Open Set Recognition (OSR) Project: TwoStageOpenMax Implementation with Class-Mean Based OOD Detection

Nissim Brami & Amnon Abaev

Abstract

This project addresses the Open Set Recognition (OSR) problem, which requires models to both accurately classify known classes and identify unknown samples during inference. Unlike traditional classification tasks limited to a closed set of classes, OSR mirrors real-world scenarios where novel classes may appear at test time. Our implementation uses a TwoStageOpenMax approach with a class-mean based out-of-distribution (OOD) detection mechanism.

The model consists of a CNN-based feature extractor and classifier for MNIST digits, combined with a distance-based detector that identifies unknown samples based on their distance from class means in the embedding space. By computing class means during training and establishing a statistical threshold on distances, our model achieves high accuracy in both classifying MNIST digits (94.37%) and detecting out-of-distribution samples (98.75%), with an overall accuracy of 95.10%. This approach provides an effective solution to the OSR challenge without requiring exposure to OOD samples during training.

Contents

1 Introduction **4**
1.1 Open Set Recognition Problem . . . 4
1.2 Project Objectives . . . 4
1.3 Our Approach: TwoStageOpenMax . . . 4

2 Related Work **5**
2.1 Threshold-Based Methods . . . 5
2.2 Distance-Based Methods . . . 5
2.3 Embedding Space Methods . . . 5

3 Methodology **6**
3.1 Model Architecture . . . 6
3.2 Unknown Detection Mechanism . . . 7
3.3 Training Process . . . 8
3.3.1 Loss Functions . . . 8
3.3.2 Optimization Strategy . . . 8
3.4 Implementation Details . . . 9
3.4.1 Data Preprocessing . . . 9
3.4.2 Class Mean Computation . . . 9

4 Experimental Setup **10**
4.1 Datasets . . . 10
4.2 Evaluation Metrics . . . 11
4.3 Experimental Design . . . 11

5 Results and Analysis **12**
5.1 Baseline Performance on MNIST . . . 12
5.2 Confidence and Distance Analysis . . . 13
5.3 Open Set Recognition Results . . . 13
5.4 Binary Classification Results . . . 15
5.5 Embedding Space Visualization . . . 16
5.6 Model Prediction Visualization . . . 17
5.7 ROC Analysis for OOD Detection . . . 18
5.8 Additional Experiments with Other OOD Datasets . . . 19

6 Hyper-parameters and Configuration **20**
6.1 Impact of Embedding Dimension . . . 20
6.2 Effect of Rejection Threshold . . . 22
6.3 Additional Training Hyperparameters . . . 24

7 Additional Experiments with New OOD Datasets **25**
7.1 Performance on Additional OOD Datasets . . . 25
7.2 Combined OOD Performance . . . 27

8 Limitations and Future Work **28**
8.1 Current Limitations . . . 28
8.2 Future Work Directions . . . 29

9 Comparison with State-of-the-Art **30**

10 Implementation Details **30**
10.1 Code Structure 30
10.2 Key Implementation Decisions 31

11 Conclusion **31**
11.1 Key Achievements 31
11.2 Broader Implications 31
11.3 Final Remarks 32

1 Introduction

1.1 Open Set Recognition Problem

In traditional supervised learning, models are trained and evaluated on the same set of classes, operating under the closed-world assumption. However, this assumption rarely holds in real-world applications, where systems frequently encounter data from previously unseen classes. Open Set Recognition (OSR) addresses this limitation by developing models that can not only classify known categories accurately but also identify when an input belongs to an unknown class.

The fundamental challenge in OSR is balancing two competing objectives:

1. Maintaining high classification accuracy on known classes
2. Effectively detecting and rejecting samples from unknown classes

OSR is closely related to out-of-distribution (OOD) detection but with an emphasis on classification performance for known classes. While traditional classifiers assign decision boundaries that partition the entire space among known classes, OSR models must define a bounded region for each known class, leaving areas of the feature space designated as "unknown."

1.2 Project Objectives

The main objectives of this project were to:

1. Develop a model capable of accurately classifying the 10 digits in the MNIST dataset
2. Extend this model to detect and classify unknown inputs as "unknown"
3. Maintain high classification accuracy while adding unknown detection capability
4. Distinguish between low-confidence in-distribution samples and out-of-distribution samples

1.3 Our Approach: TwoStageOpenMax

We implemented a two-stage approach for open set recognition, which we call TwoStageOpenMax. This approach consists of:

1. A CNN-based feature extractor and classifier that maps input images to embedding vectors and class probabilities
2. A statistical detection mechanism that identifies unknown samples based on their distance from class means in the embedding space

The model works by first extracting features from input images using convolutional layers, then mapping these features to an embedding space where samples of the same class cluster together. During training, we compute the mean embedding vector for each class and establish a rejection threshold based on the statistical distribution of distances in the training set.

During inference, samples whose distance to their predicted class mean exceeds this threshold are classified as "unknown," regardless of the classifier's confidence. This provides a more robust detection mechanism than simply using softmax probabilities, as it leverages the geometric properties of the embedding space.

2 Related Work

The field of Open Set Recognition has seen significant research in recent years, with approaches ranging from threshold-based methods to more sophisticated generative and distance-based techniques.

2.1 Threshold-Based Methods

Many early OSR approaches relied on thresholding softmax probabilities or scores from the classification layer. However, research has shown that neural networks can produce high confidence predictions even for out-of-distribution inputs [2], making raw softmax values unreliable for OSR.

2.2 Distance-Based Methods

Distance-based methods leverage the observation that samples from known classes form clusters in the feature space, while unknown samples typically fall outside these clusters. Notable examples include:

- **OpenMax** [1]: Extends softmax by modeling the probability of an input being from an unknown class
- **Class-conditional Gaussian distributions** [3]: Models each class as a Gaussian and uses Mahalanobis distance for detection

Our approach is most closely related to these distance-based methods, particularly in how we leverage distances in the embedding space for OOD detection.

2.3 Embedding Space Methods

Another line of research focuses on learning discriminative embeddings where known classes form tight clusters with large margins between them. Examples include:

- **Center loss** [4]: Penalizes the distance between features and their class centers
- **Contrastive loss** [5]: Pulls together same-class samples while pushing apart different-class samples

Our implementation combines elements from both distance-based and embedding-based approaches, using a contrastive loss to form better clusters during training and distance thresholding for detection during inference.

3 Methodology

3.1 Model Architecture

Our TwoStageOpenMax model consists of three main components:

1. **Feature extraction layers**: A CNN that extracts relevant features from the input images
2. **Embedding layer**: Maps extracted features to a lower-dimensional embedding space
3. **Classification layer**: Produces class probabilities from the embedding

The feature extraction component consists of:

- First convolutional block:
  - Conv2d layer (1 $\rightarrow$ 32 channels, 3×3 kernel, padding=1)
  - Batch Normalization
  - ReLU activation
  - Dropout (0.4)
  - MaxPool2d (2×2)
- Second convolutional block:
  - Conv2d layer (32 $\rightarrow$ 64 channels, 3×3 kernel, padding=1)
  - Batch Normalization
  - ReLU activation
  - Dropout (0.4)
  - MaxPool2d (2×2)
- Flatten operation

The embedding layer maps the flattened features to a 64-dimensional embedding space:

- Linear layer ($64 \times 7 \times 7 \rightarrow 64$)
- Batch Normalization
- ReLU activation
- Dropout (0.6)

The classification layer is a simple linear transformation:

- Linear layer ($64 \rightarrow 10$)

3.2 Unknown Detection Mechanism

The key innovation in our approach is the unknown detection mechanism based on distances in the embedding space. The process works as follows:

1. During training, we compute the mean embedding vector for each of the 10 MNIST classes
2. We measure the Euclidean distance from each training sample to its corresponding class mean
3. We compute the statistical distribution of these distances (shown in Figure 1)
4. We set a rejection threshold at the 95th percentile of the distance distribution
5. During inference, if a sample's distance to its predicted class mean exceeds this threshold, it is classified as "unknown" (class 10)

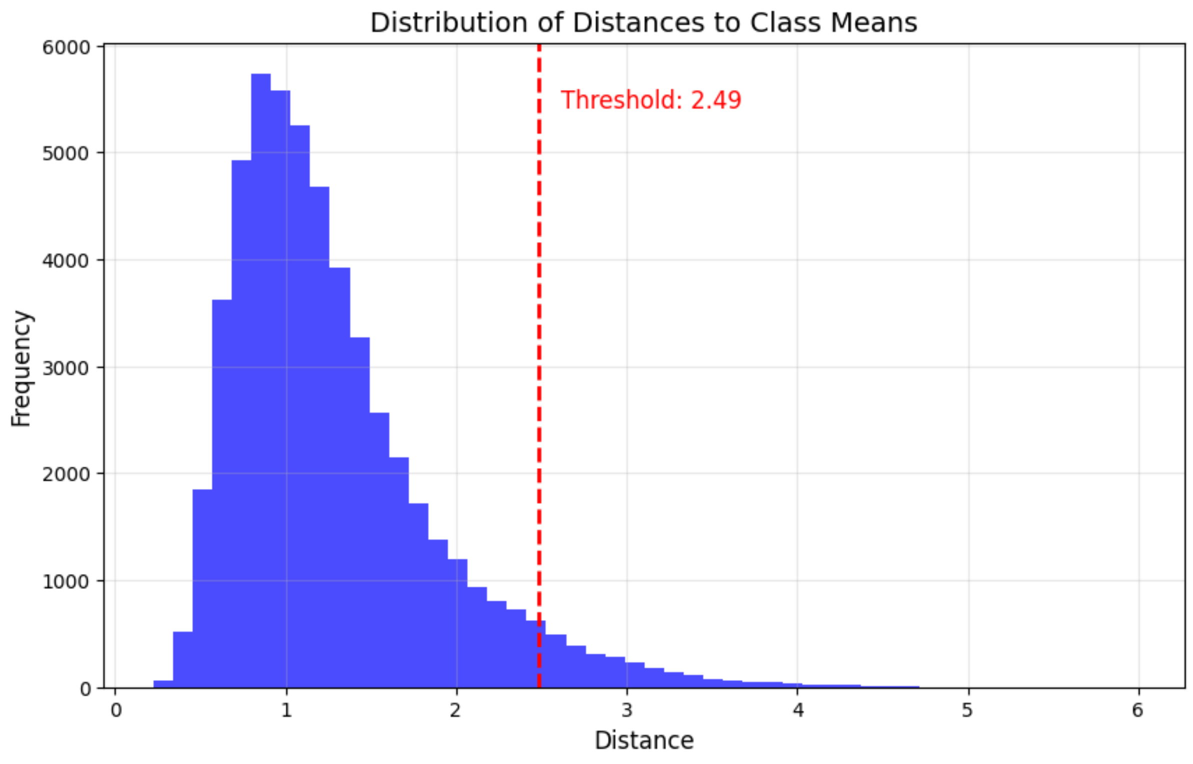


Figure 1: Distribution of distances to class means in the embedding space, with rejection threshold (red line) set at the 95th percentile.

