Advanced Out-of-Distribution Detection Leveraging Vision Transformer with Adversarial Attack

JongHyeok Ahn, Leesang Youn, Minho Song

MS. Candidate

University of Seoul

Contents

- Introduction
- Related works
- Method
- Experiment

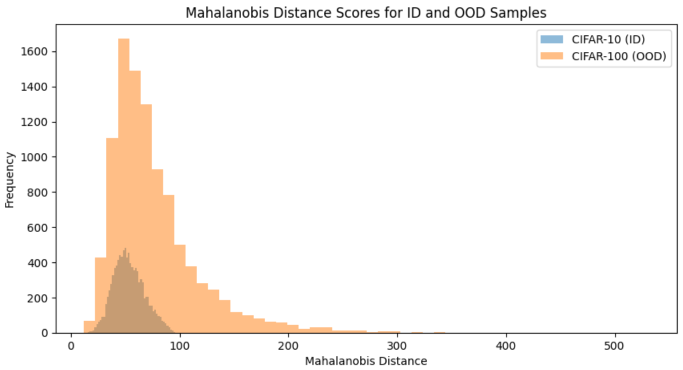
Limitation of prior Out-of-Distribution Detection

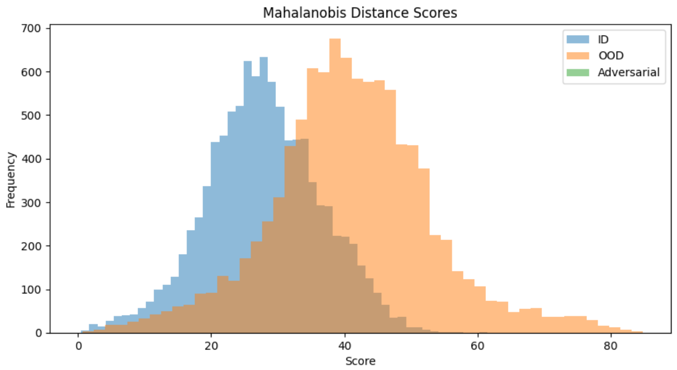
Introduction

Challenged of Near-OOD detection

- Near-OOD data is much more similar to ID data to make it harder for existing methods
- Distinguishing between CIFAR-10 and CIFAR-100 is challenging Near-OOD detection

- To overcome these limitations, we proposed the Attention Masking method to improve model to more accurately distinguish between ID and OOD data

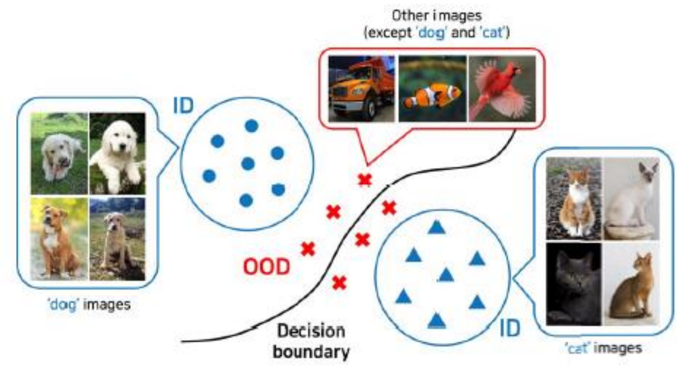




Objective of Attention Masking OOD detection

Introduction

- Utilizing Vision Transformer(ViT) Models
- Identifying and preprocessing important patches
  - This approach allows the model to focus more on significant parts of the image
- Applying Mahalanobis distance calculation
- Near Out-of-Distribution image classification
  - Got improved results on CIFAR-10(in) vs. CIFAR-100(out) and vice versa



Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks

Shiyu Liang
Coordinated Science Lab, Department of ECE
University of Illinois at Urbana-Champaign
sliang26@illinois.edu

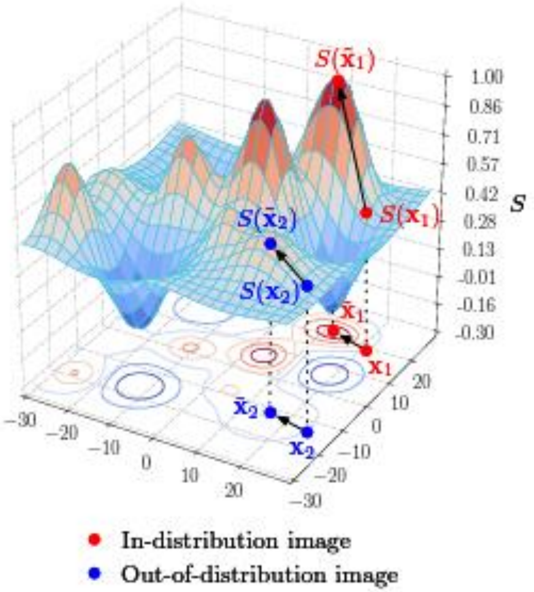
Yixuan Li
University of Wisconsin-Madison*
sharonli@cs.wisc.edu

R. Srikant
Coordinated Science Lab, Department of ECE
University of Illinois at Urbana-Champaign
rsrikant@illinois.edu

Input pre-processing(ODIN)

Related works

- In contrast to traditional FGSM (Fast Gradient Sign Method), which focuses on reducing the max probability by adding noise
- The approach in the referenced study introduces noise to increase the confidence score.
- The input pre-processing equation used in this method is:
  - $\hat{x} = x - \epsilon \cdot sign(\nabla_x \log S(x)) = x - \epsilon \cdot sign(\nabla_x L(x))$
  - We call this method as Reversed FGSM



Attention Masking for Improved Near Out-of-Distribution Image Detection

Minho Sim
School of Computing
KAIST
Daejeon, Republic of Korea
smh3946@kaist.ac.kr

Jongwhoa Lee
School of Computing
KAIST
Daejeon, Republic of Korea
jongwhoa.lee@kaist.ac.kr

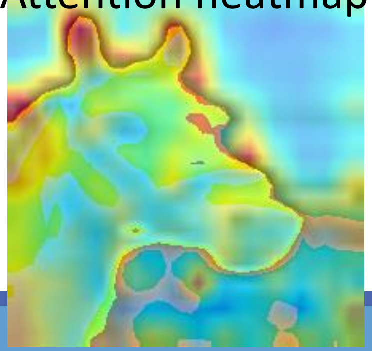
Ho-Jin Choi
School of Computing
KAIST
Daejeon, Republic of Korea
hojinc@kaist.ac.kr

