Mixup: Beyond Empirical Risk Minimization

Radek Bartyzal

GLAMI AI

27. 10. 2020

Motivation

- Empirical Risk Minimization = minimize errors on samples from dataset
- Data Augmentation = create new samples "around" the existing samples = capture more of the "true" distribution
- $\implies$ regularization $\implies$ better generalization

Downsides of classic Data Augmentation:

- dataset dependent $\implies$ expert knowledge
- assumes examples in the vicinity share the same class
- $\implies$ does not model vicinity relation across examples of different classes

Mixup

- data agnostic generation of new training samples
- linear interpolation

$$\tilde{x} = \lambda x_i + (1 - \lambda) x_j, \qquad \text{where } x_i, x_j \text{ are raw input vectors}$$
$$\tilde{y} = \lambda y_i + (1 - \lambda) y_j, \qquad \text{where } y_i, y_j \text{ are one-hot label encodings}$$

Figure: Generate new training samples. $\lambda \in [0, 1]$

Mixup: Easy implementation

```
# y1, y2 should be one-hot vectors
for (x1, y1), (x2, y2) in zip(loader1, loader2):
    lam = numpy.random.beta(alpha, alpha)
    x = Variable(lam * x1 + (1. - lam) * x2)
    y = Variable(lam * y1 + (1. - lam) * y2)
    optimizer.zero_grad()
    loss(net(x), y).backward()
    optimizer.step()
```

(a) One epoch of *mixup* training in PyTorch.

Mixup: Smoother decision boundary

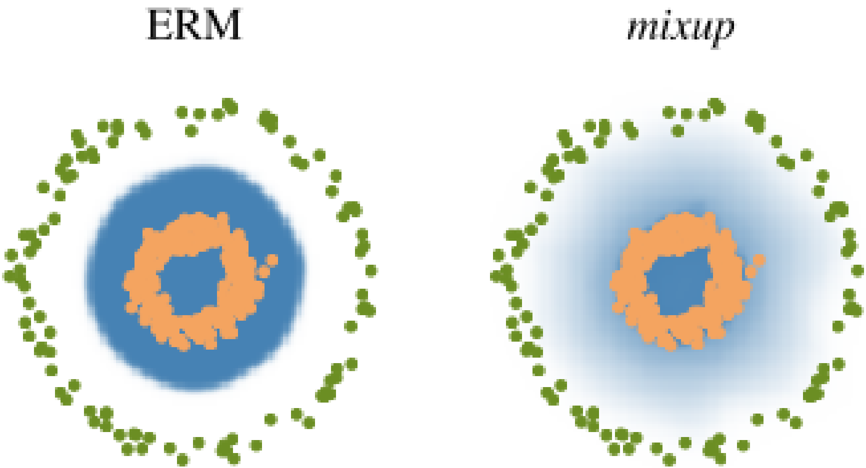


Figure: Effect of mixup ($\alpha = 1$) on a toy problem. Green: Class 0. Orange: Class 1. Blue shading indicates $p(y = 1|x)$.

Mixup: Vicinal Risk Minimization (VRM)

VRM approximation of true distribution P:

$$P_\nu(\tilde{x}, \tilde{y}) = \frac{1}{n} \sum_{i=1}^{n} \nu(\tilde{x}, \tilde{y} | x_i, y_i),$$

- ν is a vicinity distribution that measures the probability of finding the virtual feature-target pair $(\tilde{x}, \tilde{y})$ in the vicinity of the training feature-target pair (x_i, y_i).
- to train we sample the vicinal distribution to construct a dataset D_ν and minimize the empirical vicinal risk: $R_\nu(f)$

$$R_\nu(f) = \frac{1}{m} \sum_{i=1}^{m} \ell(f(\tilde{x}_i), \tilde{y}_i).$$

Mixup: Vicinal Risk Minimization (VRM)