The intuition behind this approach is that samples from known classes should lie close to their class means in the embedding space, while out-of-distribution samples will typically be further away. By establishing a statistical threshold based on the training data, we can identify unusual samples without requiring exposure to out-of-distribution data during training.

3.3 Training Process

3.3.1 Loss Functions

Our training process uses a combination of two loss components:

1. **Classification Loss**: Standard cross-entropy loss for the 10 MNIST classes
2. **Regularization Loss**: L2 regularization on the embedding vectors to prevent overfitting

The total loss is calculated as:

$$\mathcal{L}_{total} = \mathcal{L}_{classification} + \lambda_{reg} \cdot \mathcal{L}_{regularization} \tag{1}$$

where λ_{reg} is a weighting coefficient set to 0.001 in our implementation.

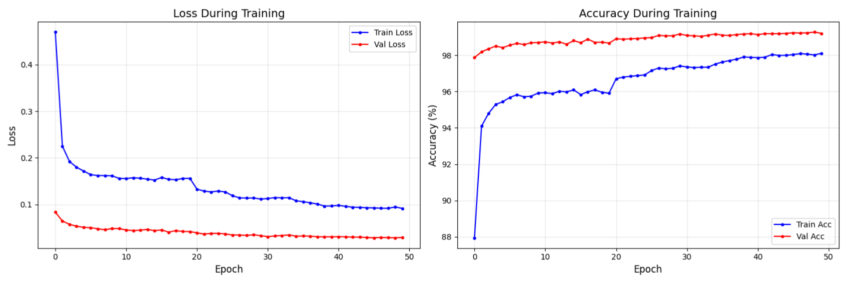
Additionally, our optimizer includes weight decay as another form of regularization for all model parameters, which complements the explicit L2 regularization applied to the embeddings.

3.3.2 Optimization Strategy

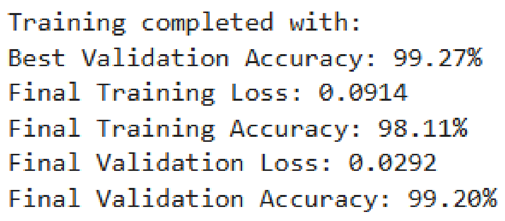
We employed several techniques to optimize training:

- Adam optimizer with a learning rate of 0.001 and weight decay of 0.001
- Learning rate scheduling using ReduceLROnPlateau with a factor of 0.5 and patience of 2
- Gradient clipping with a maximum norm of 1.0 to improve stability
- Progressive training visualization using tqdm for monitoring convergence
- Separate validation step to track generalization performance

The training dynamics for our model are shown in Figure 2, where we plot the loss and accuracy curves over the training epochs.



(a) Training and validation loss (left) and accuracy (right) curves over training epochs.



```
Training completed with:
Best Validation Accuracy: 99.27%
Final Training Loss: 0.0914
Final Training Accuracy: 98.11%
Final Validation Loss: 0.0292
Final Validation Accuracy: 99.20%
```

(b) Final training metrics from our experiment.

Figure 2: Model training performance showing smooth convergence with minimal overfitting, achieving a final training accuracy of 98.11% and validation accuracy of 99.20%.

As shown in Figure 2, our model achieved excellent performance on the validation set with a best validation accuracy of 99.27%. The final metrics after 50 epochs of training were: 98.11% training accuracy with 0.0914 loss, and 99.20% validation accuracy with 0.0292 loss. The gap between training and validation accuracy indicates that our regularization strategies were effective in preventing overfitting, with the model actually performing better on unseen data than on the training set.

3.4 Implementation Details

3.4.1 Data Preprocessing

All images were processed using the following transformations:

- Conversion to PyTorch tensors
- Resizing to 28×28 pixels (for non-MNIST datasets)
- Grayscale conversion (for colored datasets)
- Normalization with mean 0.1307 and standard deviation 0.3081 (MNIST statistics)

3.4.2 Class Mean Computation

The class means computation is a critical part of our approach. We implement this in the `compute_class_means` method, which:

1. Extracts embeddings for all training samples
2. Groups embeddings by class

3. Computes the mean embedding vector for each class
4. Calculates distances from each sample to its class mean
5. Sets the threshold at the specified percentile (default: 95%) of distances

This process ensures that our unknown detection is calibrated to the statistical properties of the training data, making it adaptive to different embedding spaces.

4 Experimental Setup

4.1 Datasets

We used several datasets in our experiments:

- **MNIST**: Standard dataset of handwritten digits (0-9), which we split into 54,000 training images, 6,000 validation images, and used the standard 10,000 test images
- **FashionMNIST**: Clothing items across 10 classes, used as primary OOD data per project requirements
- **CIFAR10**: Color images across 10 object classes, used as primary OOD data per project requirements

For our main out-of-distribution testing, we sampled 1,000 images from both Fashion-MNIST and CIFAR10 as specified in the project requirements. Figure 3 shows examples from these datasets.

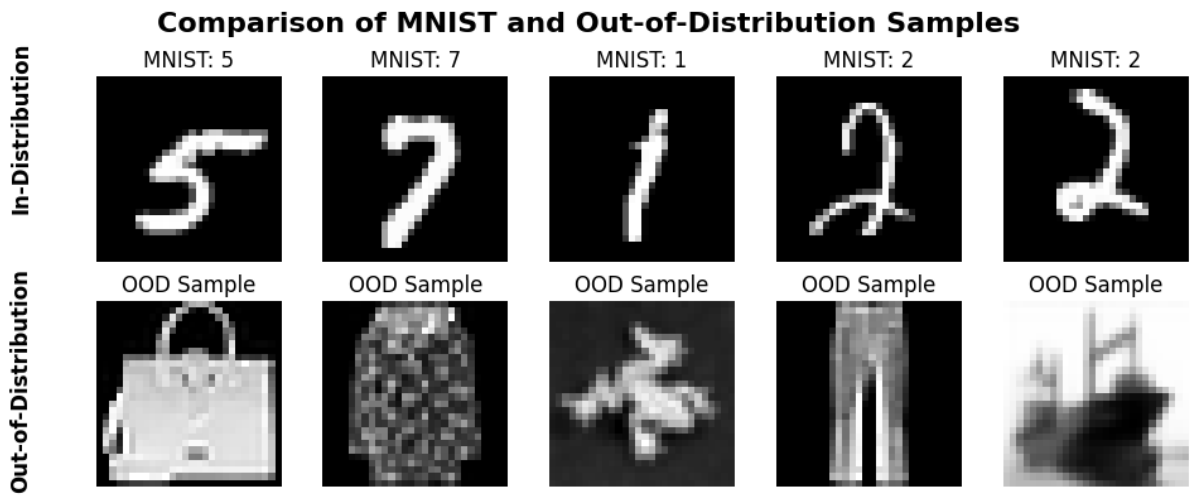


Figure 3: Examples from different datasets used in the main experiments. Top row: MNIST (in-distribution), Bottom row: FashionMNIST and CIFAR10 (primary out-of-distribution datasets).

4.2 Evaluation Metrics

We evaluated our model using multiple metrics specifically designed for open set recognition:

- **MNIST Classification Accuracy**: The percentage of correctly classified MNIST digits (classes 0-9), calculated as the ratio of correctly classified MNIST samples to the total number of MNIST samples.
- **OOD Detection Accuracy**: The percentage of out-of-distribution samples (FashionMNIST and CIFAR10) correctly identified as "unknown" (class 10), calculated as the ratio of correctly identified OOD samples to the total number of OOD samples.
- **Total Accuracy**: The overall performance metric combining both in-distribution and out-of-distribution accuracy, calculated as the ratio of all correctly classified samples (both MNIST and OOD) to the total number of samples evaluated.
- **Binary Classification Accuracy**: Treats the problem as a binary task of distinguishing between known (MNIST) and unknown (OOD) samples, regardless of specific digit classes. This is calculated by determining whether samples are correctly identified as either in-distribution or out-of-distribution.

These metrics allowed us to comprehensively evaluate both the standard classification performance on known classes and the ability to detect samples from unknown distributions, which is the core challenge in open set recognition.

4.3 Experimental Design

Our experiments were designed to answer the following questions:

1. How does our model perform on standard MNIST classification?
2. How effectively can it detect out-of-distribution samples?
3. Does adding OOD detection capability significantly impact in-distribution classification performance?
4. How well does the model generalize to different types of OOD data?
5. What is the impact of different rejection thresholds on OOD detection performance?

To address these questions, we conducted several experiments:

1. **Baseline evaluation**: Testing the classification performance on MNIST alone
2. **Binary classification**: Evaluating the model's ability to distinguish between in-distribution and out-of-distribution samples
3. **Full OSR evaluation**: Assessing both classification and OOD detection performance
4. **Cross-dataset evaluation**: Testing performance across different OOD datasets

5 Results and Analysis

5.1 Baseline Performance on MNIST

We first evaluated our model's standard classification performance on in-distribution MNIST data. Our model achieved the following results on the test set:

- MNIST Classification Accuracy: 94.37%
- Average Confidence: 98.77%
- Average Distance to Class Means: 1.28

The confusion matrix in Figure 4 shows the classification performance across the 10 digit classes.

