Machine Learning Homework 8
Anomaly Detection

ML TAs
ntu-ml-2021spring-ta@googlegroups.com
Goal

● Unsupervised anomaly detection in computer vision: Whether a machine learning model is able to tell a testing image is of the same class (distribution) as the training images
Goal

- Unsupervised anomaly detection in computer vision: Whether a machine learning model is able to tell a testing image is of the same class (distribution) as the training images.

Training

Model
Goal

- Unsupervised anomaly detection in computer vision: Whether a machine learning model is able to tell a testing image is of the same class (distribution) as the training images.
Goal

- Unsupervised anomaly detection in computer vision: Whether a machine learning model is able to tell a testing image is of the same class (distribution) as the training images.
Data

- Trainingset: About 140k human faces (size 64*64*3)
- Testingset: Another 10k data from the same distributions as the trainingset (normal data, of class label 0) along with 10k human face images from the other distributions (anomalies, of class label 1)
- Notice: Additional training data and pretrained models are prohibited
- Data format: tar zxvf data-bin.tar.gz
- data-bin/
  - trainingset.npy
  - testingset.npy
Method - Autoencoder

**Training**
- Training data
- Encoder
- Code
- Decoder
- Training data

As close as possible

**Testing**
- Anomaly
- Encoder
- Code
- Decoder
- Cannot be reconstructed
Autoencoder

- When to stop training? Training should stop when the mse loss converges.
- During inference, we calculate the reconstruction error between the input image and the reconstructed one.
- The reconstruction error will be referred to as abnormality (anomaly score).
- The abnormality of an image can be a metric of the possibility that it’s distribution is unseen during training.
- Therefore we use the abnormality as our predicted values.
Accuracy score

- Usually we compute accuracy scores for classification tasks.
- Here, our model functions as a sensor (or a detector) rather than a classifier.
- Thus, we need a threshold with respect to abnormality (usually the reconstruction error) to determine whether a piece of data is an anomaly.
- If we used accuracy score for this assignment, you would have to try every possible threshold for one single model to get a satisfactory score.
- However, what we want is a sensor that gets the highest accuracy on the average of every possible threshold.
Which sensor is better?
Metric - ROC_AUC score

- A good sensor should
  - Give high anomaly scores to the anomalies and low scores to the normal data
  - Exhibit a large gap between the scores of 2 groups

- An ROC is suitable for our task
- Each point on the ROC curve stands for true positive rate and false positive rate at certain threshold
- The Area Under the ROC curve is calculated to measure the general ability of the model
ROC_AUC score

https://en.wikipedia.org/wiki/Receiver_operating_characteristic
ROC_AUC score

https://en.wikipedia.org/wiki/Receiver_operating_characteristic
Kaggle

Metric: ROC_AUC score
Sample output:

<table>
<thead>
<tr>
<th>Id, Predicted</th>
</tr>
</thead>
<tbody>
<tr>
<td>0, 10000.0</td>
</tr>
<tr>
<td>1, 10000.0</td>
</tr>
<tr>
<td>2, 10000.0</td>
</tr>
<tr>
<td>3, 10000.0</td>
</tr>
<tr>
<td>4, 10000.0</td>
</tr>
<tr>
<td>5, 10000.0</td>
</tr>
<tr>
<td>6, 10000.0</td>
</tr>
<tr>
<td>7, 10000.0</td>
</tr>
<tr>
<td>8, 10000.0</td>
</tr>
<tr>
<td>9, 10000.0</td>
</tr>
<tr>
<td>10, 10000.0</td>
</tr>
<tr>
<td>11, 10000.0</td>
</tr>
<tr>
<td>12, 10000.0</td>
</tr>
<tr>
<td>13, 10000.0</td>
</tr>
<tr>
<td>14, 10000.0</td>
</tr>
</tbody>
</table>
## How ROC AUC is calculated

### Calculation Details

The area under the ROC curve (AUC) is calculated using the trapezoidal rule.

### Table 1

<table>
<thead>
<tr>
<th>ID</th>
<th>Anomaly Score</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>11383</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>256676</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>862365</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>152435</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>848171</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 2

<table>
<thead>
<tr>
<th>ID</th>
<th>Anomaly Score</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>862365</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>848171</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>256676</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>152435</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>11383</td>
<td>0</td>
</tr>
</tbody>
</table>

### Sort by Score

The data is sorted by the anomaly score, which is crucial for calculating ROC AUC.

