i-Mix

A DOMAIN-AGNOSTIC STRATEGY FOR CONTRASTIVE REPRESENTATION LEARNING

By MohamedElfatih MohamedElkhair

Abstract

- Contrastive learning approaches are well designed for vision domains only.

- Combine Contrastive learning approaches with Mixup.

- Improve the performance of contrastive learning approaches in across domains (image, speech, tabler)

What is contrastive learning?

Augmentation

Encoder

h_i

Maximize Agreement

h_i

SimCLR

What is Mixup?

0.4 x  + 0.6 x  = 

Cat: 1.0
Dog: 0.0

Cat: 0.0
Dog: 1.0

Cat: 0.4
Dog: 0.6

How Mixup is applied in Supervised setting?

$$\ell_{\text{Sup}}(x_i, y_i) = -\sum_{c=1}^{C} y_{i,c} \log \frac{\exp(w_c^\top f_i)}{\sum_{k=1}^{C} \exp(w_k^\top f_i)}$$

$$\ell_{\text{Sup}}^{\text{MixUp}}\big((x_i, y_i), (x_j, y_j); \lambda\big) = \ell_{\text{Sup}}(\lambda x_i + (1-\lambda)x_j, \lambda y_i + (1-\lambda)y_j)$$

Then....

How to apply it in Self-Supervised settings?

$$(x_i, x_j; \lambda) = \lambda x_i + (1-\lambda) x_j$$

$$\ell^{i\text{-Mix}}\big((x_i, v_i), (x_j, v_j); \mathcal{B}, \lambda\big) = \ell(\mathbf{Mix}(x_i, x_j; \lambda), \lambda v_i + (1-\lambda) v_j; \mathcal{B})$$

$$\mathbf{CutMix}(x_i, x_j; \lambda) = M_\lambda \odot x_i + (1-M_\lambda) \odot x_j$$

Contrastive Learning Approaches

- SimCLR
- Moco
- Byol

SimCLR

Loss for SimCLR

$$\ell_{\text{SimCLR}}(x_i;\mathcal{B}) = -\log \frac{\exp\left(s(f_i, f_{(N+i) \bmod 2N})/\tau\right)}{\sum_{k=1,k\neq i}^{2N} \exp\left(s(f_i, f_k)/\tau\right)}$$

After introducing virtual labels

$$\ell_{\text{N-pair}}(x_i, v_i;\mathcal{B}) = -\sum_{n=1}^{N} v_{i,n} \log \frac{\exp\left(s(f_i, \tilde{f}_n)/\tau\right)}{\sum_{k=1}^{N} \exp\left(s(f_i, \tilde{f}_k)/\tau\right)}$$

After applying Mixup

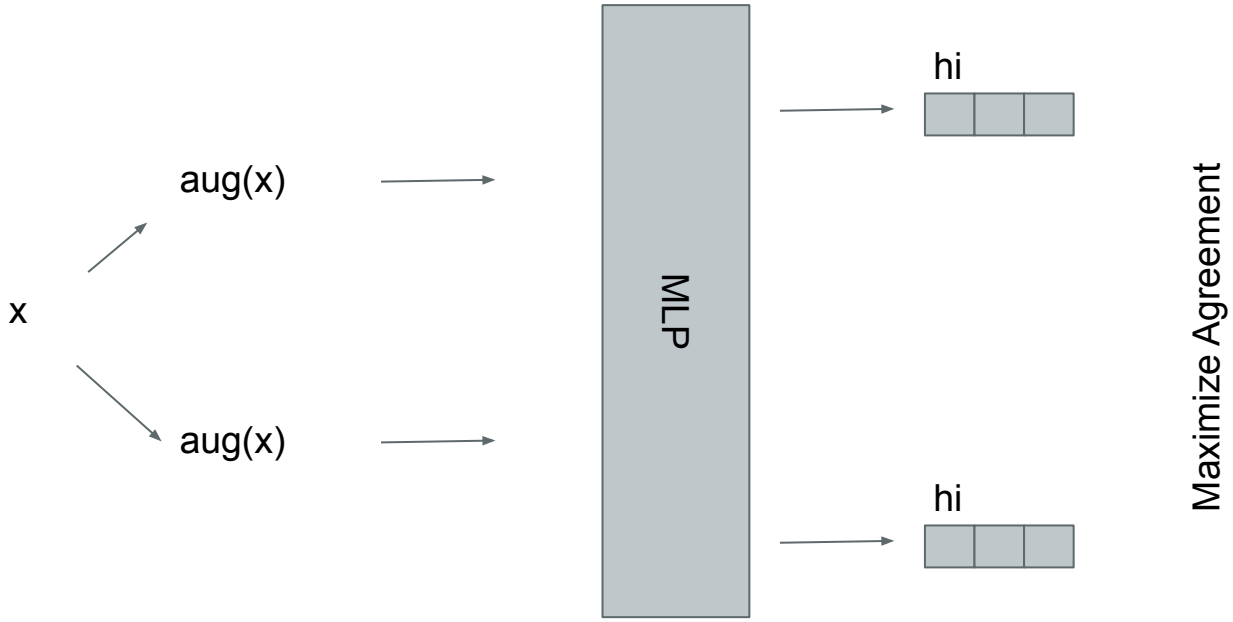
$$\ell_{\text{N-pair}}^{i\text{-Mix}}\big((x_i, v_i), (x_j, v_j);\mathcal{B}, \lambda\big) = \ell_{\text{N-pair}}(\lambda x_i + (1-\lambda)x_j, \lambda v_i + (1-\lambda)v_j;\mathcal{B})$$