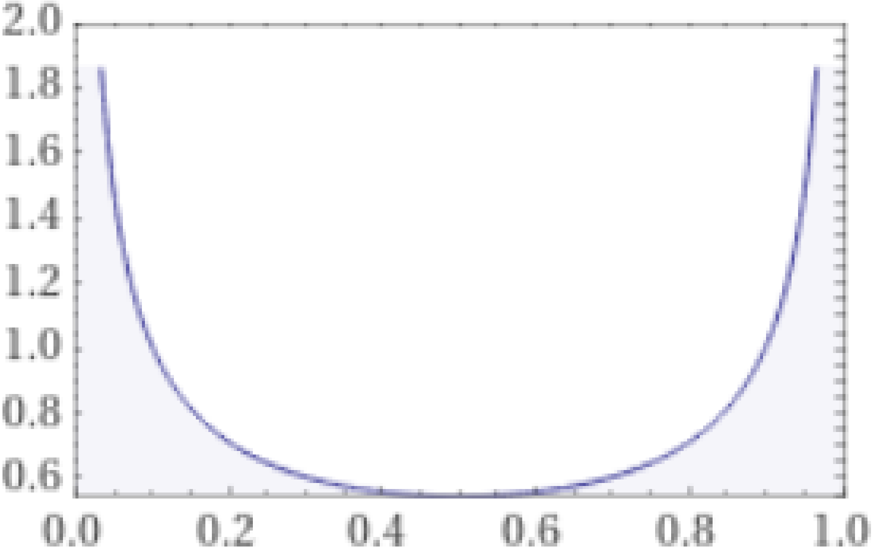
Contribution: Generic vicinal distribution, called mixup:

$$\mu(\tilde{x}, \tilde{y}|x_i, y_i) = \frac{1}{n}\sum_{j}^{n} \mathbb{E}_{\lambda}\left[\delta(\tilde{x} = \lambda \cdot x_i + (1-\lambda) \cdot x_j, \tilde{y} = \lambda \cdot y_i + (1-\lambda) \cdot y_j)\right]$$

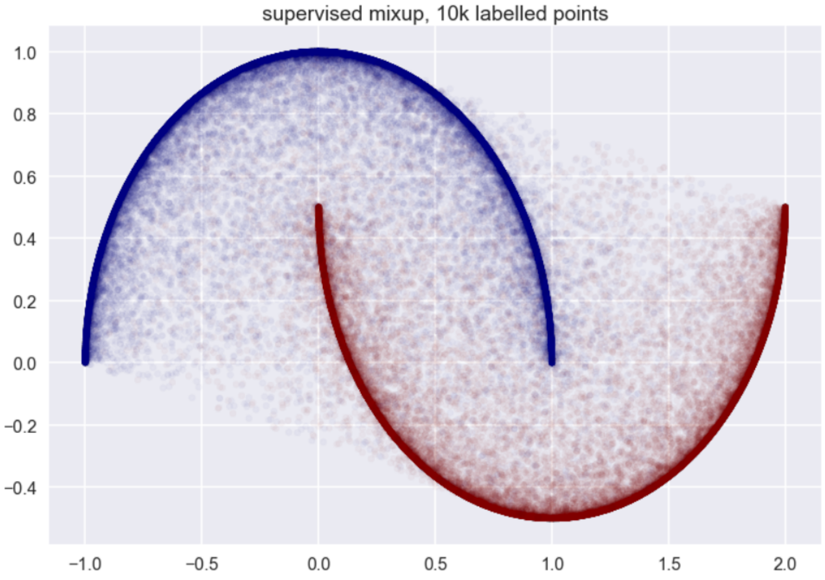
- where $\lambda \sim \mathit{Beta}(\alpha, \alpha)$, for $\alpha \in (0, \inf)$. In a nutshell, sampling from the mixup vicinal distribution produces virtual feature-target vectors:

$$\tilde{x} = \lambda x_i + (1-\lambda)x_j,$$
$$\tilde{y} = \lambda y_i + (1-\lambda)y_j,$$

Beta distribution ($\alpha = 0.4, \beta = 0.4$)