---

[https://towardsdatascience.com/how-to-calculate-use-the-auc-score-1fc85c9a8430](https://towardsdatascience.com/how-to-calculate-use-the-auc-score-1fc85c9a8430)
# How ROC AUC is calculated

<table>
<thead>
<tr>
<th>ID</th>
<th>Anomaly score</th>
<th>Label</th>
<th>fp before normalization</th>
<th>tp before normalization</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>862365</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>848171</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>256676</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>152435</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>11383</td>
<td>0</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>
How ROC AUC is calculated

<table>
<thead>
<tr>
<th>ID</th>
<th>Anomaly score</th>
<th>Label</th>
<th>fp</th>
<th>tp</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>11383</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>152435</td>
<td>0</td>
<td>0</td>
<td>0.333333</td>
</tr>
<tr>
<td>1</td>
<td>256676</td>
<td>1</td>
<td>0.333333</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>848171</td>
<td>0</td>
<td>0.666667</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>862365</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Area Under Curve: $0.5 \times \frac{1}{3} + \frac{2}{3} = 0.8333$
Scoring

- Code submission: 4 pt
- Baselines 6 pt (3 pt for the public ones and the other 3 pt for the private ones)
  - Simple public: 1 pt  (public score: 0.64046)
  - Medium public: 1 pt  (public score: 0.75719)
  - Strong public: 0.5 pt  (public score: 0.81304)
  - Boss public: 0.5 pt  (public score: 0.86590)
  - Simple private: 1 pt
  - Medium private: 1 pt
  - Strong private: 0.5 pt
  - Boss private: 0.5 pt
- Bonus for submitting report: 0.5 pt
If you succeed in beating both boss baselines, you can get extra 0.5 pt by submitting a brief report to explain your methods (in less than 100 English words), which will be made public to the whole class

Report Template
Baseline guides

- **Simple**
  - FCN autoencoder

- **Medium**
  - CNN autoencoder
  - Try smaller models (less layers)
  - Smaller batch size

- **Strong**
  - Add BatchNorm
  - Train for longer

- **Boss:**
  - Add an extra classifier
  - Sample random noises as anomaly images
  - Or one-class-classification (OCC) with GANs: OCGAN, End-to-end OCC, paper pool for Anomaly Detection
Baseline training statistics

- **Simple**
  - Number of parameters: 3176419
  - Training time on colab: ~ 30 min

- **Medium**
  - Number of parameters: 47355
  - Training time on colab: ~ 30 min

- **Strong**
  - Number of parameters: 47595
  - Training time on colab: 4 ~ 5 hrs

- **Boss:**
  - Number of parameters: 4364140
  - Training time on colab: 1.5~3 hrs
Strong baseline training curve

![Graph showing a strong baseline training curve with decreasing loss as steps increase. The x-axis represents steps from 0 to 1k, and the y-axis represents loss from 3e-3 to 0.]
Code Submission

- **Zip** your code and name the compressed file `<student_id>_hw8.zip`
- And it should contain
  - Your codes
  - report.pdf (only for those beating both boss baselines)
- Submit `<student_id>_hw8.zip` via NTU COOL
Code Submission

● DO
  ○ Specify the source of your code. (You may refer to Academic Ethics Guidelines)
  ○ Organize your code and make it easy to read (not necessary)

● DO NOT
  ○ Submit empty or garbage files
  ○ Submit the dataset or model
  ○ Compress your codes into other formats like .rar or .7z and simply rename it to .zip

● Note
  ○ We can only see your last submission
  ○ Do not submit your model or dataset
  ○ If your code is not reasonable, your semester grade \* 0.9
Regulations

- Plagiarism is not allowed
- Do not modify your prediction file
- Do not share your prediction file with anyone
- Do not submit your prediction file more than 5 times to Kaggle in any way
- **Do NOT search or use additional data or pre-trained models.**
- Violators are subject to \( x \ 0.9 \) of their semester grades
- Prof. Lee & TAs preserve the rights to change the rules & grades
Important dates

- Kaggle deadline: 5/21 23:59 (GMT+8)
- Code & report deadline: 5/23 23:59 (GMT+8)
- Late submissions are NOT allowed
Links

- Kaggle: https://www.kaggle.com/c/ml2021spring-hw8
- Colab: https://colab.research.google.com/drive/1D_8lkhzLfoVhA6bTef-Yw82o6P4g1rQ?usp=sharing
Contact TAs

● NTU COOL (recommended)
  ○ https://cool.ntu.edu.tw/login/portal

● Email:
  ○ ntu-ml-2021spring-ta@googlegroups.com
  ○ The title should begin with [hw8]

● TA hour
  ○ Each Monday 19:00 ~ 21:00 at Room 101, EE2 (電二 101)
  ○ Each Friday before (13:30 ~ 14:20) & during class at Lecture Hall