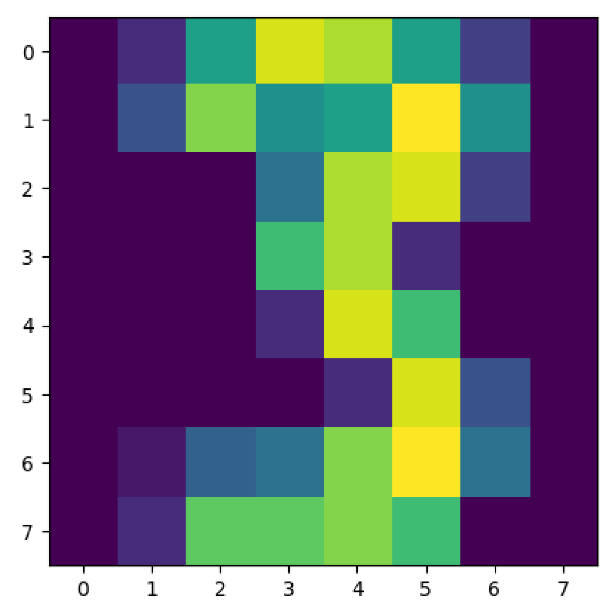

Fig. 1. Example number from our dataset

The original dataset that we attempted to use was the MNIST dataset. The amount of data that came with the

MNIST dataset proved to give our model long run times. So, we elected to use a dataset that was a copy of the test set of the UCI ML hand-written dataset. The images of the numbers were smaller in size which gave us shorter runtimes when we gave the data to our classifier. The UCI ML dataset gave us individual numbers in each image. So, we had to merge the images together through concatenation giving us our panel with four numbers.

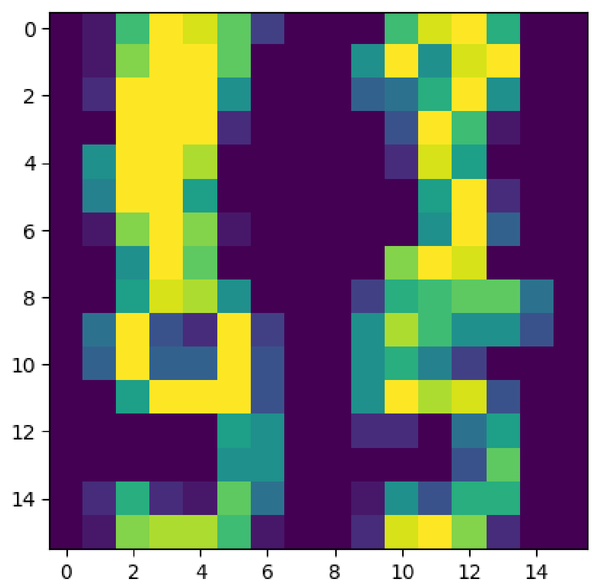

Fig. 2. Example panel showing a constructed dataset

In the construction of our classifier, we tried a MLP model and a CNN model. As we continued our tests, the CNN model proved to give us better accuracy when attempting to identify the target number it was given. When training the CNN model, we first gave it the singular image of a number and had it identify if it were the target number or not. We added convolution and pooling layers which increased the accuracy. Once we were satisfied with the model's work on a singular image, we gave the model the non-tilted concatenated dataset that we created. After the model achieved satisfactory results with the non-tilted dataset, we then gave the model the tilted dataset and continued in a similar manner until we were satisfied with the accuracy that the model achieved.

We then further tested our model by modifying datasets by randomly rotating each number within our panel by zero, ten or twenty degrees. We continually generated these panels, where the numbers were rotated, to test our classifier until we observed satisfactory results.

IV. Experiments

We first explored the results of our model when using the target number one. We varied the amount of images and rotation of our target number for different iterations. We then outputted the test accuracy based of off 500 test panels rotated by the same degree as the training data. We also recorded the test loss for each iteration.

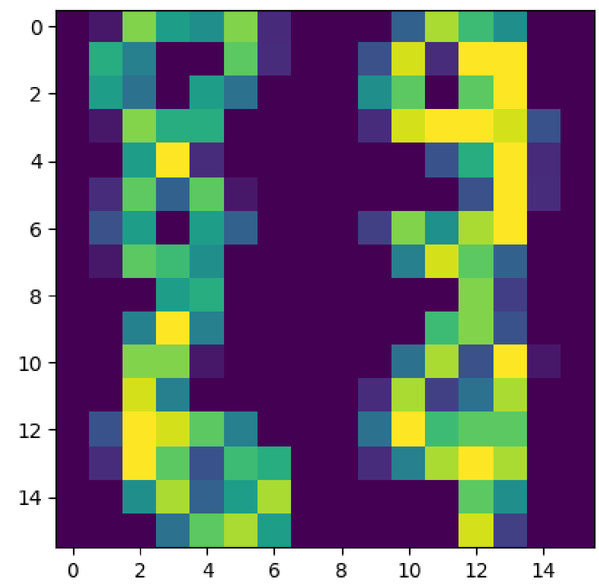

Fig. 3. Example panel showing a constructed rotated dataset

Target Num.	Num. Images	Rot Deg	Test Acc.	Test Loss
1	200	0	0.8100	0.4501
1	1000	0	0.8950	0.2262
1	2000	0	0.9750	0.0718
1	200	10	0.6500	0.6600
1	1000	10	0.8600	0.3215
1	2000	10	0.9500	0.0130
1	200	20	0.6980	0.6240
1	1000	20	0.8620	0.3202
1	2000	20	0.9500	0.1400

TABLE I

Results of attempting to identify the target number 1. The number of images used to train the model is shown as well as the rotation of the images. The last two columns show the test accuracy and the the test loss.