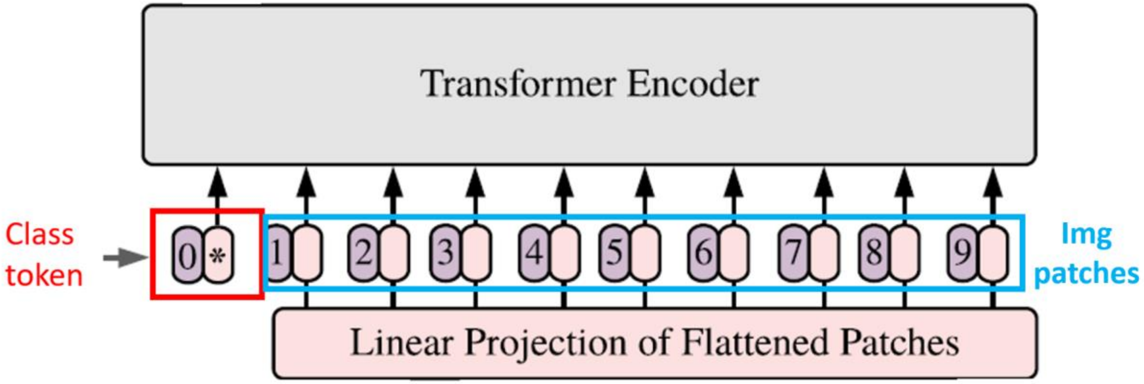
CLS Token for Attention Rollout Algorithm

Related works

- **Class Token(CLS Token)**
  - CLS Token is a special token added to the input of the ViT model that learns the summary information of the entire image.
  - In each Transformer layer, the CLS token interacts with other patches through the attention mechanism, ultimately aggregating the important information of the entire image.
- **Attention Weights**
  - Attention rollout heatmaps show how much the CLS token focuses on the image. This is what attention weights represent.

Attention heatmap

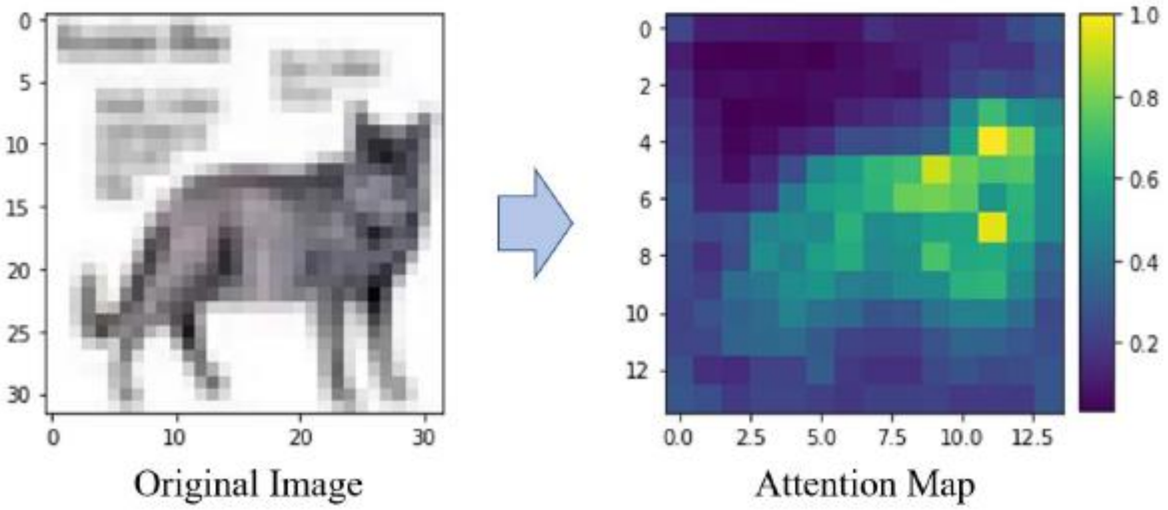




Visualization of Attention Mechanism

methods

- Visualizing attention maps helps in understanding the model's decision-making process.
- The attention mechanism allows the model to focus on the most relevant parts of the image.

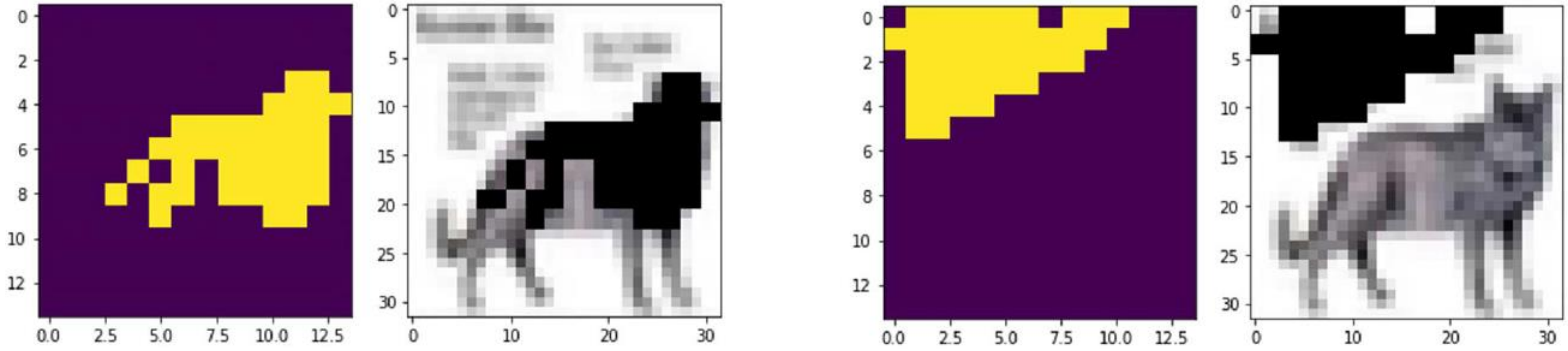


Masking strategy

Related works

- In this paper the ratio was set to 20%

Masking Strategy	Patches to be Masked
Top-Ratio	Top N% of attention values
Bottom-Ratio	Bottom N% of attention values



(a) top-ratio (b) bottom-ratio

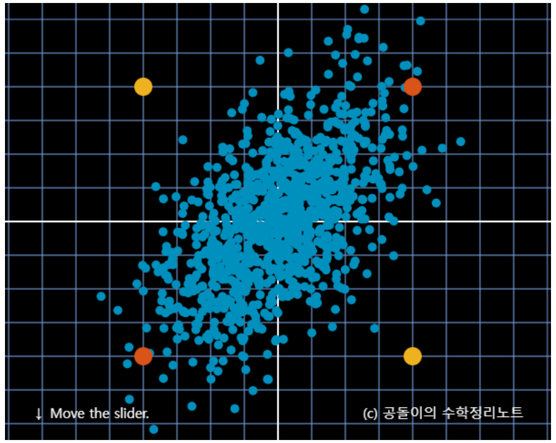
How can we detect OODs?