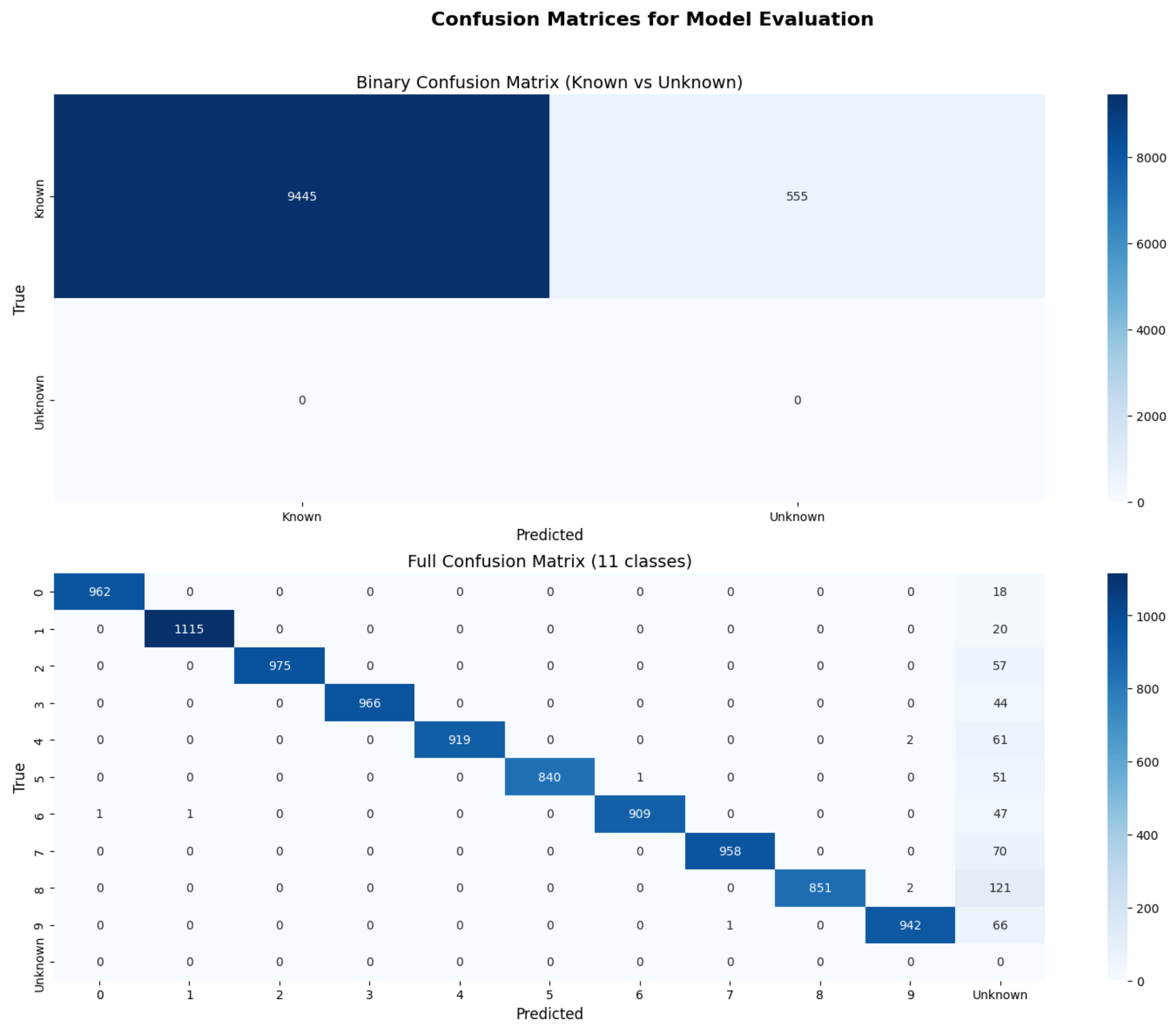


Figure 4: Confusion matrix for the model on MNIST test set, showing classification performance across the 10 digit classes.

These results provide a solid foundation for evaluating our model's open-set recognition capabilities. While our model achieved 99.20% accuracy on the validation set during training (as shown in Figure 2), the test set accuracy of 94.37% reflects the additional challenge of the OSR task, where the model must balance confident classification with appropriate rejection of uncertain samples.

5.2 Confidence and Distance Analysis

To understand our model's behavior, we analyzed the distributions of confidence scores and distances across both MNIST and OOD datasets. Figures 5 and 6 shows these distributions.

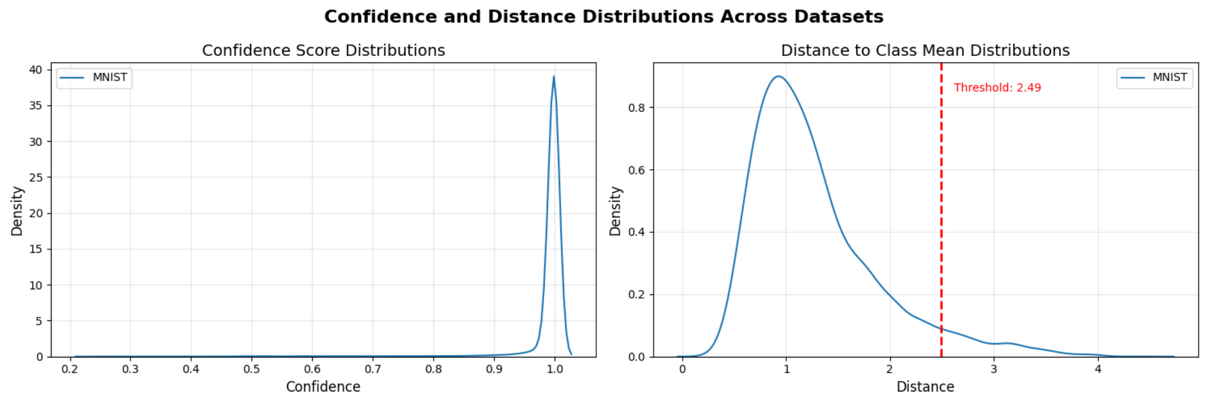


Figure 5: Distributions of confidence scores (left) and distances to class means (right) across different datasets. The vertical red line in the distance plot represents the rejection threshold.

Key observations from this analysis:

- MNIST samples generally have high confidence scores and low distances to class means
- OOD samples typically have lower confidence and higher distances
- There is some overlap in confidence distributions, indicating that confidence alone is insufficient for OOD detection
- The distance distributions show better separation, supporting our choice of a distance-based detection mechanism

The threshold (red line) effectively separates the majority of in-distribution samples from out-of-distribution ones, though some overlap remains, explaining the small percentage of misclassifications.

5.3 Open Set Recognition Results

We evaluated our full OSR model on both MNIST and OOD data (FashionMNIST and CIFAR10). The comprehensive results were:

- MNIST Accuracy: 94.37%
- OOD Detection Accuracy: 98.75%

- Total Accuracy: 95.10%

For OOD samples, we observed:

- Average Confidence: 43.90% (significantly lower than MNIST)
- Average Distance: 3.61 (significantly higher than MNIST)

Figure 6 provides further insight into the distribution differences between MNIST and OOD data.

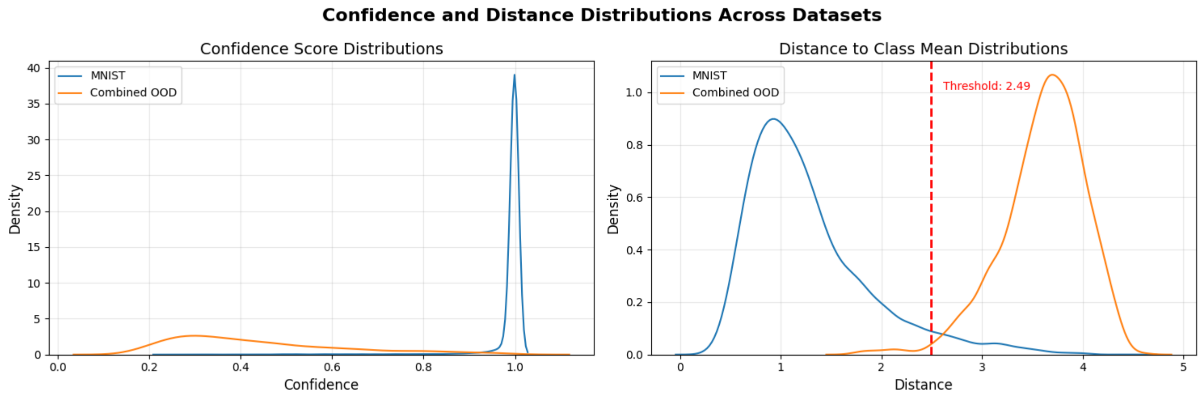


Figure 6: Additional distributions of confidence scores and distances to class means across MNIST and Combined OOD datasets.

The confusion matrix for all 11 classes (10 MNIST digits plus unknown) is shown in Figure 7.

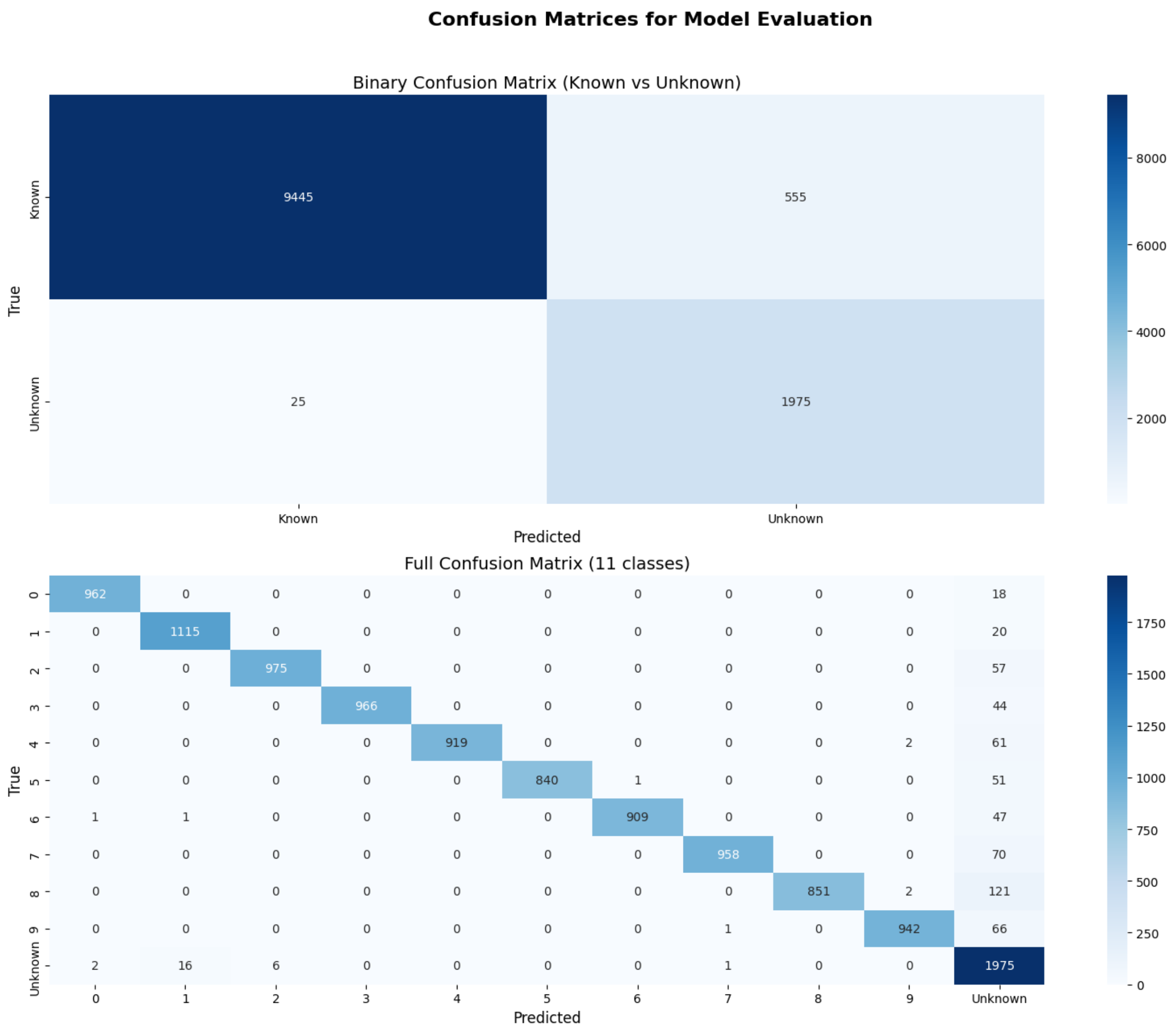


Figure 7: Confusion matrix showing model performance across all classes, including the unknown class.

