# Medical image classification

By BEOM GON YU

# Contents

# introduction

## Data ( csv file and images)

| | | | | | |
|---|---|---|---|---|---|
| 30217 | c1e228e4-b7b4-432b-a735-36c48fdb806f | NaN | ... | NaN | 0 |
| 30218 | c1e3eb82-c55a-471f-a57f-fe1a823469da | NaN | ... | NaN | 0 |
| 30219 | c1e73a4e-7afe-4ec5-8af6-ce8315d7a2f2 | 666.0 | ... | 223.0 | 1 |
| 30220 | c1e73a4e-7afe-4ec5-8af6-ce8315d7a2f2 | 316.0 | ... | 273.0 | 1 |
| 30221 | c1ec14ff-f6d7-4b38-b0cb-fe07041cbdc8 | 609.0 | ... | 284.0 | 1 |
| 30222 | c1ec14ff-f6d7-4b38-b0cb-fe07041cbdc8 | 185.0 | ... | 379.0 | 1 |
| 30223 | c1edf42b-5958-47ff-a1e7-4f23d99583ba | NaN | ... | NaN | 0 |
| 30224 | c1f6b555-2eb1-4231-98f6-50a963976431 | NaN | ... | NaN | 0 |
| 30225 | c1f7889a-9ea9-4acb-b64c-b737c929599a | 570.0 | ... | 345.0 | 1 |
| 30226 | c1f7889a-9ea9-4acb-b64c-b737c929599a | 233.0 | ... | 356.0 | 1 |

(1024*1024*1) * 26684

# introduction

resnet
imagenet 2015에서 우승한 model.

vggnet에서 skip connection을 연결함으로써 Deep하게 NN을 쌓을 수 있게 됨.

단순함과 성능 모두 만족하며 현재까지도 여러 논문에서 base line으로 사용됨

resnet 50 사용함 (torchvision pretraining model을 사용)

training data로 Fine tuning (transfer learning)

# model 분석

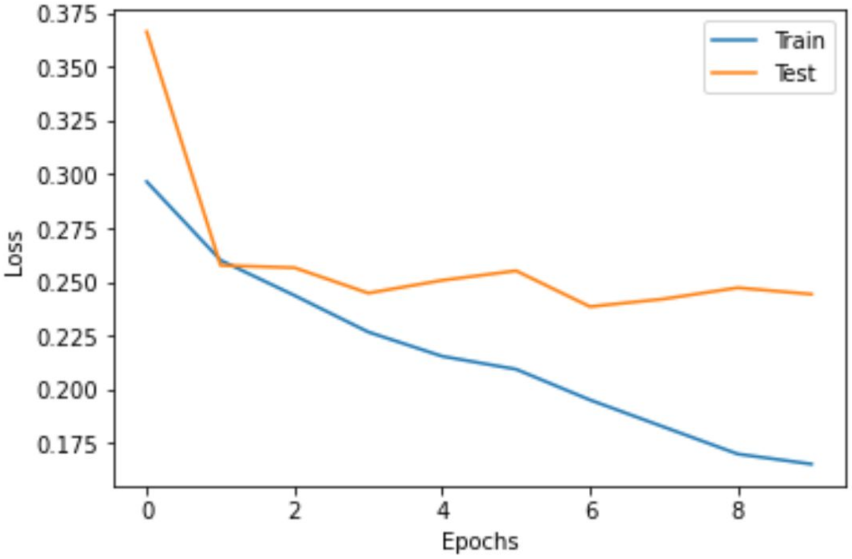
1. Data preprocessing (e.g., train/validation split, data loading, resize)
    a. train_df, test_df = train_test_split(df, test_size=0.1)
    b. MDataset (code 참조)

```
img_arr = pydicom.read_file(loc).pixel_array

img_arr = img_arr/img_arr.max()

img_arr = (255*img_arr).clip(0, 255).astype(np.uint8)

img_arr = Image.fromarray(img_arr).convert('RGB') # model expects 3 channel image
```

    c. transform = [transforms.Resize(224), transforms.RandomHorizontalFlip() , transforms.ToTensor()]

# model 분석

1. Train/validation loss analysis (or graph)
   a. F1 loss,



# model 분석

1. Performance metrics (e.g. f1-score, confusion matrix)

| Predicted Values | Actual Values: Positive (1) | Actual Values: Negative (0) |
|---|---|---|
| Positive (1) | TP | FP |
| Negative (0) | FN | TN |

Recall = TP / (TP + FN)
Precision = TP / (TP + FP)
Accuracy = (TP + TN) / (TP + FP + FN + TN)
F1-Score = 2 * (Recall * Precision) / (Recall + Precision)

```
ensenble
TP 345, FP 170, TN 1903, FN 251
Recall 0.5789 precision 0.6699, accuracy 0.8423, F1 score 0.6211
```

https://mhttps://minimin2.tistory.com/49inimin2.tistory.com/49

# etc

1. Imbalanced data handling
   a. class별 sample개수의 비율이 약 1:4로 고르지 않다.
      정상 class에 대해 학습이 더 많이 되는 부작용 발생,
      less class more samples, more class less samples
   b. class별 개수를 센 후 역수배의 비율로 sampling한다.