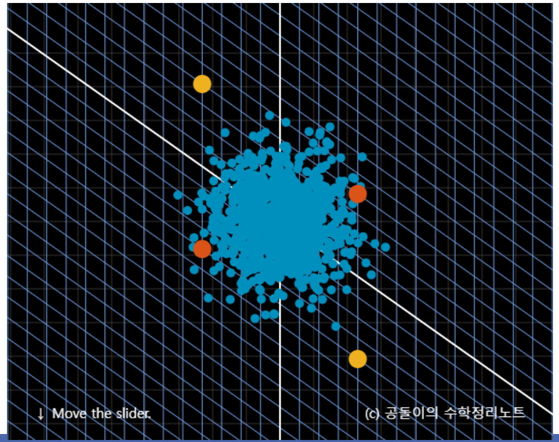
- The feature is extracted from the penultimate layer to capture high-level abstract representations of the input
- Calculate the distance between the instance penultimate layer & mean penultimate vector of ID distribution.
  - Close distance: ID
  - Far distance: OOD
- Which distance should we implement?

Mahalanobis distance based OOD detection

Mahalanobis distance is a measure of the distance between a point and a distribution.

- It is used to identify how far a point is from the mean of a distribution, normalized by the covariance of the distribution.
- In the context of OOD detection, it helps in distinguishing between ID and OOD samples based on their distance from the ID distribution.





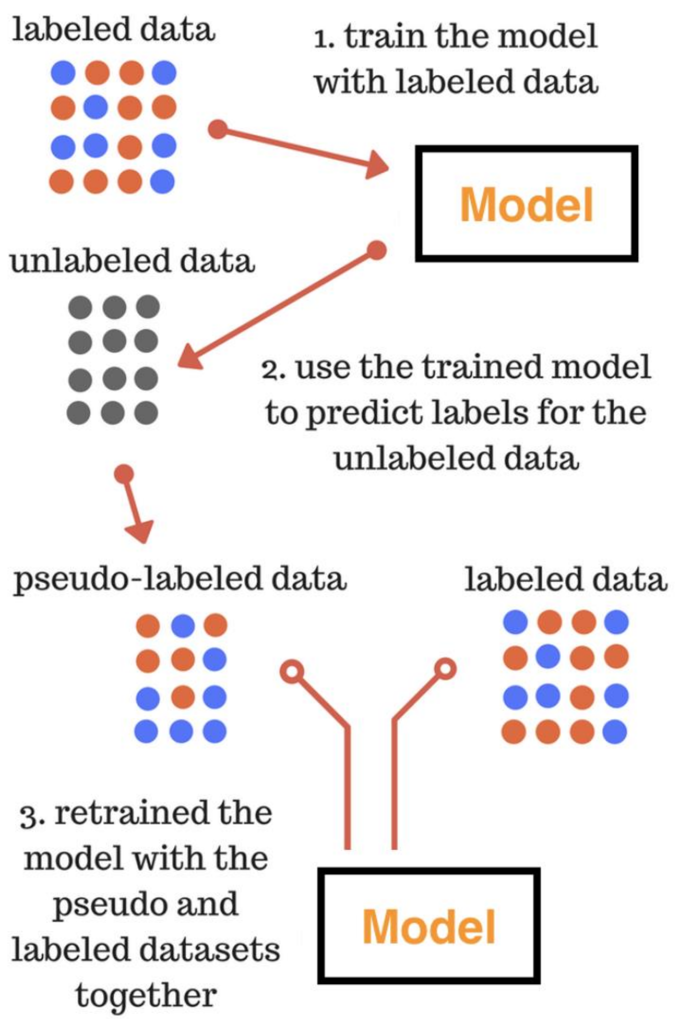
Mahalanobis distance based OOD detection

- $\widehat{\mu_c} = \frac{1}{N_c} \sum_{i;y_i \in c} f(x_i)$, c : class label
- $\widehat{\sum} = \frac{1}{N} \sum_c \sum_{i:y_i=c} (f(x_i) - \widehat{\mu_c})(f(x_i) - \widehat{\mu_c})^T$
- $M(x) = \min_c (f(x_i) - \widehat{\mu_c})^T \widehat{\sum^{-1}} . (f(x_i) - \widehat{\mu_c})$
- Attention Masking effectively helps in differentiating ID and OOD samples.
- By focusing on the most relevant parts of the image, the model enhances its ability to distinguish between ID and OOD data
  - 1.Assume a Gaussian distribution for each of the 10 labels(Cifar-10)
  - 2. Calculate the Mahalanobis distance for each distribution
  - 3. Select the nearest label

Pseudo Labeling for input pre-processing

methods

- Pseudo-labeling is applied in our method, where OOD(Out-of-distribution) data is unknown .
- In our method, unknown data is pseudo-labeled with the nearest label.
- $\hat{x} = x - \epsilon \cdot sign\left(\nabla_x L(x)\right),\ L(x)$: CE loss
  - Target is unknown for test dataset



Input pre-processing

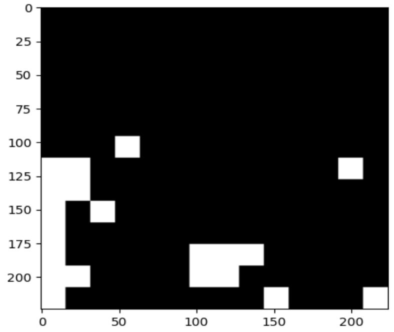
methods

- Reduce the distortion of the original image
- So, input pre-processing is applied to partial patches with attention scores strategies.
- The input pre-processing equation used in this method is:
  - $\hat{x} = x - \epsilon \cdot sign\left(\nabla_x L(x)\right),\ L(x)$: CE loss with pseudo label