As shown in Table 1., when increasing the number of images our model trains on, we observe an overall increase in our test accuracy and a decrease in our test loss. When increasing the rotation of the target number we observed a slight decrease in our test accuracy and a slight increase in our test loss.

We then explored the results of our model when using different target numbers such as two, five, and nine. We still varied the amount of images and rotation of each specific target number for different iterations. Then we outputted the test accuracy based of off 500 test panels rotated by the same degree as the training data and recorded the test loss for each iteration.

As shown in Table 2., when increasing the number of images our model trains on, we still observe an overall increase in our test accuracy and decrease in our test loss. When increasing the rotation of each specific target number we also still observed a slight decrease in our test accuracy and slight increase in our test loss. When comparing the overall test accuracy for different target numbers, we noticed that when using nine as our target number we outputted the highest overall

Target Num.	Num. Images	Rot Deg	Test Acc.	Test Loss
2	200	0	0.8560	0.3355
2	1000	0	0.9559	0.1055
2	1000	10	0.9400	0.1380
2	2000	10	0.9800	0.0434
2	1000	20	0.9480	0.1370
5	200	0	0.6940	0.7341
5	1000	0	0.9200	0.2069
5	1000	10	0.9160	0.2150
5	2000	10	0.9500	0.1452
5	1000	20	0.9140	0.2631
9	200	0	0.9300	0.1800
9	1000	0	0.9500	0.1300
9	1000	10	0.9400	0.1800
9	2000	10	0.9300	0.1700
9	1000	20	0.9000	0.2900

TABLE II
RESULTS OF ATTEMPTING TO IDENTIFY DIFFERENT TARGET NUMBERS. THE NUMBER OF IMAGES USED TO TRAIN THE MODEL IS SHOWN AS WELL AS THE ROTATION OF THE IMAGES. THE LAST TWO COLUMNS SHOW THE TEST ACCURACY AND THE THE TEST LOSS.

test accuracy, and when using five as our target number we outputted the lowest overall test accuracy.

After observing satisfactory results from out model, where we are able to identify a randomly prompted target number, we now want to locate which quadrant of our panel the target number is in. In order to do so, we first constructed a panel where our target number was present. Then we also constructed a second panel where our target number was not present, called our cleanslate panel that our model did not misidentify. We used the cleanslate to block three quadrants at a time while we tested to see whether or not the unblocked quadrant had our target number with a very accurate predictor. Then we repeated this process for the remaining quadrants, recording the result of each quadrant.

To find the general accuracy of the model, we needed to perform this process over multiple iterations. While performing each iteration, we were recording the accuracy for each quadrant. Then we computed the percentage that we correctly located the target number; meaning all four quadrants were correctly predicted.

Below we have a table that shows the rate of the model guessing the correct location and how accurate the model was for each quadrant. These percentages were found over 500 iterations with 0 rotation.

Target Num.	Classifier Acc.	Correct Location	Quadrant Acc.
2	0.9820	0.5780	0.8810
4	0.9740	0.4440	0.8380
5	0.9660	0.3960	0.8010

TABLE III
RESULTS OF THE MODEL TO LOCATE SPECIFIC TARGET NUMBERS IN A PANEL.

V. DISCUSSION AND CONCLUSION

Overall, our model proved to be useful in locating and identifying a prompted target number within a given panel. Despite after rotating and randomizing the numbers within the panels, our model continued to produce high accuracy results.

We noticed that the accuracy increased over several factors: number of images, the rotation, and the target number itself. Notice in Table 1. that we had multiple accuracies less than 0.70 and this would indicate the model was not learning the dataset in the way we intended. Increasing the number of training panels improved the overall accuracy of the model even though it had longer run times.

As the rotation increased, we saw that more panels were required to properly learn the dataset. In Table 1, observe the lower accuracies with the rotation of 10 and 20 degrees when train on 200 panels in comparison to a rotation of 0 degrees (no rotation).

When comparing number to number, there was different accuaracies given by the model indicating that some numbers proved to be more difficult to learn compared to others. Comparing the target numbers 5 and 9 in Table 2, the number 5 had lower accuracies than the number 9 at the same benchmarks. This is most apparent when the number of images tested on is 200, the rotation is 0 degrees. The target number 5 may suffer in its accuracy due to the large variation in how people write the number.

When locating the target number in the panel, we found the accuaracies to not as good as we hoped that they would be. Observing Table 3, we can see that for any of the accuracies in the table are lower than what we would expect. This would indicate to us that the model may look the boundaries of the numbers that we disrupt in our method of locating. This would give us reason to attempt to use our model on a dataset that contains larger images.

Potentially in the future we can use this model on the MNIST dataset, which is has much larger images in the dataset, once we have refined the model. We would expect the same trends in accuracy would be similar if not the same in the MNIST dataset compared to the one we used. However, we would the accuracies when locating the target number in the panel would improve because the image is larger and the boundaries may not have as big of an impact on the model.

VI. REFERENCES

https://scikitlearn.org/stable/modules/generated/sklearn.datasets.load_digits.html

https://archive.ics.uci.edu/ml/datasets/optical+recognition+of+handwritten+digits

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

https://www.tensorflow.org/api_docs/python/tf/keras/layers

https://keras.io/guides/sequential_model/