After Linearization

$$\lambda\ell_{\text{N-pair}}(\lambda x_i + (1-\lambda)x_j, v_i;\mathcal{B}) + (1-\lambda)\ell_{\text{N-pair}}(\lambda x_i + (1-\lambda)x_j, v_j;\mathcal{B})$$

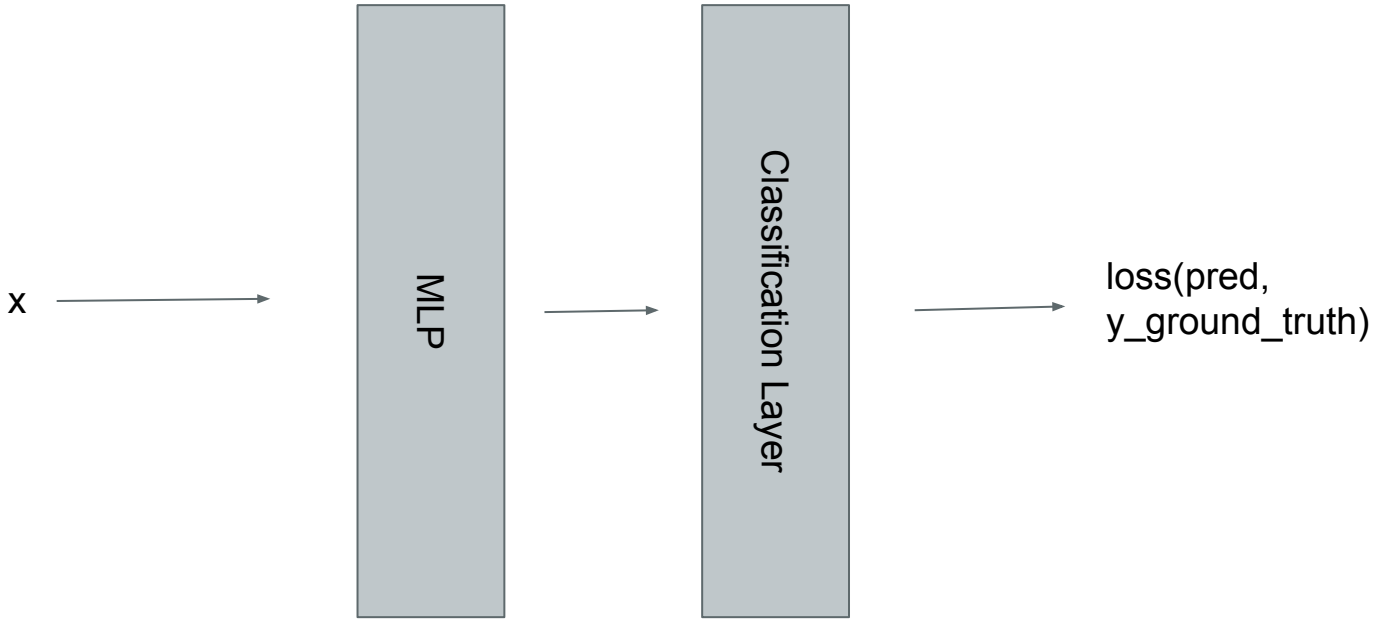
Implementation

Pre-train step



5 Layers MLP with batchnormalization

Fine-tune step



5 Layers MLP with batchnormalization

Npair pre-train code

```
def training_step(self, train_batch, batch_idx):
  t = 0.1
  x,y = train_batch

  aug_x, _ = covtype_aug(x.shape, x.float(), args.alpha, self.device)

  x_outs = self.forward(x.float())
  aug_x_outs = self.forward(aug_x.float())

  norm_aug_x = F.normalize(aug_x_outs, dim=-1)
  norm_x = F.normalize(x_outs, dim=-1)
  logits = norm_x @ norm_aug_x.T / t

  labels = torch.tensor(range(len(x))).to(self.device)

  loss = self.cross_entropy_loss(logits, labels)

  self.log('train loss', loss)
  self.saved_fts.append(x_outs)
  self.saved_labels.append(y)
  return loss
```

imix-Npair pre-train code

```
def training_step(self, train_batch, batch_idx):
    t = 0.1
    x,y = train_batch

    aug_x, _ = covtype_aug(x.shape, x.float(), args.alpha, self.device

    # Code added
    lam = np.random.beta(args.beta, args.beta)
    randidx = np.random.permutation(len(x))
