Regularization, Data Augmentation and Self-supervised Learning

Efficient Deep Learning - Session 4

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

2022

Course organisation

Sessions

1. Introduction/Refresher on Deep Learning
2. Quantization,
3. Pruning,
4. Regularization, Data Augmentation,
5. Factorization,
6. Distillation,
7. Embedded Software and Hardware for DL,
8. Final session.

Course organisation

Sessions

1. Introduction/Refresher on Deep Learning
2. Quantization,
3. Pruning,
4. Regularization, Data Augmentation,
5. Factorization,
6. Distillation,
7. Embedded Software and Hardware for DL,
8. Final session.

Why this session ?

Regularization

Constrain the training for faster convergence and better generalization.

Data Augmentation (DA)

Help generalization by sampling training examples from a larger distribution using randomized transforms.

Self-supervised Learning (SSL)

Exploit DA and regularization tricks for learning representations, without labels

Significance

- In some (most?) cases, DA regularizes training and is needed.
- Large networks can't be trained without regularization.

Why this session ?

Regularization

Constrain the training for faster convergence and better generalization.

Data Augmentation (DA)

Help generalization by sampling training examples from a larger distribution using randomized transforms.

Self-supervised Learning (SSL)

Exploit DA and regularization tricks for learning representations, without labels

Significance

- In some (most?) cases, DA regularizes training and is needed.
- Large networks can't be trained without regularization.

Why this session ?

Regularization

Constrain the training for faster convergence and better generalization.

Data Augmentation (DA)

Help generalization by sampling training examples from a larger distribution using randomized transforms.

Self-supervised Learning (SSL)

Exploit DA and regularization tricks for learning representations, without labels

Significance

- In some (most?) cases, DA regularizes training and is needed.
- Large networks can't be trained without regularization.

Why this session ?

Regularization

Constrain the training for faster convergence and better generalization.

Data Augmentation (DA)

Help generalization by sampling training examples from a larger distribution using randomized transforms.

Self-supervised Learning (SSL)

Exploit DA and regularization tricks for learning representations, without labels

Significance

- In some (most?) cases, DA regularizes training and is needed.
- Large networks can't be trained without regularization.

Regularization

Weight Decay

An old idea (Krogh and Herz 1991): ℓ_2 penatly term is added to the loss, limits the growth of model weights.

Has been shown to increase generalization and suppresses irrelevant model weights.

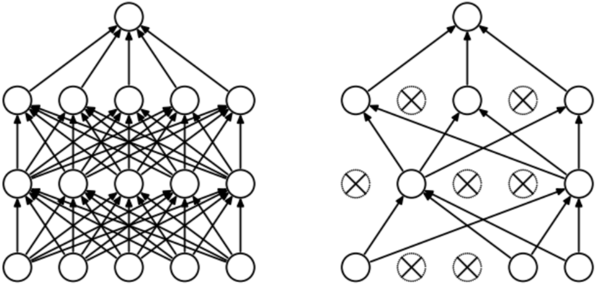
Ressources :

- https://proceedings.neurips.cc/paper/1991/file/8eefcfdf5990e441f0fb6f3fad709e21-Paper.pdf
- https://ja.d2l.ai/chapter_deep-learning-basics/weight-decay.html
- Readily available in pytorch (optimizer options)

Regularization

Dropout

Randomly "drops" some units during training with a certain probability.



(a) Standard Neural Net (b) After applying dropout.

- Was introduced to train very large networks
- Can prevent overfitting
- Adds hyperparameters : where to drop ? How often ?

https://www.jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

Regularization

Batch Normalization (Ioffe & Szegedy, 2015)

Normalize feature distributions to the standard distribution by learning batch statistics.

- Consider a batch X
- Calculate $m = E(X)$ and $\sigma = Var(X)$
- Compute $\hat{X} = \frac{X-m}{\sigma}$
- m and σ are continuously updated across batches using running statistics

Notes

- Has been shown to accelerate training, increase generalization
- Can remove the need for DropOut
- Should be included by default after convolutions

`http://proceedings.mlr.press/v37/ioffe15.pdf`

Data Augmentation using image transformations

Translations, rotations, Scaling, Shifting in RGB, Crops, ...

Image from Albumentations `https://albumentations.ai/docs/examples/pytorch_classification/`

Mixup, Cutout and Cutmix

Mixup

For a network F trained using Cross Entropy (CE),

- Sample x_i, x_j from the training data, associated to labels y_i, y_j.
- Defined mixed up data samples as $\tilde{x} = x_i + (1 - \lambda)x_j$
- $loss = \lambda CE(F(\tilde{x}), y_i) + (1 - \lambda)CE(F(\tilde{x}), y_j)$, where $\lambda \in [0, 1]$
- Train with backprop

Notes

- Has been shown to regularize training and achieves better generalization.
- Should be included most of the time when training classification networks !
- See Lab4.md for a proposed implementation

`https://arxiv.org/pdf/1710.09412.pdf`

Mixup, Cutout and Cutmix

	ResNet-50	Mixup [47]	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)**

Table 1: Overview of the results of Mixup, Cutout, and our CutMix on ImageNet classification, ImageNet localization, and Pascal VOC 07 detection (transfer learning with SSD [23] finetuning) tasks. Note that CutMix significantly improves the performance on various tasks.

https://openaccess.thecvf.com/content_ICCV_2019/papers/Yun_CutMix_Regularization_Strategy_to_Train_Strong_Classifiers_With_Localizable_Features_ICCV_2019_paper.pdf

Application to Self supervised Learning

Self-Supervised Learning

Learn representations of input samples without labels or annotations

How ?

Train encoders (e.g. ResNet) on pre-text tasks:

- Self-Prediction
- Contrastive Learning

Trained encoders are expected to learn general features that generalize to supervised tasks.

Application to Self supervised Learning

Self-Supervised Learning

Learn representations of input samples without labels or annotations

How ?

Train encoders (e.g. ResNet) on pre-text tasks:

- Self-Prediction
- Contrastive Learning

Trained encoders are expected to learn general features that generalize to supervised tasks.

Application to Self supervised Learning

Contrastive Learning : SimCLR.

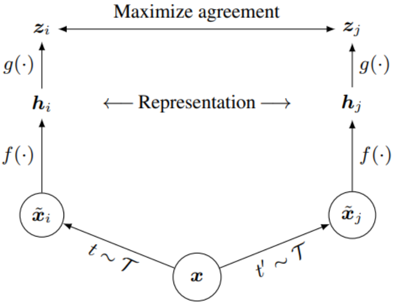


Figure 2. A simple framework for contrastive learning of visual representations. Two separate data augmentation operators are sampled from the same family of augmentations ($t \sim \mathcal{T}$ and $t' \sim \mathcal{T}$) and applied to each data example to obtain two correlated views. A base encoder network $f(\cdot)$ and a projection head $g(\cdot)$ are trained to maximize agreement using a contrastive loss. After training is completed, we throw away the projection head $g(\cdot)$ and use encoder $f(\cdot)$ and representation $\boldsymbol{h}$ for downstream tasks.

https://arxiv.org/pdf/2002.05709.pdf