Data Processing

Data Split: 80% training / 20% validation (using stratified split to ensure class balance)
Data Augmentation: Both resize to 224×224, use random horizontal flip (p=0.5), and normalization

Strong Augmentation

- For **unlabeled data** in FixMatch
- Shift/Scale/Rotate (shift=0.1, scale=0.2, rotate=20°)
- Gaussian Noise / Gaussian Blur (p=0.3)
- Strong Color Jitter (brightness/contrast/saturation=0.4, hue=0.2)
- Coarse Dropout (Cutout, 32×32 pixels)

Weak Augmentation

- For **labeled data**
- Color Jitter (brightness=0.2, contrast=0.2, saturation=0.2, hue=0.1)

Data Processing - Why Effective ?

JPEG Compression Augmentation

AI-generated images are more sensitive to compression artifacts; this augmentation improves model generalization to real-world scenarios

Cutout/CoarseDropout

Forces the model to learn local features and avoid over-reliance on global textures

Weak-Strong Augmentation Separation

Core strategy of FixMatch, utilizing consistency regularization to improve unlabeled data utilization

Model Architecture

Base Model Selection: **ConvNeXt-Base (fb_in1k)**

- Parameters: **88.6M**
- Pre-trained Weights: **ImageNet-1k**
- Larger Receptive Field: Better for capturing global anomalies in AI-generated images
- Superior Transfer Learning: Excellent performance on downstream tasks

```
model = timm.create_model(
    'convnext_base.fb_in1k'
    pretrained=True,
    num_classes=2,          #
    drop_rate=0.15          #
)
```

Training Strategy

Semi-Supervised Learning: FixMatch

- **Weak Augmentation** → Model prediction → Generate pseudo-labels (confidence > threshold)
- **Strong Augmentation** → Model prediction → Compute loss against pseudo-labels
- Only use high-confidence predictions as pseudo-labels (**threshold = 0.93**)
- **Unsupervised weight = 0.65**: ConvNeXt-Base is a large model requiring more data, hence higher unsupervised weight

EMA (Exponential Moving Average)

- teacher_params = 0.999 × teacher_params + 0.001 × student_params
- Maintains a moving average of model parameters for more stable pseudo-labels

```
model = train_model(
    model=model,
    train_loader=train_loader,
    val_loader=val_loader,
    unlabeled_loader=unlabeled_loader,
    epochs=10,
    lr=3e-5,
    device=device,
    use_semi_supervised=True,
    threshold=0.93,
    unsup_weight=0.65
)
```

Other Optimization

Regularization

- **Label smoothing (0.1)**: Prevents overconfident predictions
- **Weight Decay (5e-4)**: L2 regularization
- **Dropout (0.15)**: Random neuron dropout during training
- **Early Stopping (patience=5)**: Stops training if validation loss doesn't improve for 5 epochs

My Score – Public Leaderboard

43	112550198	0.99796	2	15h

Your Best Entry!
Your most recent submission scored 0.99796, which is an improvement over your previous score of 0.99733. Great job!

Tweet this