Pipeline of Attention Adversarial OOD

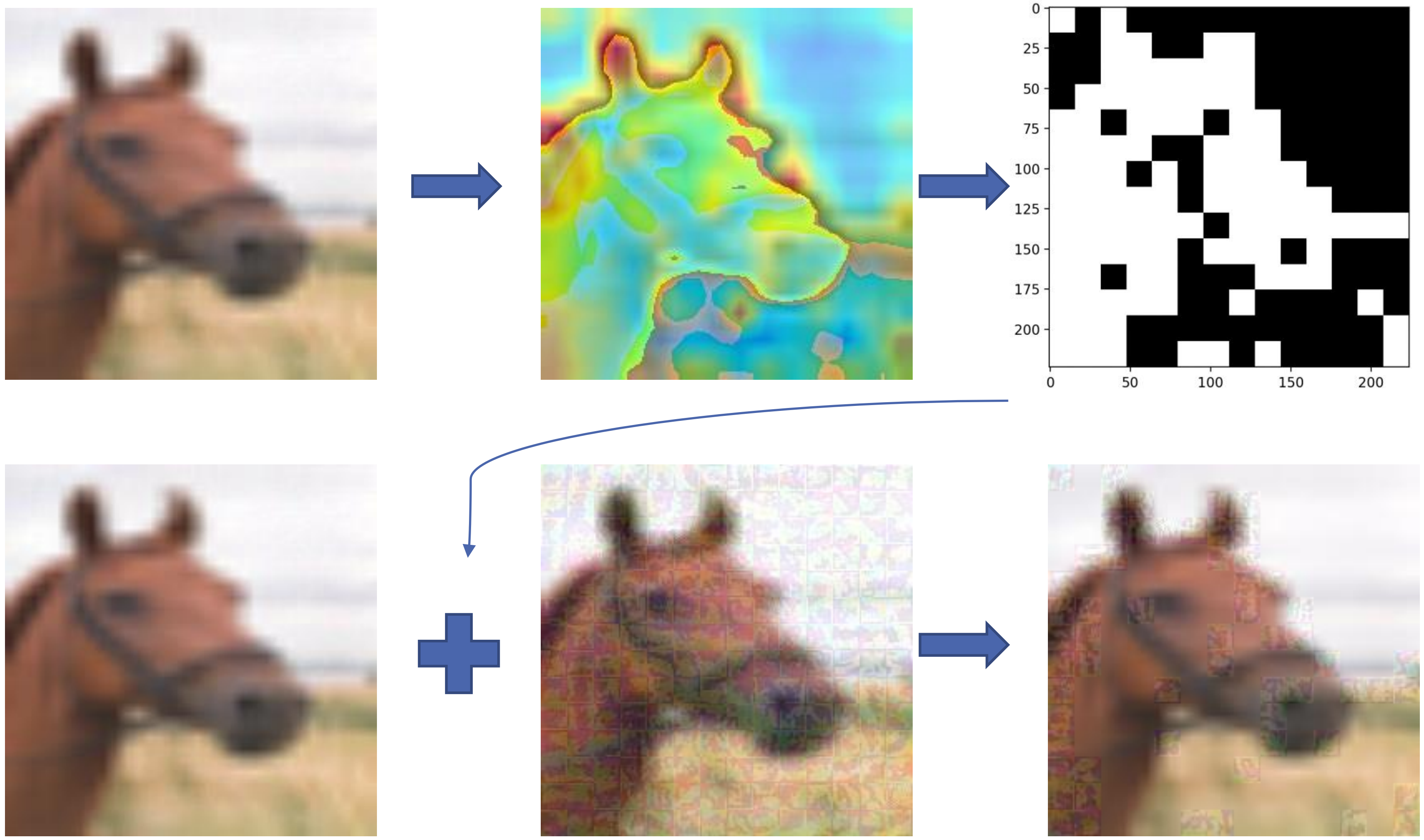
methods

- 1. The process begins with the input image. This is a raw image from the dataset
- 2. The input image is divided into smaller, non-overlapping patches. Each patch is processed individually by the Vision Transformer (ViT) model
- 3. The Vision Transformer model computes attention scores for each patch.
  - An attention map is generated, highlighting the areas of the image that the model focuses on the most.
- 4. Based on the attention scores, more important patches are identified and subtract adversarial noise.
  - This step enhances the model's focus on the significant parts of the image by removing or de-emphasizing the less relevant areas.
- 5. Generate perturbed image
  - Adversarial image by fgsm attack
  - Adversarial image * masking + original image * (1 – masking)