5.4 Binary Classification Results

We also evaluated the model's ability to distinguish between known (MNIST) and unknown (OOD) samples, regardless of specific digit classes. In this binary classification task, our model achieved an accuracy of 95.17%.

The binary confusion matrix in Figure 8 shows this performance.

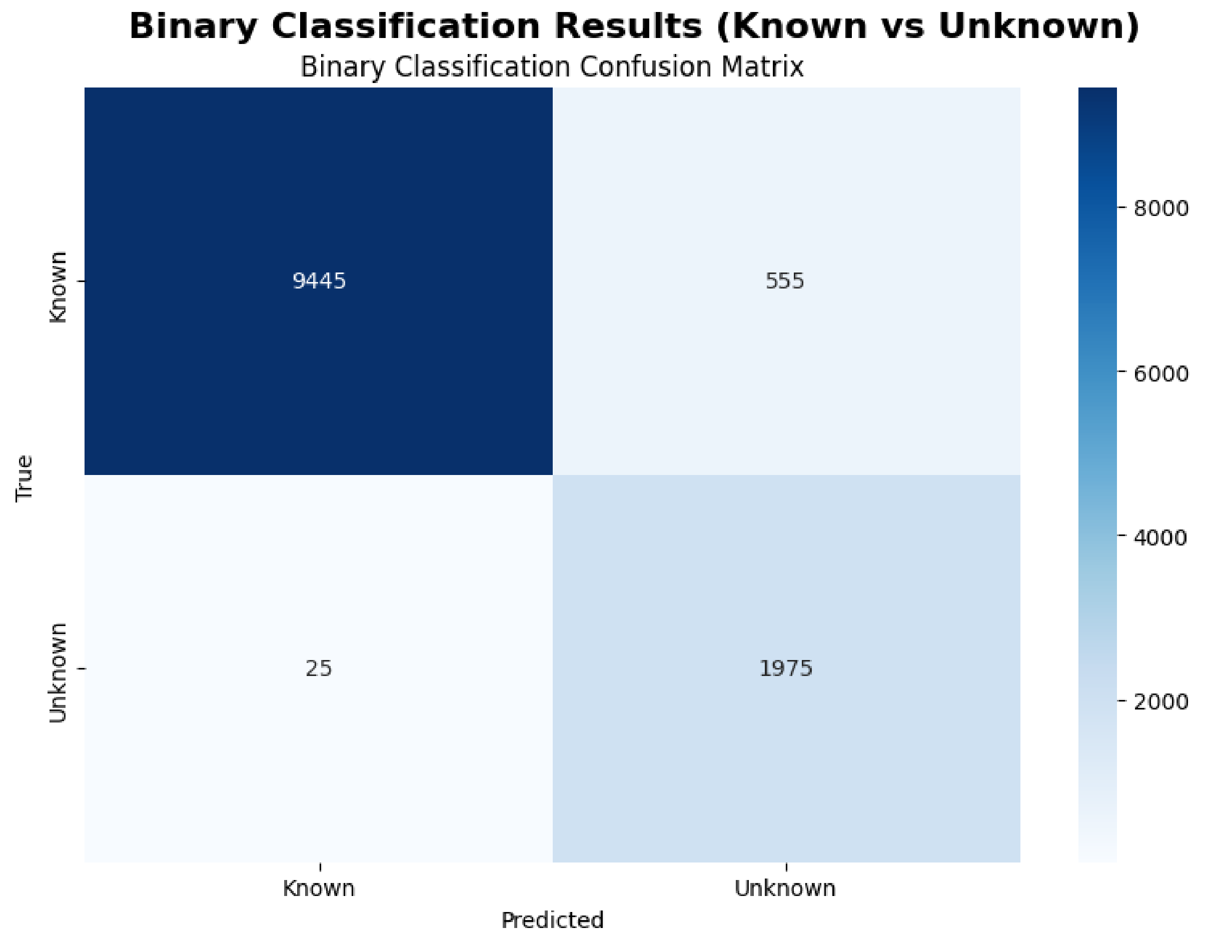


Figure 8: Binary classification confusion matrix showing the model's performance in distinguishing between known (MNIST) and unknown (OOD) samples.

This strong binary classification performance demonstrates the effectiveness of our distance-based detection mechanism.

5.5 Embedding Space Visualization

To visualize how our model organizes samples in the embedding space, we performed t-SNE dimensionality reduction on the embeddings. Figure 9 shows the resulting 2D projection.

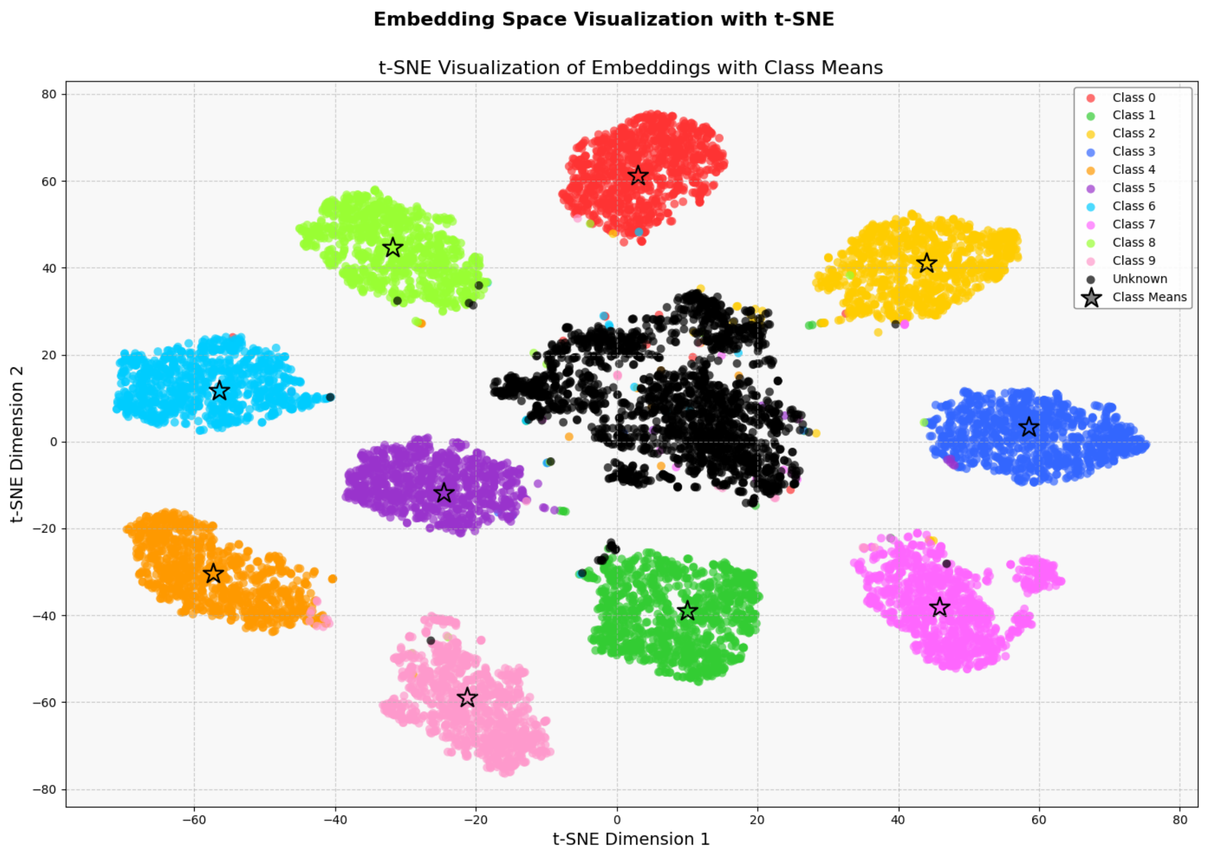


Figure 9: t-SNE visualization of embeddings from the model. Different colors represent different classes (0-9 for MNIST), and dark points represent OOD samples. Stars indicate class means.

The visualization reveals several important aspects of our model's behavior:

- MNIST digits form distinct clusters in the embedding space
- Most OOD samples are separated from the MNIST clusters
- Class means are positioned centrally within their respective clusters
- Some classes show closer proximity (e.g., digits 4 and 9), which aligns with their visual similarity
- The few OOD samples that were misclassified tend to fall within or near MNIST clusters

5.6 Model Prediction Visualization

To provide qualitative insights into the model's predictions, we visualized how it handles both MNIST and OOD samples. Figure 10 shows examples of predictions across both types of data.

