HW1

2014314143 김규용

Q3. Handling outliers in MNIST or any image classification problem involves detecting and removing them during data preprocessing or addressing them during model training. Here are some general approaches to handle outliers in MNIST:

Data Augmentation: One way to handle outliers is to use data augmentation techniques such as random cropping, flipping, or rotation. These techniques generate new images from the original ones, making the model more robust to outliers.

Outlier Detection: Another approach is to detect outliers during data preprocessing. Outliers can be identified using clustering or density-based methods such as k-means or DBSCAN. Once the outliers are detected, they can be removed from the dataset.

Semi-Supervised Learning: In some cases, outliers are caused by mislabeling of data. Semi-supervised learning can help in such cases, where the model learns from both labeled and unlabeled data. Unlabeled data can help the model identify mislabeled data, which can then be removed from the dataset.

Ensemble Models: Ensemble models combine multiple models to make a prediction. By combining the outputs of several models, ensemble models can be more robust to outliers. For example, if one model makes an incorrect prediction due to an outlier, the other models in the ensemble can help correct it.

Robust Loss Functions: Using robust loss functions such as Huber loss or Cauchy loss can help in handling outliers. These loss functions are less sensitive to outliers and can reduce their impact on the model's performance.

Overall, handling outliers in MNIST involves a combination of data preprocessing and model training techniques. By detecting and removing outliers or making the model more robust to outliers, we can improve the model's performance and accuracy.

구글링 및 chatGPT 검색 결과를 통합한 anomaly detection에 대한 검색을 하였을 때의 방법들입니다만, 저는 신뢰도가 높은 방식인 robust loss function 중 휴버 손실 방식을 이용하여 시스템을 디자인할 것 같습니다.

Q4. 휴버 손실 함수는 평균제곱 오차(MSE), 즉 예측값과 실제값 사이의 평균 제곱 차이를 계산하여 이상치를 추출하는 함수와 절대 오차 손실 함수를 결합한 것입니다. MSE의 경우 이상치에 민감하기 떄문에 손실 함수의 값에 큰 영향을 미치게 하여, 손실률을 크게 만들기 때문에 이 부분을 보완하고자 절대오차 손실 함수와 같이 사용하는 함수입니다. 작동 방식은 오차가 작은 경우에는 MSE를, 오차가 큰 경우에는 절대 오차 손실 함수를 사용하는 방식입니다. 그렇기 떄문에 이상치에 덜 민감해져 유효 데이터의 손실을 최소화하면서 outlier를 잡을 수 있습니다. 이 방식을 아용하는 이유는 제 사진의 경우 결과값이

1/1 [==============================] - 0s 88ms/step

This image most likely belongs to 4 with a 23.20 percent confidence.

0 : 8.53%

1 : 8.53%

2 : 8.53%

3 : 8.53%

4 : 23.20%

5 : 8.53%

6 : 8.53%

7 : 8.53%

8 : 8.53%

9 : 8.53%

이렇게 나오고, 이 경우엔 절대 오차가 작동하게 될 것이고,

개 사진의 경우

1/1 [==============================] - 0s 10ms/step

This image most likely belongs to 0 with a 23.20 percent confidence.

0 : 23.20%

1 : 8.53%

2 : 8.53%

3 : 8.53%

4 : 8.53%

5 : 8.53%

6 : 8.53%

7 : 8.53%

8 : 8.53%

9 : 8.53%

이번 회차 결과값은 제 사진과 비슷한 확률로 값을 측정하게 나왔지만, 다른 경우엔 한 값에 11퍼, 다른 값에 17퍼 로 인식하게 되어, 이 경우엔 MSE 를 사용하게 될 것입니다.