Pipeline of Attention Adversarial OOD

methods



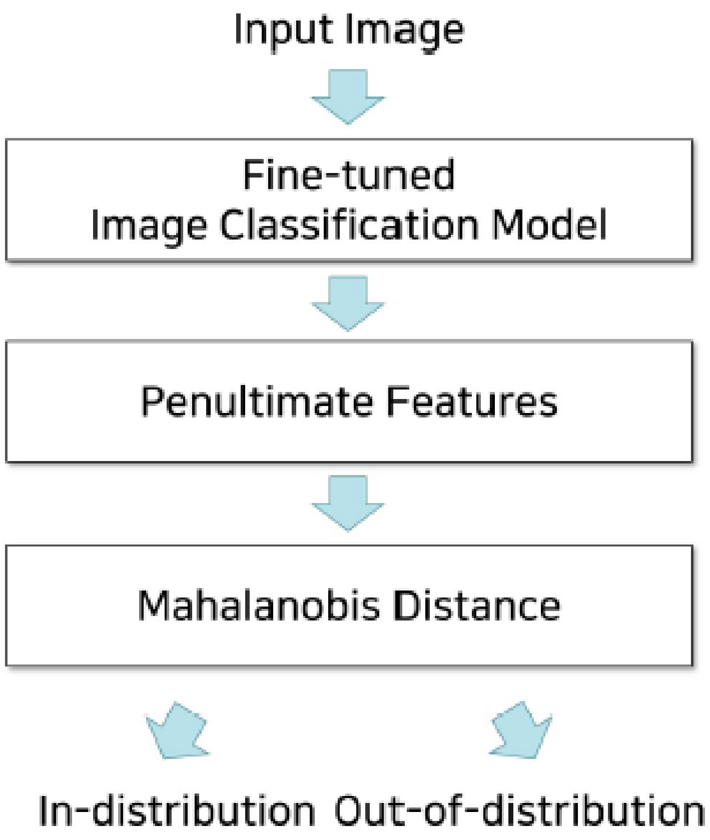
Pipeline of Attention Adversarial OOD

methods

- 6. Input the masked image back into the ViT model to extract new feature vectors(F_{adv}).
- 7. Mahalanobis Distance Calculation and classify ood
  - Mahalanobis distance is calculated using the feature vector extracted from the penultimate layer of the ViT model through which the masked image is passed.
  - Mahalanobis distance is used to measure how much a given feature vector deviates from the training data distribution
    - $D_M = \sqrt{(F_{adv} - \mu)^T \sum^{-1} (F_{adv} - \mu)}$

Pseudo code

methods



Algorithm 1 Differential masking OOD detection

Require: Pre-trained ViT model, Input image I, OOD detection score threshold T, Perturbation ϵ
Ensure: OOD Detection score

Step 1: Extract Attention Weights
Initialize the ViT model with pre-trained weights
Extract attention weights from all transformer layers for input I
Step 2: Compute Attention Map
$A_{\text{rollout}_0} = I$ (Identity matrix)
for each layer L in ViT **do**
  Compute attention scores A_L for each patch in I
  $A_{\text{rollout}_L} = (A_L + I) \cdot A_{\text{rollout}_{L-1}}$
end for
$A_{\text{final}} = A_{\text{rollout}_L}$
Step 3: Identify Important Patches
Determine threshold T for masking based on attention scores
for each patch p in I **do**
  if Attention score of $p > T$ **then**
    Point patch p
  end if
end for
Step 4: Adversarial Noise
Set parameters: ϵ, iterations, τ
for each image I, label in dataset **do**
  Initialize image gradient
  for 1 to iterations **do**
    Compute loss and gradients
    Apply attention filter: $grad \leftarrow grad \times (attention > \tau)$
    Update image: $I \leftarrow I - \frac{\epsilon}{\text{iterations}} \cdot \text{sign}(grad)$ for pointed patches only
    Clamp image to valid range
  end for
  Evaluate model accuracy with and without partial input preprocessing
end for
Step 5: Evaluate Updated Image
Forward perturbed image $I_{\text{perturbed}}$ through ViT model
Extract penultimate layer features $F_{\text{perturbed}}$
Calculate Mahalanobis distance D_M using $F_{\text{perturbed}}$ and training distribution statistics
Step 6: OOD Detection
Determine OOD score based on D_M
if D_M > threshold **then**
  Classify I as OOD
else
  Classify I as ID
end if
return OOD Detection score

Experiment

- **Datasets**: CIFAR-10(id), CIFAR-100(ood)
- **Models**: CIFAR-10-finetuned ViT
- **Image Processing:**
  - CIFAR images scaling 32x32 to 224x224
- **Metrics**: AUROC, AUPR_ood
- **Experiment**:
  - Near-OOD detection with fine-tuning
- **Baseline method:**
  - Mahalanobis Distance

Result

experiment

Table 1: Perturb Attention Top 50%

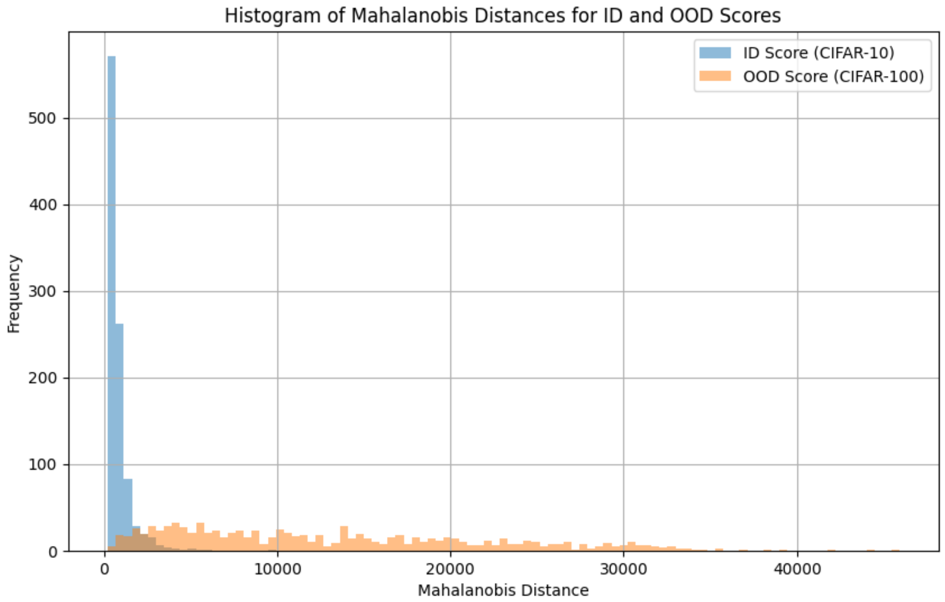
Metric	baseline	$\epsilon = 0.1$	$\epsilon = 0.01$	$\epsilon = 0.001$	$\epsilon = -0.001$	$\epsilon = -0.01$
auroc	0.9857	0.9371	0.9379	0.9846	0.9859	0.9348
aupr	0.9875	0.9465	0.945	0.9865	0.9877	0.9463