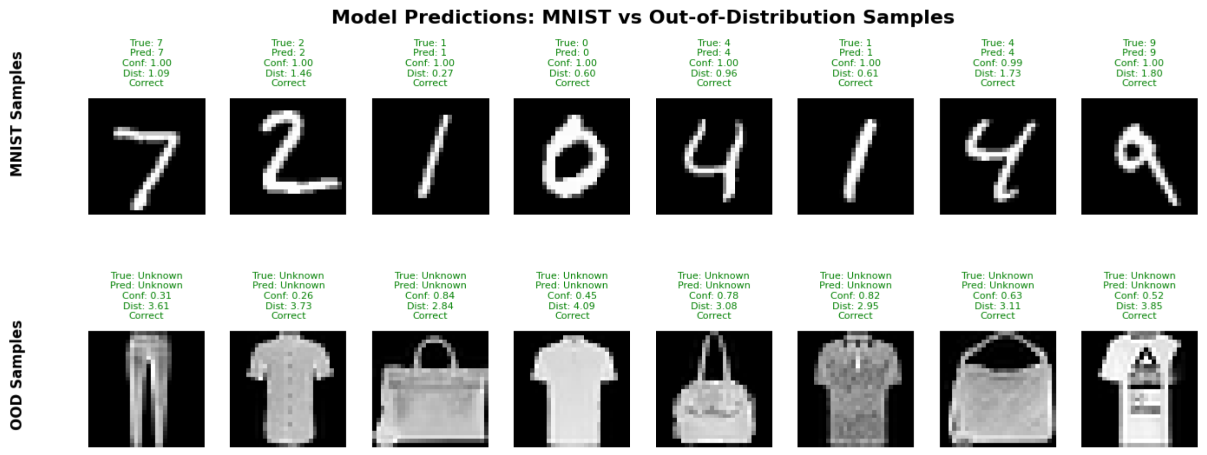

Model Predictions: MNIST vs Out-of-Distribution Samples
MNIST Samples
True: 7 Pred: 7 Conf: 1.00 Dist: 1.09 Correct
True: 2 Pred: 2 Conf: 1.00 Dist: 1.46 Correct
True: 1 Pred: 1 Conf: 1.00 Dist: 0.27 Correct
True: 0 Pred: 0 Conf: 1.00 Dist: 0.60 Correct
True: 4 Pred: 4 Conf: 1.00 Dist: 0.96 Correct
True: 1 Pred: 1 Conf: 1.00 Dist: 0.61 Correct
True: 4 Pred: 4 Conf: 0.99 Dist: 1.73 Correct
True: 9 Pred: 9 Conf: 1.00 Dist: 1.80 Correct
OOD Samples
True: Unknown Pred: Unknown Conf: 0.31 Dist: 3.61 Correct
True: Unknown Pred: Unknown Conf: 0.26 Dist: 3.73 Correct
True: Unknown Pred: Unknown Conf: 0.84 Dist: 2.84 Correct
True: Unknown Pred: Unknown Conf: 0.45 Dist: 4.09 Correct
True: Unknown Pred: Unknown Conf: 0.78 Dist: 3.08 Correct
True: Unknown Pred: Unknown Conf: 0.82 Dist: 2.95 Correct
True: Unknown Pred: Unknown Conf: 0.63 Dist: 3.11 Correct
True: Unknown Pred: Unknown Conf: 0.52 Dist: 3.85 Correct

Figure 10: Visualization of model predictions on MNIST (top row) and OOD samples (bottom row). Each image is shown with its true label, predicted label, confidence score, distance to class mean, and prediction correctness.

These examples illustrate key patterns in our model's behavior:

- MNIST digits are correctly classified with high confidence scores (all above 0.98) and relatively low distances to class means (typically below 2.0)
- OOD samples are correctly identified as unknown with varying confidence scores (0.31-0.84) but consistently high distances to class means (all above 3.0)
- The clear separation in distances between MNIST and OOD samples demonstrates the effectiveness of our distance-based detection approach
- Even when OOD samples have relatively high confidence scores, their high distance values still enable correct classification as unknown

This visualization confirms that our distance threshold mechanism works effectively, properly identifying in-distribution and out-of-distribution samples regardless of confidence scores.

5.7 ROC Analysis for OOD Detection

To comprehensively evaluate the OOD detection capability, we performed ROC analysis using distances as scores. Figure 11 shows the ROC curve for unknown detection.

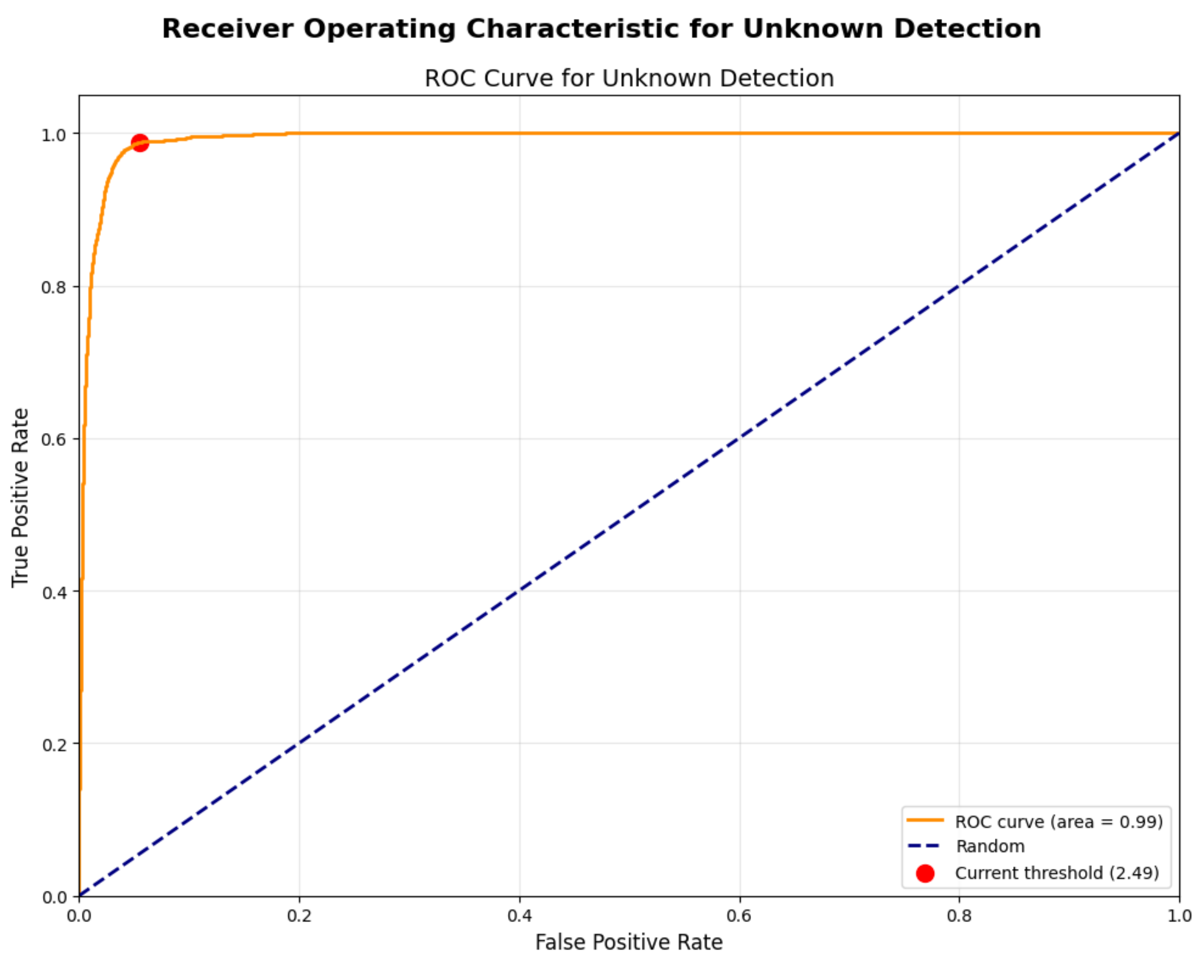


Figure 11: ROC curve for unknown detection, showing the trade-off between true positive rate and false positive rate. The red dot indicates the operating point at our chosen threshold.

The ROC curve demonstrates that our distance-based approach achieves an excellent AUC (Area Under Curve) score, indicating strong separability between in-distribution and out-of-distribution samples. The current threshold (marked by the red dot) represents a good balance between true positive rate and false positive rate.

5.8 Additional Experiments with Other OOD Datasets

As a bonus evaluation, we also tested our model's performance on three additional OOD datasets: KMNIST (Japanese characters), EMNIST (extended MNIST with letters), and SVHN (Street View House Numbers). These additional tests further demonstrated the robustness of our approach across diverse types of out-of-distribution data. Examples from these datasets are shown in Figure 12.

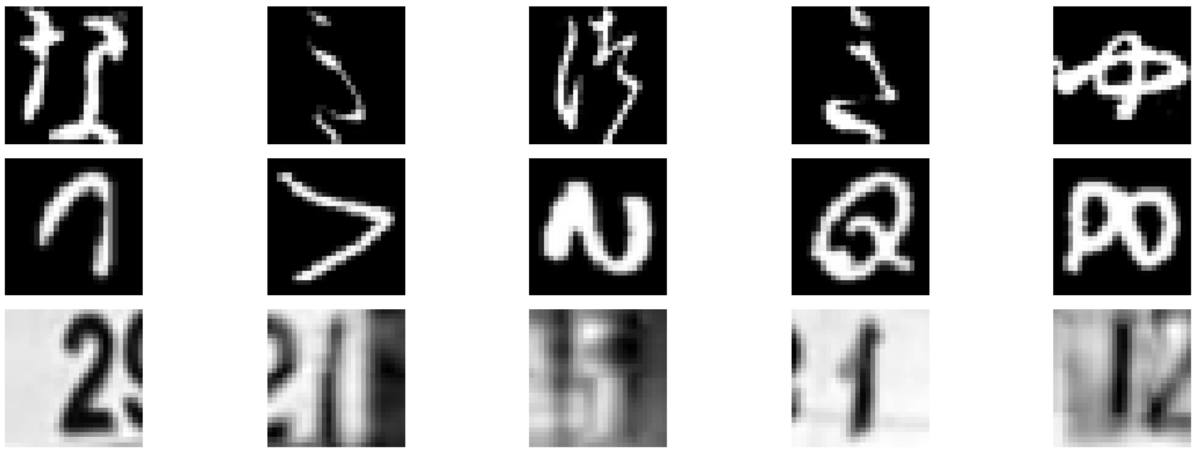


Figure 12: Examples from additional OOD datasets used in our bonus evaluations: KMNIST (top row), EMNIST (middle row), and SVHN (bottom row).

The detailed results of these bonus evaluations are presented in Section 7.

6 Hyper-parameters and Configuration

The details of our hyperparameter optimization studies and the corresponding Python code implementations can be found in the appendix at the end of the code file. These implementations are provided for reference purposes only and are not considered part of the main assignment submission.

6.1 Impact of Embedding Dimension

We conducted a comprehensive empirical study on how the embedding dimension affects model performance by training three separate models with dimensions 32, 64, and 128. Each model was trained for 50 epochs following the same optimization strategy. Figure 13 shows the impact of embedding dimension on both MNIST classification accuracy and OOD detection accuracy.