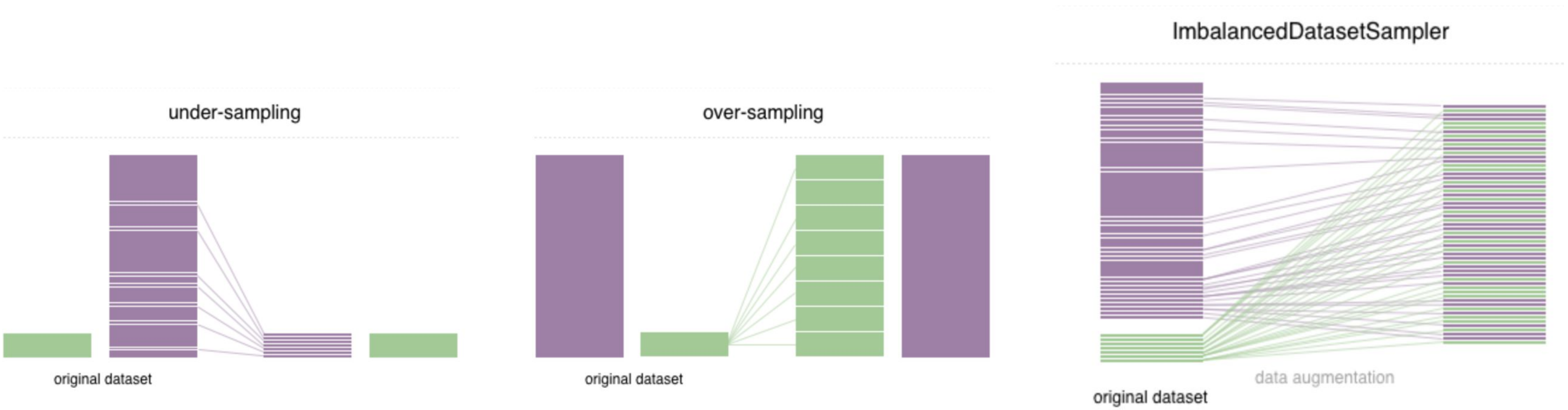
```
from torchsampler import ImbalancedDatasetSampler

train_loader = torch.utils.data.DataLoader(
    train_dataset,
    sampler=ImbalancedDatasetSampler(train_dataset),
    batch_size=args.batch_size,
    **kwargs
  )
```

https://github.com/ufoym/imbalanced-dataset-sampler

# Imbalanced data handling

## class balanced loss based on effective number of samples

- loss 계산시 가중치를 class당 sample 개수가 아닌 effective sample 개수 이용



# Imbalanced data handling

2000개의 sample로 테스트

f1 loss사용

data loader : 0.767

balanced dataloader : 0.8318

ensenble(⅓ *3) : 0.736 (0.7423, 0.7531, 0.7728)

0.7477 vs 0.7493 ( dataloader vs imbalanced dataloader), 전체 sample

# Imbalanced data handling

cross entropy vs focal loss vs f1 loss

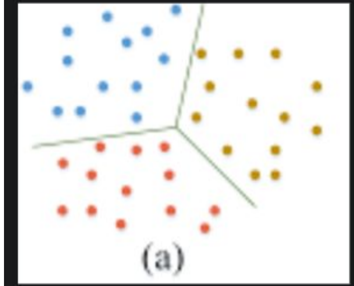
# etc

1. eXplainable AI
   a. model이 점점 커지지만 그 안의 동작은 이해하지 못하는 경우가 많다.
      단순히 classify를 잘하는 것에서 나아가 왜 이렇게 classify를 했는지 이해하고 싶다.
      (adversarial attack 등 방지)
   b. adversarial attack 방지 관련
      contrastive learning(Supervised Contrastive Learning)
      adversarial training(Adversarial Examples Improve Image Recognition)
   c. 각각의 CNN bock의 feature map visualization(ZFNET, 2013)
   d. 앙상블 with small model
      small model -> accuracy down, but more robustness
   e. IID -> OOD (in distribution -> out of distribution)
   f. 그 외 explanable AI에 대한 소개