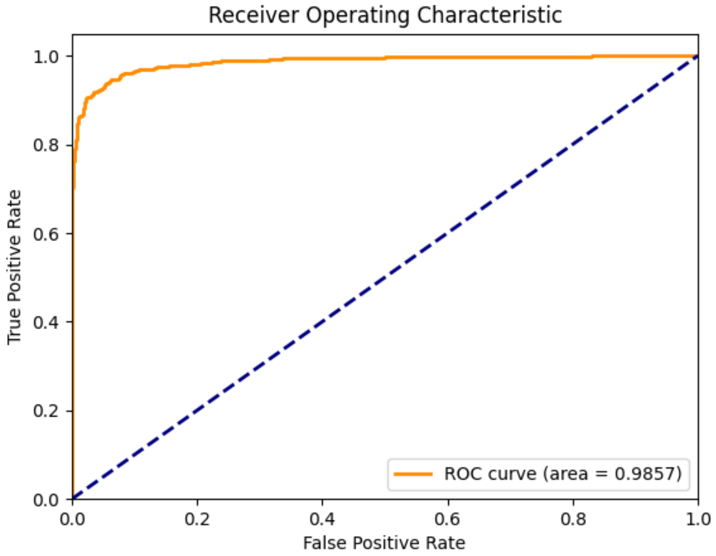
Table 2: Perturb Attention Bottom 50%

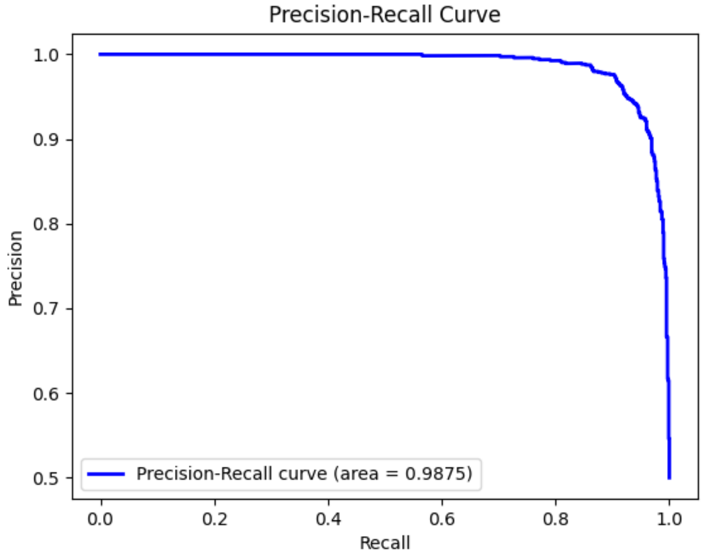
Metric	baseline	$\epsilon = 0.1$	$\epsilon = 0.01$	$\epsilon = 0.001$	$\epsilon = -0.001$	$\epsilon = -0.01$
auroc	0.9857	0.8607	0.8907	0.9844	0.9860	0.8544
aupr	0.9875	0.8782	0.9008	0.9846	0.9880	0.8991