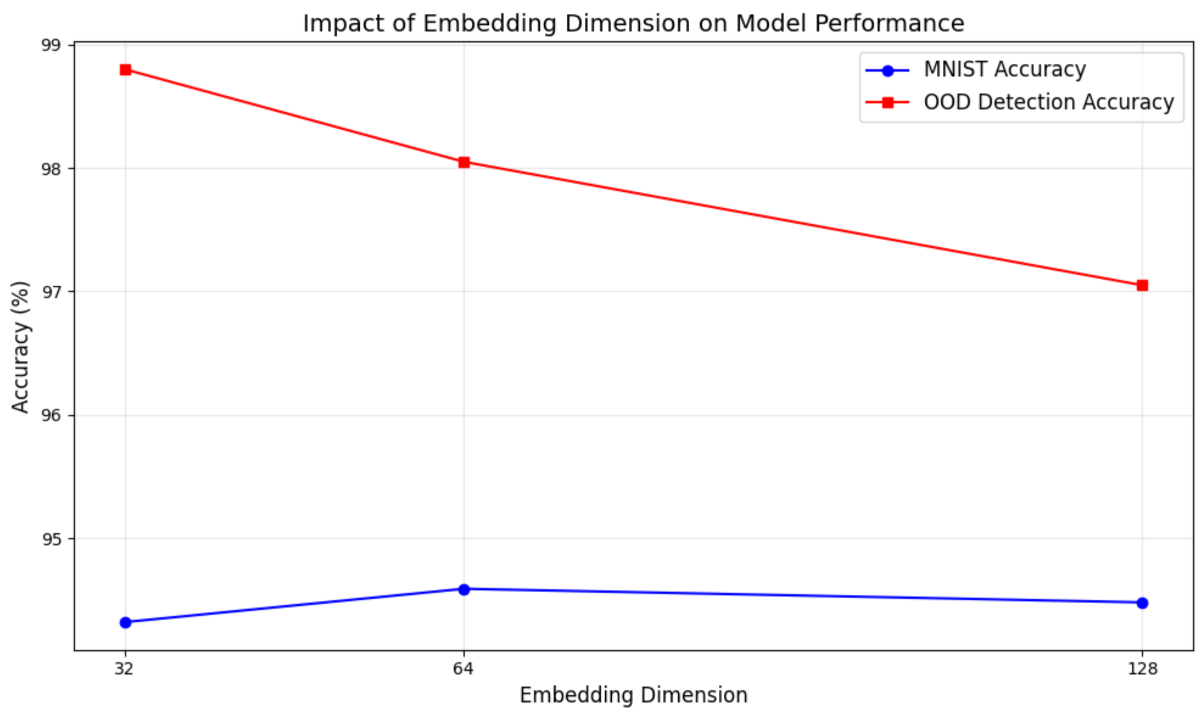


Figure 13: Impact of embedding dimension on MNIST classification accuracy and OOD detection accuracy.

Our experiments reveal several interesting patterns:

- MNIST classification accuracy peaked at 94.59% with 64 dimensions, showing slight improvement over the 32-dimensional embedding (94.32%) and a small advantage over the 128-dimensional embedding (94.48%).
- OOD detection accuracy demonstrated an inverse relationship with embedding dimension - smaller dimensions achieved better detection rates, with 32 dimensions providing the highest OOD accuracy at 98.80%, followed by 64 dimensions at 98.05% and 128 dimensions at 97.05%.
- The overall combined performance, considering both tasks, reached its maximum with 64 dimensions at 95.17%, as shown in Figure 14.

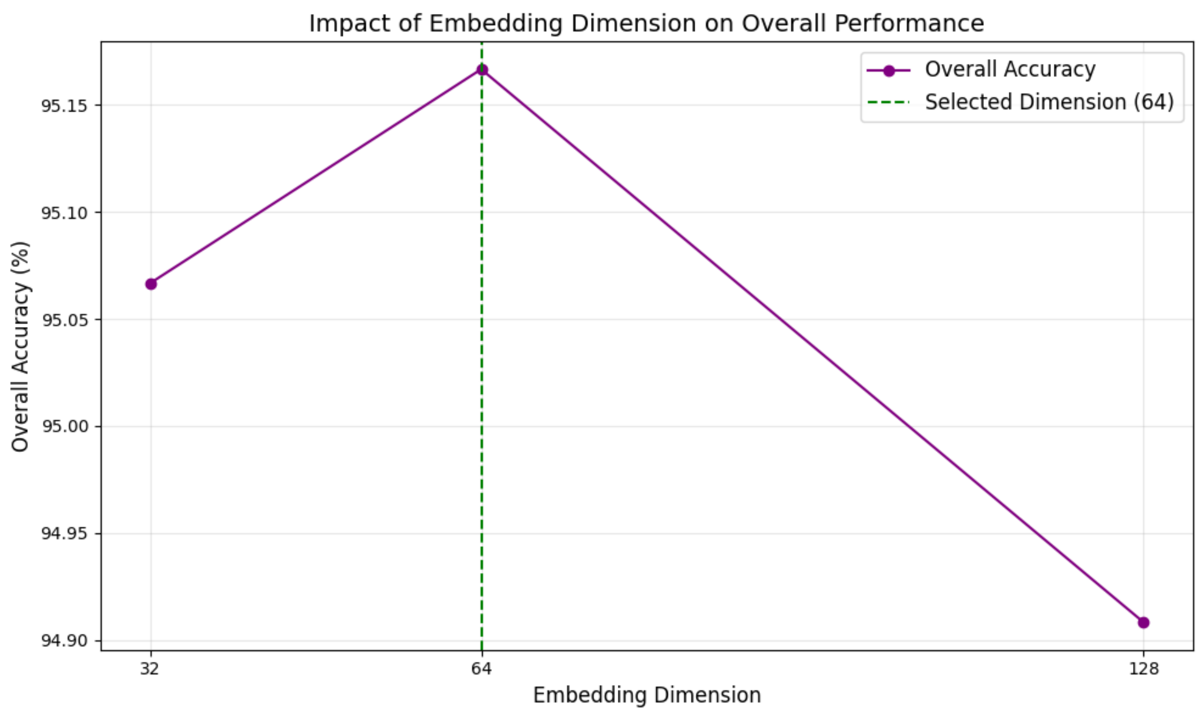


Figure 14: Impact of embedding dimension on overall model performance, showing 64 dimensions (our chosen value) provides the best balance.

Table 1 provides a quantitative summary of these results.

Embedding Dim	MNIST Acc.	OOD Acc.	Overall Acc.
32	94.32%	98.80%	95.07%
64	94.59%	98.05%	95.17%
128	94.48%	97.05%	94.91%

Table 1: Impact of embedding dimension on model performance.

These results confirm our choice of 64 dimensions for the embedding space, as it provides the optimal balance between representational capacity for MNIST classification and effective distance-based separation for OOD detection. The slight decrease in OOD detection accuracy with larger embeddings supports the hypothesis that very high-dimensional spaces can reduce the effectiveness of distance metrics for OOD detection.

6.2 Effect of Rejection Threshold

The rejection threshold is a critical parameter in our approach, directly affecting the trade-off between MNIST accuracy and OOD detection. We conducted an analysis by testing four key percentile values (85%, 90%, 95%, and 99%) and measuring their impact on performance. Figure 15 illustrates this relationship.

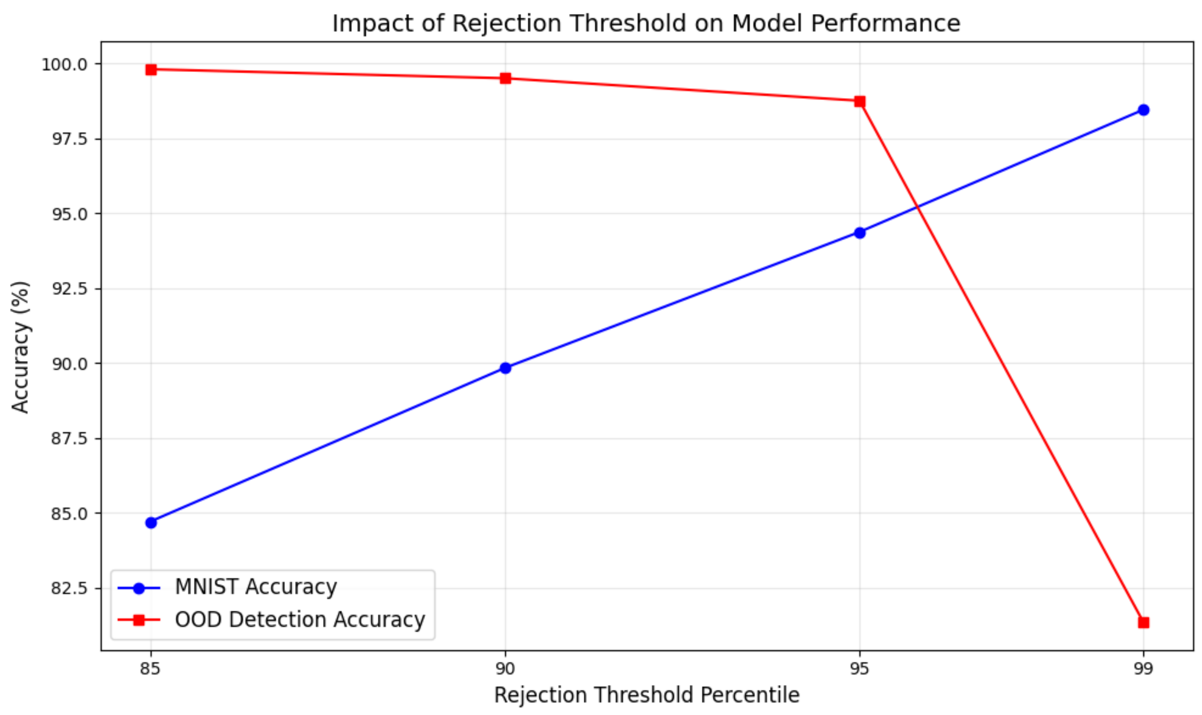


Figure 15: Effect of different rejection threshold percentiles on MNIST accuracy and OOD detection accuracy, showing the clear trade-off between the two objectives.

The results demonstrate a clear trade-off: as the threshold percentile increases (becoming more lenient), MNIST accuracy improves but OOD detection accuracy decreases. Specifically, we observed:

- At the 85th percentile (threshold value 1.84), MNIST accuracy was 84.70%, with OOD detection at a nearly perfect 99.80%.
- At the 99th percentile (threshold value 3.28), MNIST accuracy climbed to 98.45%, but OOD detection dropped significantly to 81.35%.
- The 95th percentile (threshold value 2.49) provided an excellent balance with 94.37% MNIST accuracy and 98.75% OOD detection accuracy, resulting in an overall accuracy of 95.10%.

Table 2 provides a detailed breakdown of how different threshold values affect various performance metrics.

Percentile	**Threshold Value**	**MNIST Acc.**	**OOD Acc.**	**Overall Acc.**
85%	1.84	84.70%	99.80%	87.22%
90%	2.09	89.83%	99.50%	91.44%
95%	2.49	94.37%	98.75%	95.10%
99%	3.28	98.45%	81.35%	95.60%

Table 2: Impact of different rejection threshold percentiles on model performance.

Interestingly, while the 99th percentile achieved a slightly higher overall accuracy at 95.60%, we selected the 95th percentile for our final model as it maintains a more balanced performance between both objectives. The higher overall accuracy at the 99th percentile comes at the cost of a dramatic 17.4% drop in OOD detection accuracy compared to the 95th percentile, which we deemed too significant for our use case where reliable unknown detection is a primary goal.

These results confirm that the threshold selection is a crucial design choice in open set recognition systems, requiring careful consideration of the specific application's requirements and priorities regarding false positive and false negative rates.