    x = lam * x + (1 - lam) * x[randidx]

    x_outs = self.forward(x.float())
    aug_x_outs = self.forward(aug_x.float())

    norm_aug_x = F.normalize(aug_x_outs, dim=-1)
    norm_x = F.normalize(x_outs, dim=-1)

    logits = norm_x @ norm_aug_x.T / t


    labels = torch.tensor(range(len(x))).to(self.device)
    labels_perm = randidx

    loss = lam * self.cross_entropy_loss(logits, labels) \
    + (1 - lam) * self.cross_entropy_loss(logits, labels_perm)

    self.log('train loss', loss)

    self.saved_fts.append(x_outs)
    self.saved_labels.append(y)

    return loss
```

Linearized form of the mixup loss

Experiment

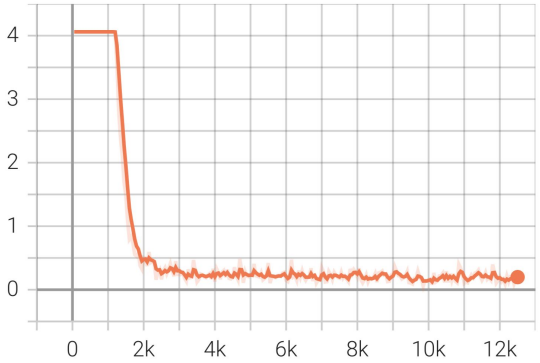
Setup

- 500/150 (pretrain/finetune) epochs
- 512 Batch size
- Learning rate of .125
- Linear warmup for 10 epochs a followed consine Annealing
- Covertype tabular data 15k training and 566k for testing

Results

NPair

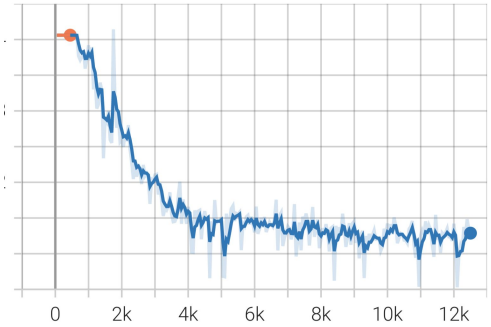
pretrain_loss



Test ACC 67.7

NPair + imix

pretrain_loss



Test ACC 74.8

Comparing with the paper

Test Accuracy

Implementation		Paper	
Npair	+ imix	Npair	+ imix
67.7	74.8	68.5	72.1

Future Works

- Implementing BYOL, MOCO
- Experimenting in Speech commands and CIFAR
- Organizing the code

QUESTIONS?