baseline

experiment



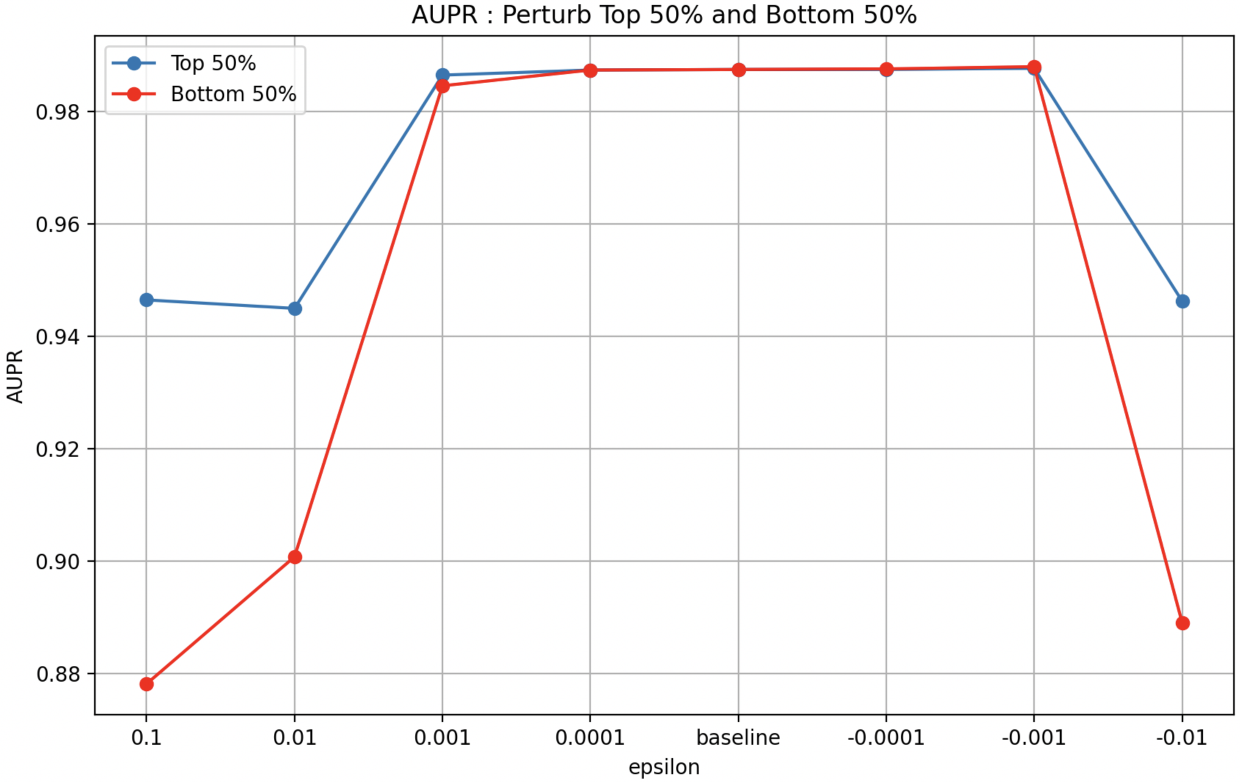


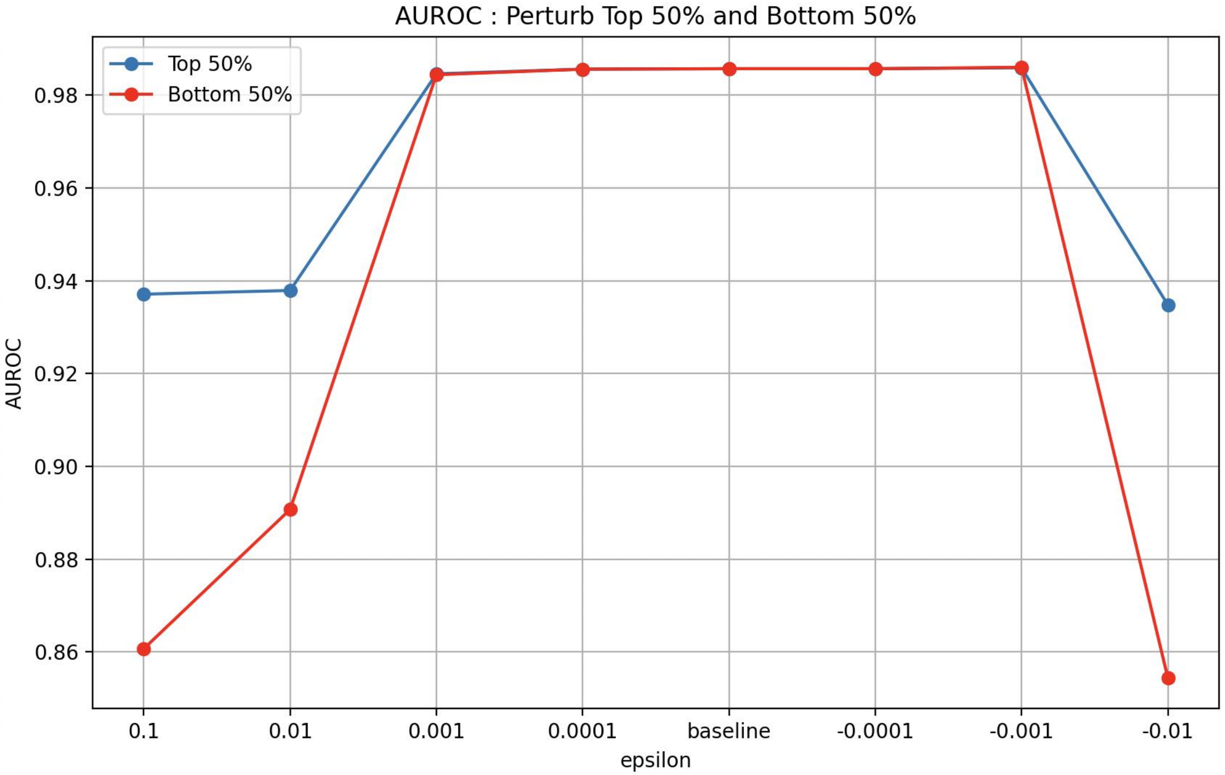


Result

experiment

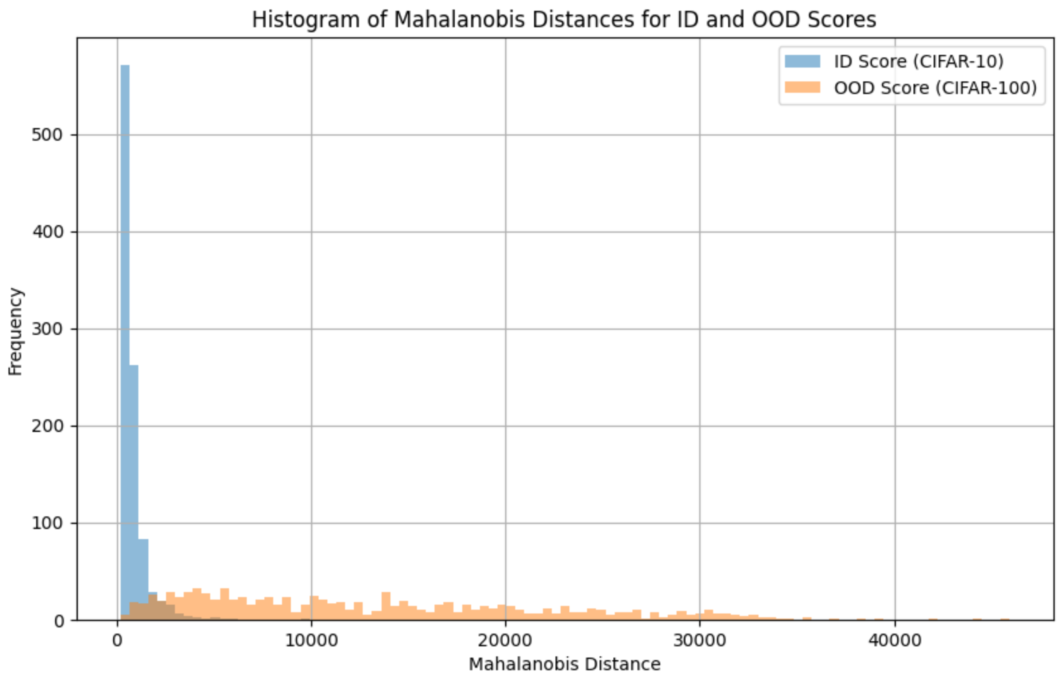
- Compare two strategy top 50% and bottom 50%