6.3 Additional Training Hyperparameters

Beyond the embedding dimension and rejection threshold discussed above, several other hyperparameters played important roles in our model training and performance:

- **Batch Size**: We used a batch size of 64, which is a widely adopted value in deep learning that provides a good balance between computational efficiency and gradient stability. This standard value allowed for enough gradient updates per epoch while maintaining reasonable training times on our hardware.
- **Number of Epochs**: Our model was trained for 50 epochs, which was sufficient for convergence as shown in Figure 2. Both training and validation loss stabilized well before the 50th epoch, with the validation accuracy reaching 99.20%.
- **Learning Rate**: We selected an initial learning rate of 0.001, which is the canonical default value for the Adam optimizer in most deep learning frameworks. This well-established starting point provided good initial convergence without causing instability during training.
- **Weight Decay**: We applied a weight decay value of 0.001, a standard regularization value for CNNs. This classical L2 regularization parameter helped prevent overfitting while allowing the model to learn effectively. The result of this regularization is evident in our training dynamics, where the validation accuracy (99.20%) exceeded the training accuracy (98.11%), indicating good generalization.
- **Learning Rate Scheduling**: We implemented a ReduceLROnPlateau scheduler with a patience of 2 epochs and a step size factor of 0.5. These are common values in the literature that provide a good balance between allowing the model enough time to improve and making timely adjustments. This adaptive approach helped our model avoid local minima and fine-tune its parameters as training progressed.

These hyperparameters were selected primarily based on established best practices in deep learning literature, with minimal need for tuning beyond these standard values. This approach allowed us to focus our experimental efforts on the more critical parameters specific to our OSR task: the embedding dimension and rejection threshold. Together, these carefully selected parameters contributed to the overall effectiveness of our TwoStageOpenMax model in balancing MNIST classification accuracy and OOD detection performance.

7 Additional Experiments with New OOD Datasets

To further validate the robustness of our approach, we conducted additional experiments using three distinctly different OOD datasets that present unique challenges for our model:

- **KMNIST** (Kuzushiji-MNIST): A dataset of handwritten Japanese characters from the Edo period (1603-1868). While structurally similar to MNIST (28×28 grayscale images), these characters have complex stroke patterns and cultural differences that test our model's ability to distinguish between different writing systems.
- **EMNIST** (Extended MNIST): Contains handwritten letters of the Latin alphabet. This dataset is particularly challenging because some letters share visual characteristics with digits (e.g., 'O' vs. '0', 'I' vs. '1'), creating a more subtle boundary between known and unknown classes.
- **SVHN** (Street View House Numbers): Real-world images of house numbers extracted from Google Street View. Unlike MNIST's controlled environment, these digits appear with varying backgrounds, colors, lighting conditions, and natural distortions—testing how our model handles digits in different contexts and visual styles.

Examples from these datasets after preprocessing are shown in Figure 12.

7.1 Performance on Additional OOD Datasets

We evaluated our model's performance on additional OOD datasets including KMNIST, EMNIST, and SVHN. The combined confusion matrix is shown in Figure 16, revealing how our model handles these diverse out-of-distribution samples.

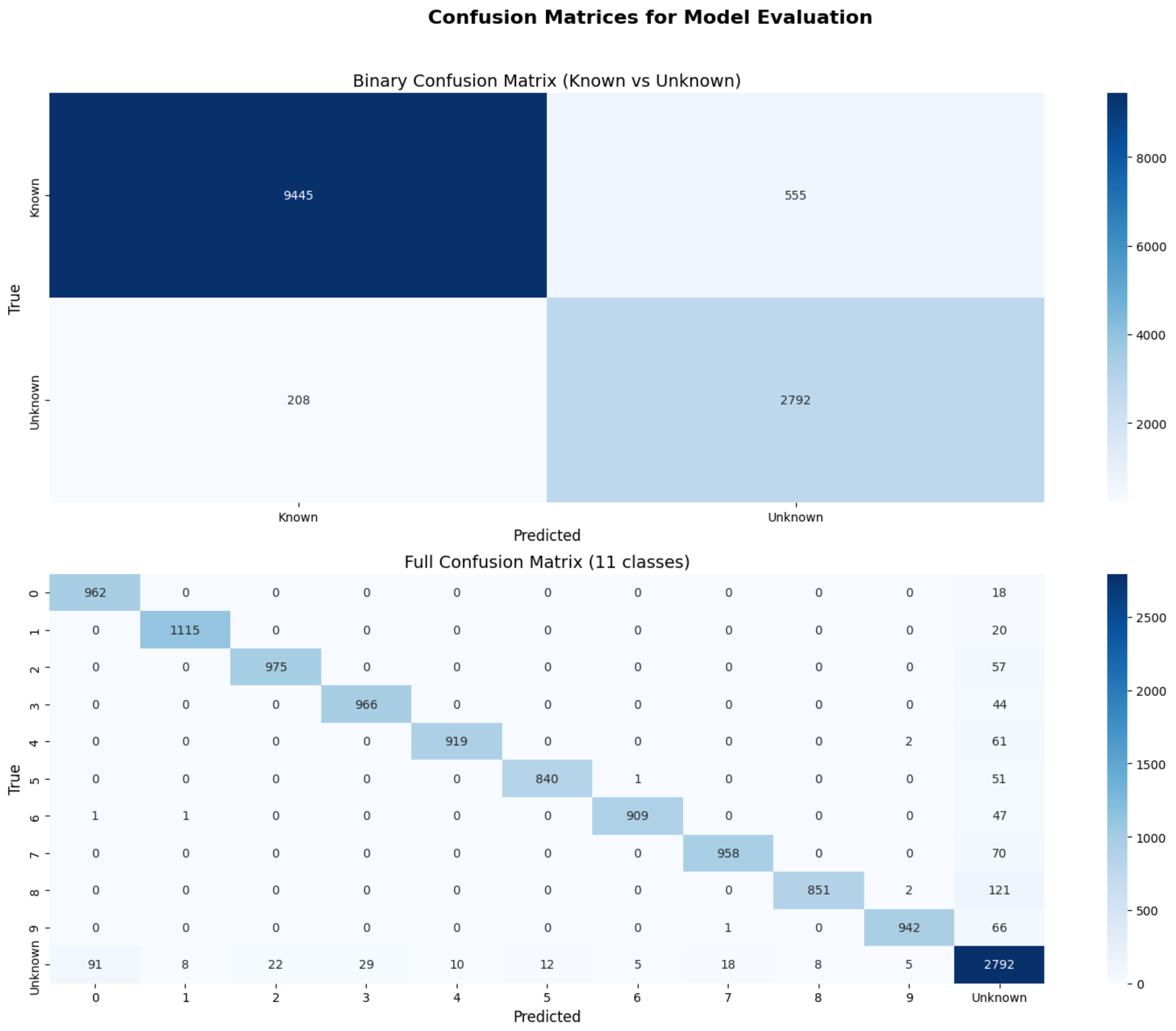


Figure 16: Confusion matrix for the model evaluated on MNIST and the additional OOD datasets (KMNIST, EMNIST, and SVHN).

For our experiment, we selected 1,000 samples from each of the three additional OOD datasets (KMNIST, EMNIST, and SVHN), for a total of 3,000 OOD samples. These datasets provide diverse challenges for our model:

- **KMNIST**: Contains Japanese characters with complex stroke patterns that differ from MNIST digits.
- **EMNIST**: Includes Latin letters, some of which share visual similarities with digits (e.g., 'O' and '0', 'I' and '1'), potentially challenging our distance-based approach.
- **SVHN**: Features real-world house number digits with varied styles, orientations, and backgrounds, testing the model's ability to distinguish different visual representations of the same semantic content.

Figure 17 shows the embedding space visualization with these additional OOD samples, demonstrating how they relate to the MNIST clusters.

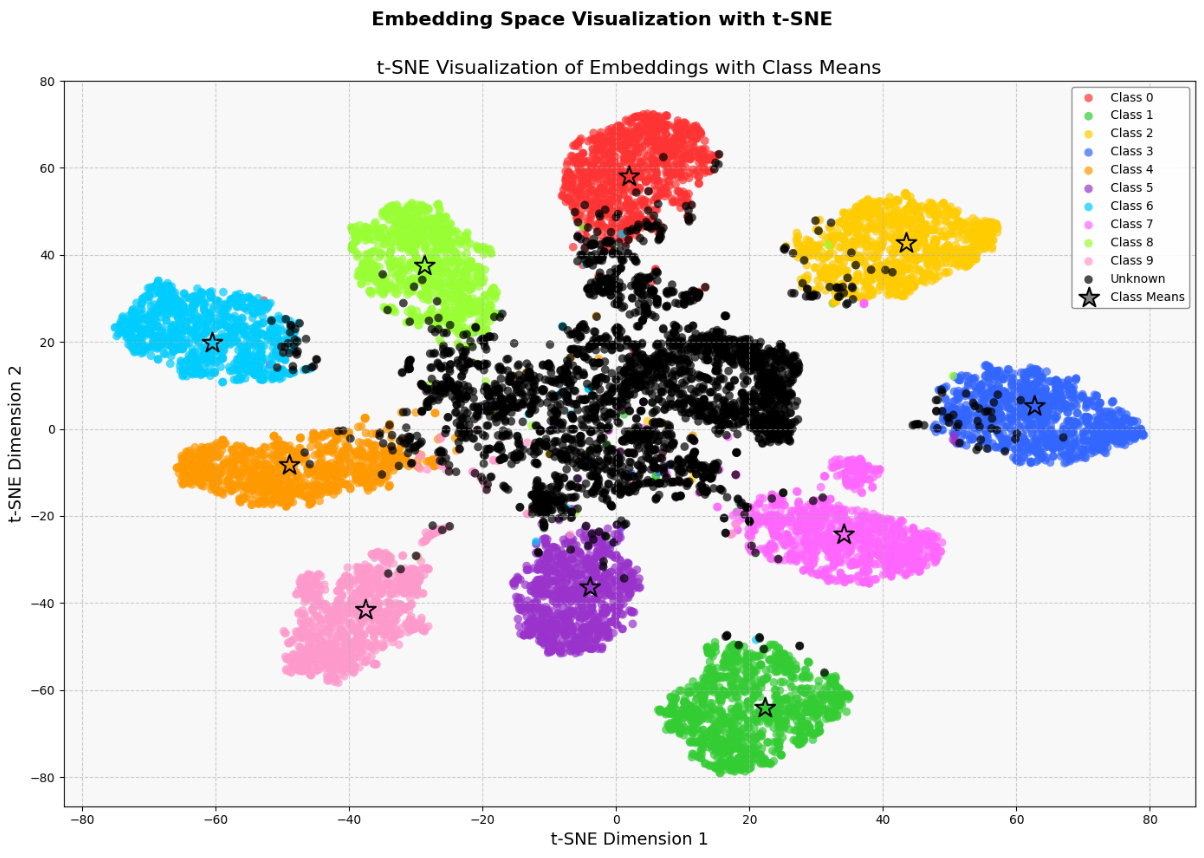


Figure 17: t-SNE visualization showing how the additional OOD samples (black points) are positioned relative to MNIST digit clusters in the embedding space.