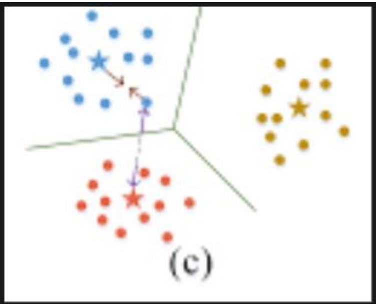
   https://theorydb.github.io/review/2020/06/09/review-book-xai/?fbclid=IwAR3XVLsCvyf5teiHLy9fpzg3jzFHHr6rIdhKniQVmZPJbjtunfUgN_USH4U

# etc

1. contrastive learning
   a. 같은 label끼리 모아주고 다른 label과는 멀어지게 학습함
      more robustness with small performance down



cross entropy

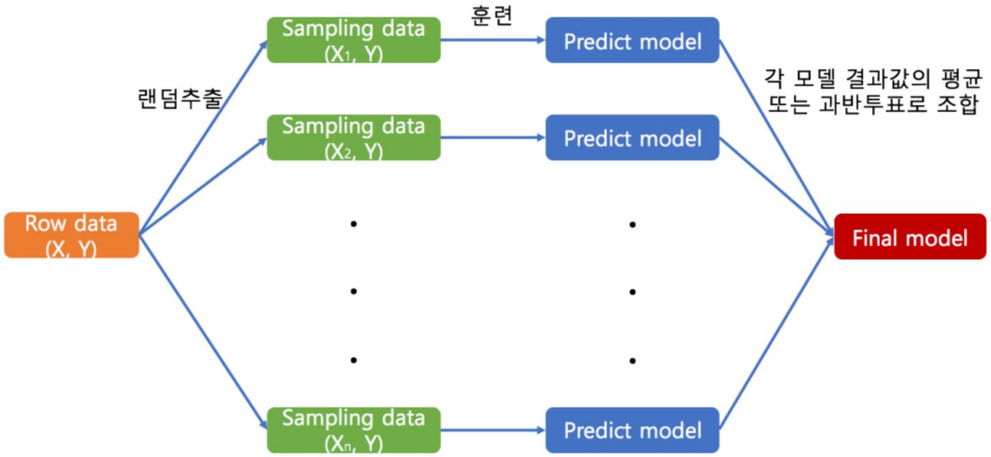


contrastive loss

# etc

## 1. Ensemble Method

training time을 줄이기 위해 전체 train data를 3등분하여 총 3개의 모델을 생성




```
ensenble : accuracy 0.8423, F1 score 0.6211
not ensenble : accuracy 0.8351, F1 score 0.7477
```

# etc

1. Vision Transformer
   자연처처리 또는 번역 등에서 transformer/bert가 큰 성공을 거두면서
   image쪽에서도 비슷한 시도들이 제안됨
   attention - non local network, senet  -> plug in 형태
   fully attentional - stand alone … -> cnn 대신 only attention으로만
   BERT/Transformer -> VideoBERT, Vision Transformer(vit) / DETR

# etc

1. Vision Transformer

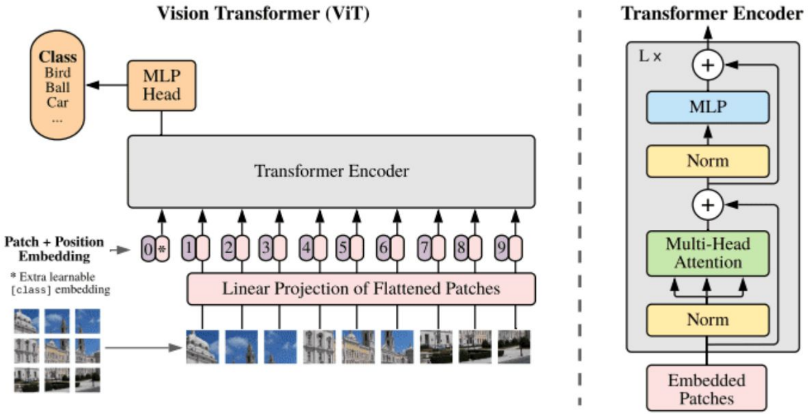



Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings to the resulting sequence of vectors, and feed the patches to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

sota(more than resnet)
no inductive bias (vs CNN, translation, loca
pretrain - fine tune
HW resource and training time
positional embedding ( 2d + relative)

# Vision Transformer

1. 최대 수억장의 label된 이미지로 pretrain
    a. pretrain된 model을 쓰고 fine tune하는 식으로 쓰자.
2. more memory, but less complexity
    a. memory critical하다면 use cnn,
       그러나 시간이 중요하다면 transformer를 쓰자.