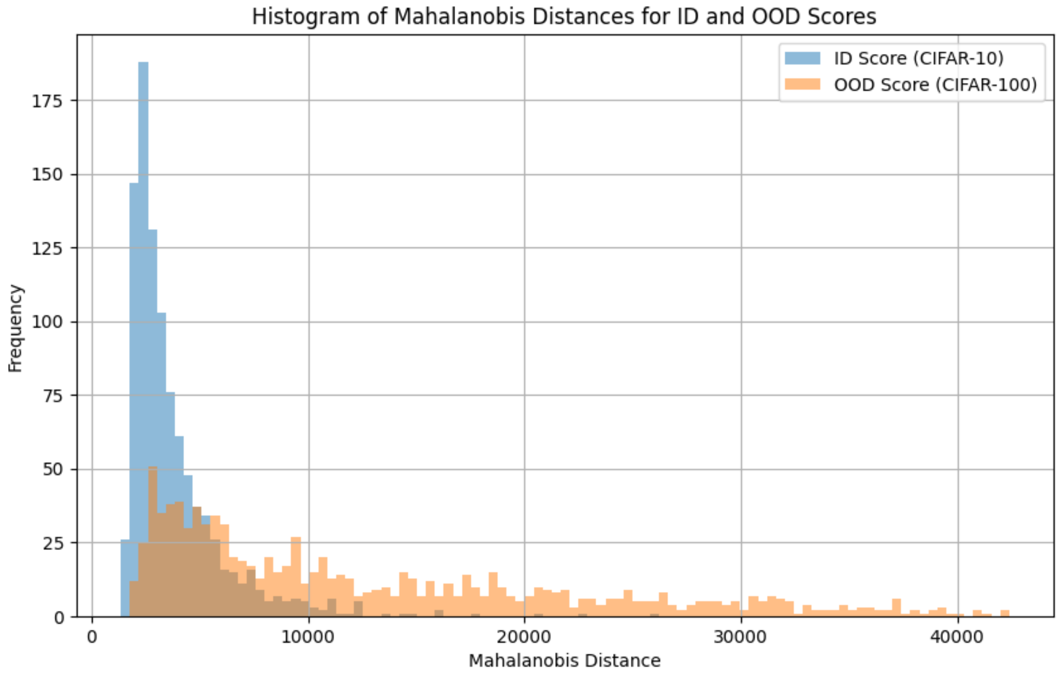


Result

Baseline



$\epsilon = 0.1$



Limitation

- **Limitation of method**
  - Dataset is too easy, making it difficult to confirm whether detection is being performed accurately. So we should try using a different dataset.
  - The base performance of ViT is too high
  - Result may depend on fine-tuned ViT model

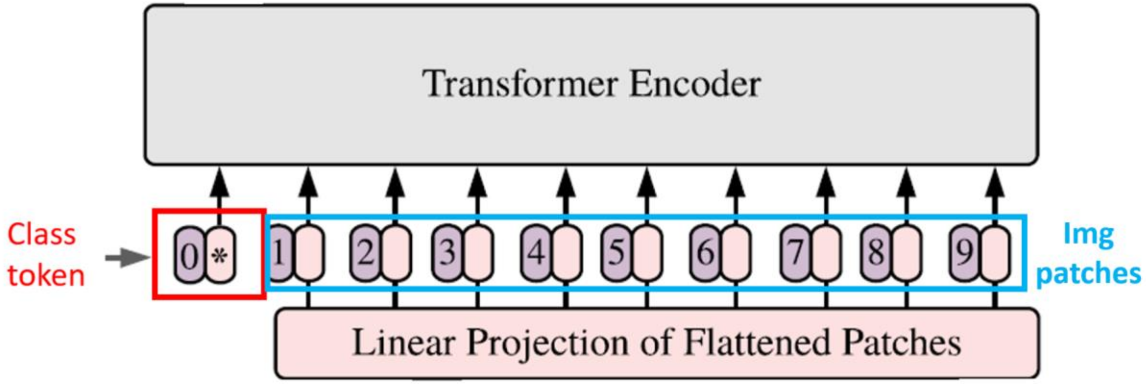
Appendix

CLS Token for Attention Rollout Algorithm

appendix

- **Calculation of Attention Rollout**
  - Attention Rollout calculates the degree to which the CLS token attends to the patches at each layer.
  - To do this, it sequentially multiplies the attention weights of each layer to compute the final attention map.

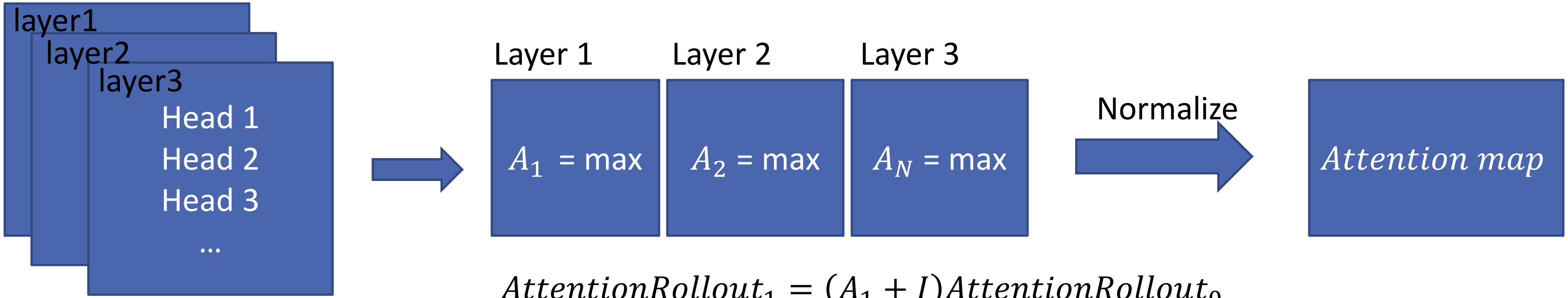
- **Conclusion**
  - The Attention Rollout Algorithm calculates how much the CLS token attends to each patch to generate the final attention map.
  - This final map visually expresses the importance of each patch in the model's final output.
  - In other words, it calculates and visualizes the importance of patches based on the attention given by the CLS token to each patch.



Attention rollout algorithm

appendix

- Proposed at “Quantifying Attention Flow in Transformers”
  - Visualizes and interprets how attention mechanisms in transformers contribute to model decisions.
  - Visualizes the effect of multi-head attention in a single graph.
- Attention rollout Visualize multi-head Attention`s effect by one graph



$$AttentionRollout_1 = (A_1 + I)AttentionRollout_0$$

$$AttentionRollout_L = (A_L + I)AttentionRollout_{L-1}$$

Attention rollout algorithm

appendix

- **Process of Attention rollout**
  - 1. **Extract Attention Weights**: $A_L = Attention\ Weights\ from\ Layer\ L$
    - For each transformer layer, extract the attention weights for all attention heads.
  - 2. **Head Fusion**: $A_L = \frac{1}{h}\sum_{k=1}^{h} A_L^k$ or $A_L = \max_{k=1}^{h} A_L^k$
    - Integrate the attention weights from multiple heads into a single matrix by averaging or taking the maximum values.
  - 3. **Recursive Aggregation**: $AttentionRollout_L = (A_L + I)AttentionRollout_{L-1}$
    - Here, A_{ij} represents how much the j-th patch in the previous layer attends to the i-th patch in the current layer.
    - $AttentionRollout_0$ is identity matrix
    - Calculate the aggregated attention rollout using the attention weights.
    - Starting from the final layer, recursively multiply the attention weights through the layers to aggregate the overall attention for each token.
  - 4. **Normalize Attention Scores**: $NormalizedAttention_L = \frac{AttentionRollout_L}{\sum_{k=1}^{N} AttentionRollout_{ij}}$
    - Normalize the aggregated attention scores to ensure they sum to 1, providing a clear distribution for focus
  - 5. **Visualization**
    - Visualize the result attention map