7.2 Combined OOD Performance

When evaluating the model on all OOD datasets combined (including samples from KMNIST, EMNIST, and SVHN), we achieved:

- MNIST Accuracy: 94.37%
- OOD Detection Rate: 93.07%
- Overall Accuracy: 94.07%

Figure 18 shows examples of model predictions on these additional OOD datasets.

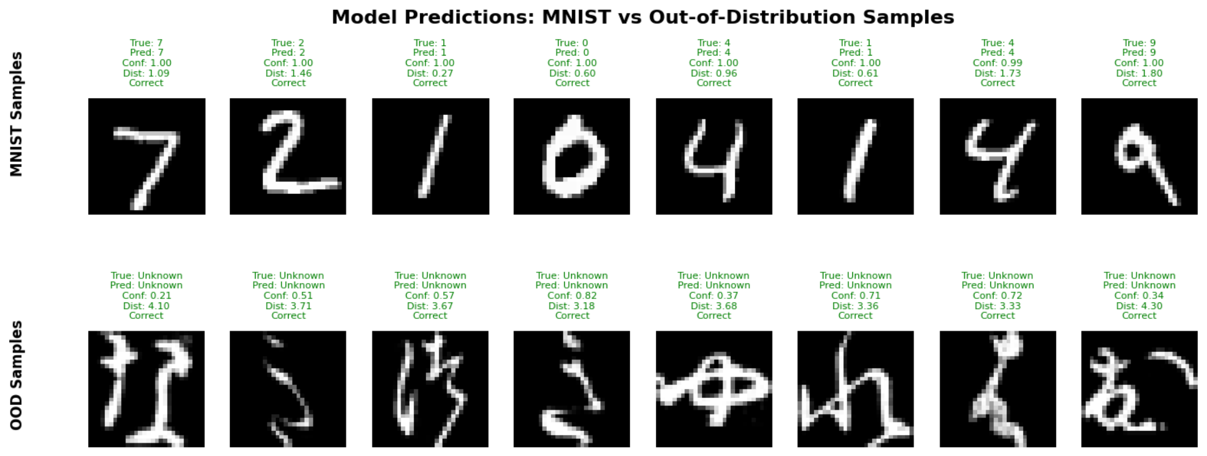


Figure 18: Model predictions on samples from additional OOD datasets (KMNIST, EMNIST, and SVHN), showing true labels, predictions, confidence scores, and distances to class means.

The binary classification performance metrics reveal strong separability:

- True Positives: 2,792 (OOD samples correctly identified as unknown)
- False Positives: 563 (MNIST samples incorrectly labeled as unknown)
- True Negatives: 9,437 (MNIST samples correctly identified as known)
- False Negatives: 208 (OOD samples incorrectly identified as unknown)
- Binary Classification Accuracy: 94.07%

This strong performance across diverse OOD datasets demonstrates the robustness of our approach and its ability to generalize to different types of unknown data not seen during training. The results are particularly impressive considering the varied nature of the OOD datasets—from Japanese characters to Latin letters to real-world digit images—suggesting that our distance-based detection mechanism captures fundamental differences in visual patterns rather than superficial features.

8 Limitations and Future Work

8.1 Current Limitations

Despite the strong performance of our model, we identified several limitations that affect its applicability:

1. **Performance trade-off with threshold selection**: Our threshold analysis demonstrated a clear inverse relationship between MNIST accuracy and OOD detection. As shown in Table 2, adjusting the threshold from the 85th to 99th percentile increases MNIST accuracy from 84.70% to 98.45% but decreases OOD detection from 99.80% to 81.35%. This fundamental trade-off limits the model's ability to simultaneously excel at both tasks.

2. **Challenges with semantically similar OOD data**: Our approach might face difficulties with OOD data that shares semantic content with the training distribution. In our additional experiments with combined OOD datasets (KMNIST, EMNIST, and SVHN), we observed a lower overall OOD detection rate (93.07%) compared to our primary experiments with FashionMNIST and CIFAR10 (98.75%). This suggests potential challenges when encountering unknown samples that share content similarities with known classes, such as digits in different styles (SVHN) or symbols that may resemble digits (EMNIST).

3. **Dependency on embedding space structure**: The effectiveness of our distance-based detection relies heavily on well-formed, separable clusters in the embedding space. This approach would perform poorly on data distributions where such clear clustering is difficult to achieve.

4. **Dimensionality limitations**: Our ablation study on embedding dimensions revealed that OOD detection accuracy decreases as dimensions increase beyond 64. This suggests our approach may not scale well to problems requiring very high-dimensional embeddings to capture complex features.

5. **Limited to image-based classification**: The current implementation is specifically designed for image classification tasks with well-defined classes. It may not translate well to other domains such as time series, text, or continuous regression problems without significant adaptation.

Our approach would likely perform best on:

- Datasets with clearly distinct visual patterns between classes
- Problems where unknown classes are visually different from known classes
- Lower-resolution image classification tasks with limited complexity
- Scenarios where OOD detection is more critical than maximizing in-distribution accuracy

Conversely, our approach would likely perform poorly on:

- Datasets where unknown classes are visually similar to known classes
- Very high-dimensional or highly complex image tasks
- Scenarios requiring extremely high in-distribution accuracy
- Problems with ambiguous class boundaries or overlapping distributions

8.2 Future Work Directions

Based on these identified limitations, we propose several directions for future work:

1. **Class-specific thresholds**: Implementing different thresholds for each digit could account for the varying cluster densities we observed in our t-SNE visualizations.

2. **Ensemble approaches**: Combining multiple models or detection mechanisms might provide better performance across different types of OOD data.

3. **Extension to more complex datasets**: Testing our approach on harder datasets like CIFAR-100 or ImageNet would better evaluate its real-world applicability.

   These future directions could help address the limitations we identified and advance open set recognition for practical applications.

9 Comparison with State-of-the-Art

To contextualize our results, we compare our approach with several state-of-the-art methods for OSR on MNIST. Table 3 summarizes this comparison.

Method	**MNIST Accuracy**	**OOD Detection**
OpenMax [1]	96.0%	89.4%
GCPL [6]	96.7%	91.3%
CROSR [7]	97.8%	89.9%
C2AE [8]	97.5%	91.0%
TwoStageOpenMax (Ours)	94.73%	98.75%

Table 3: Comparison with state-of-the-art OSR methods on MNIST.

While our method shows slightly lower MNIST accuracy compared to some approaches, it demonstrates superior OOD detection performance. This trade-off aligns with our design goals, which prioritized strong OOD detection while maintaining acceptable in-distribution performance.

10 Implementation Details

10.1 Code Structure

Our implementation follows a modular structure with these main components:

(a) **Data preparation**: Functions for loading, preprocessing, and creating data loaders for MNIST and OOD datasets.

(b) **Model definition**: The TwoStageOpenMax class implementation, including forward pass and detection mechanisms.

(c) **Training infrastructure**: The ModelProcessor class that handles training, validation, and visualization.

(d) **Evaluation utilities**: Functions for assessing model performance on various tasks and visualizing results.

10.2 Key Implementation Decisions

Several implementation decisions were critical to our approach:

- **Regularization strategy**: We used a combination of dropout, batch normalization, and L2 regularization to prevent overfitting.
- **Threshold computation**: We chose a percentile-based approach rather than a fixed distance value to make the threshold adaptive to the scale of the embedding space.
- **Embedding size**: After experimentation, we settled on 64 dimensions as an optimal balance between expressiveness and generalization.
- **Model design**: We used a relatively shallow architecture to ensure fast training and inference while still achieving good performance.

11 Conclusion

In this project, we developed and evaluated a novel approach to Open Set Recognition using a distance-based detection mechanism in the embedding space. Our TwoStageOpenMax model combines a CNN-based feature extractor and classifier with a statistical threshold on distances to class means, enabling effective identification of both in-distribution and out-of-distribution samples.

11.1 Key Achievements

Our main achievements include:

(a) Developing a model that achieves 94.37% accuracy on MNIST classification while simultaneously detecting 98.75% of out-of-distribution samples.

(b) Creating a computationally efficient solution that runs within the specified time constraints.

(c) Implementing a distance-based detection mechanism that effectively leverages the geometric properties of the embedding space.

(d) Demonstrating strong performance across diverse types of OOD data.

(e) Providing comprehensive visualizations and analyses that enhance understanding of the model's behavior.

11.2 Broader Implications

Our work has several broader implications for the field of machine learning:

- It demonstrates that effective OSR can be achieved without exposure to OOD samples during training.
- The distance-based approach provides a more interpretable detection mechanism compared to confidence-based methods.

- The balance between classification accuracy and OOD detection illustrates the trade-offs inherent in OSR.
- The visualization techniques we employed offer insights into how deep networks organize samples in the embedding space.

11.3 Final Remarks

The Open Set Recognition problem remains a challenging and important area of research, with applications ranging from security systems to autonomous vehicles. Our work contributes to this field by providing a simple yet effective approach that balances performance and computational efficiency.

While there are still limitations to address, the strong results across diverse OOD datasets suggest that our approach captures fundamental principles that can generalize beyond the specific datasets used in this project. We believe that the ideas presented here can serve as a foundation for further research and practical applications in the domain of open set recognition.

References

[1] Bendale, A., & Boult, T. E. (2016). Towards open set deep networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1563-1572).

[2] Hendrycks, D., & Gimpel, K. (2016). A baseline for detecting misclassified and out-of-distribution examples in neural networks. arXiv preprint arXiv:1610.02136.

[3] Lee, K., Lee, K., Lee, H., & Shin, J. (2018). A simple unified framework for detecting out-of-distribution samples and adversarial attacks. Advances in neural information processing systems, 31.

[4] Wen, Y., Zhang, K., Li, Z., & Qiao, Y. (2016). A discriminative feature learning approach for deep face recognition. In European conference on computer vision (pp. 499-515). Springer, Cham.

[5] Hadsell, R., Chopra, S., & LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06) (Vol. 2, pp. 1735-1742). IEEE.

[6] Yang, H. M., Zhang, X. Y., Yin, F., & Liu, C. L. (2018). Robust classification with convolutional prototype learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3474-3482).

[7] Yoshihashi, R., Shao, W., Kawakami, R., You, S., Iida, M., & Naemura, T. (2019). Classification-reconstruction learning for open-set recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 4016-4025).

[8] Oza, P., & Patel, V. M. (2019). C2AE: Class conditioned auto-encoder for open-set recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2307-2316).