Samples generated by Mixup on Two Moons Dataset



Mixup: ImageNet results

Model	Method	Epochs	Top-1 Error	Top-5 Error
ResNet-50	ERM (Goyal et al., 2017)	90	23.5	-
	mixup $\alpha = 0.2$	90	**23.3**	**6.6**
ResNet-101	ERM (Goyal et al., 2017)	90	22.1	-
	mixup $\alpha = 0.2$	90	**21.5**	**5.6**
ResNeXt-101 32*4d	ERM (Xie et al., 2016)	100	21.2	-
	ERM	90	21.2	5.6
	mixup $\alpha = 0.4$	90	**20.7**	**5.3**
ResNeXt-101 64*4d	ERM (Xie et al., 2016)	100	20.4	5.3
	mixup $\alpha = 0.4$	90	**19.8**	**4.9**
ResNet-50	ERM	200	23.6	7.0
	mixup $\alpha = 0.2$	200	**22.1**	**6.1**
ResNet-101	ERM	200	22.0	6.1
	mixup $\alpha = 0.2$	200	**20.8**	**5.4**
ResNeXt-101 32*4d	ERM	200	21.3	5.9
	mixup $\alpha = 0.4$	200	**20.1**	**5.0**

Table 1: Validation errors for ERM and *mixup* on the development set of ImageNet-2012.

Mixup: ImageNet results

- trained with standard augmentations: scale and aspect ratio distortions, random crops, and horizontal flip
- $\alpha \in [0.1, 0.4]$ leads to improved performance over ERM
- large $\alpha \implies$ underfitting
- models with higher capacities and/or longer training runs benefit more from mixup

Mixup: CIFAR results

Dataset	Model	ERM	*mixup*
CIFAR-10	PreAct ResNet-18	5.6	**4.2**
	WideResNet-28-10	3.8	**2.7**
	DenseNet-BC-190	3.7	**2.7**
CIFAR-100	PreAct ResNet-18	25.6	**21.1**
	WideResNet-28-10	19.4	**17.5**
	DenseNet-BC-190	19.0	**16.8**

(a) Test errors for the CIFAR experiments.

Mixup: Further experiments

Mixup helps with:

- speech commands recognition using VGG
- tabular data: UCI datasets with 2-layer nets trained by Adam
- robustness against adversarial attacks
- stabilisation of GAN training

Mixup: Conclusion

Implementation details:

- mix 1 batch with itself just shuffled
- sample λ for each created example
- to remove duplicates: $\lambda = max(\lambda, 1 - \lambda) \implies \lambda \in [0.5, 1.0]$
- if using mixup - lower weight decay

Further results:

- SMOTE (interpolate only between same-class samples) performs worse
- combining more than 2 samples does not help

CutMix: Cut and paste + mix labels acc. to area pasted

	ResNet-50	Mixup [48]	Cutout [3]	CutMix
Image				
Label	Dog 1.0	Dog 0.5 Cat 0.5	Dog 1.0	Dog 0.6 Cat 0.4
ImageNet Cls (%)	76.3 (+0.0)	77.4 (+1.1)	77.1 (+0.8)	**78.6** **(+2.3)**
ImageNet Loc (%)	46.3 (+0.0)	45.8 (-0.5)	46.7 (+0.4)	**47.3** **(+1.0)**
Pascal VOC Det (mAP)	75.6 (+0.0)	73.9 (-1.7)	75.1 (-0.5)	**76.7** **(+1.1)**

Sources

1. Zhang, Hongyi, et al. "mixup: Beyond empirical risk minimization." arXiv preprint arXiv:1710.09412 (2017). https://arxiv.org/abs/1710.09412

2. FastAI implementation comments. https://forums.fast.ai/t/mixup-data-augmentation/22764

3. INFERENCE blog (two moons analysis). https://www.inference.vc/mixup-data-dependent-data-augmentation/

4. Yun, Sangdoo, et al. "Cutmix: Regularization strategy to train strong classifiers with localizable features." Proceedings of the IEEE International Conference on Computer Vision. 2019. https://arxiv.org/abs/1905.04899