

# Regularization, Data Augmentation and Self-Supervised Learning

Efficient Deep Learning - Session 2



2023



### └ Course organisation

#### Sessions

- 1 Intro Deep Learning,
- 2 Data Augmentation and Self Supervised Learning,
- 3 Quantization,
- 4 Pruning,
- 5 Factorization,
- 6 Distillation,
- 7 Embedded Software and Hardware for DL,
- 8 Presentations for challenge.

### Sessions

- 1 Intro Deep Learning,
- 2 Data Augmentation and Self Supervised Learning,
- 3 Quantization,
- 4 Pruning,
- 5 Factorization,
- 6 Distillation,
- 7 Embedded Software and Hardware for DL,
- 8 Presentations for challenge.

### └ Course organisation

- Sessions
- 1 Intro Deep Learning,
  - 2 Data Augmentation and Self Supervised Learning,
  - 3 Quantization,
  - 4 Pruning,
  - 5 Factorization,
  - 6 Distillation,
  - 7 Embedded Software and Hardware for DL,
  - 8 Presentations for challenge.

### Sessions

- 1 Intro Deep Learning,
- 2 Data Augmentation and Self Supervised Learning,
- 3 Quantization,
- 4 Pruning,
- 5 Factorization,
- 6 Distillation,
- 7 Embedded Software and Hardware for DL,
- 8 Presentations for challenge.

# 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.

2024-02-09

## Regularization, DA and SSL

└ Why this session ?

Regularization prevents overfitting in neural networks, thus improve the predictions on data outside the training set

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.

2024-02-09

## Regularization, DA and SSL

└ Why this session ?

DA is a technique that increases the training set by creating new data points based on the original data during training

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, DA and SSL

└ Why this session ?

SSL in deep learning is a technique for learning data representation without the use of label for transfer learning or fine-tuning

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.

2024-02-09

# 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.

2024-02-09 Regularization, DA and SSL

└ Why this 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.

## Weight Decay

An old idea (Krogh and Herz 1991):  $\ell_2$  penalty 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/8eefcfd5990e441f0fb6f3fad709e21-Paper.pdf>
- [https://ja.d2l.ai/chapter\\_deep-learning-basics/weight-decay.html](https://ja.d2l.ai/chapter_deep-learning-basics/weight-decay.html)
- Readily available in pytorch (optimizer options)

### Regularization

Weight decay involves adding a term to the objective function that is proportional to the sum of the squares of the weights

#### Weight Decay

An old idea (Krogh and Herz 1991):  $\ell_2$  penalty term is added to the loss, limits the growth of model weights.

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

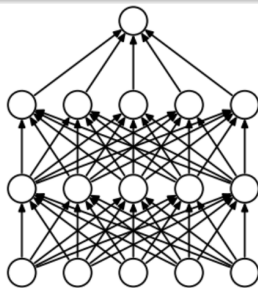
Ressources :

- <https://proceedings.neurips.cc/paper/1991/file/8eefcfd5990e441f0fb6f3fad709e21-Paper.pdf>
- [https://ja.d2l.ai/chapter\\_deep-learning-basics/weight-decay.html](https://ja.d2l.ai/chapter_deep-learning-basics/weight-decay.html)
- Readily available in pytorch (optimizer options)

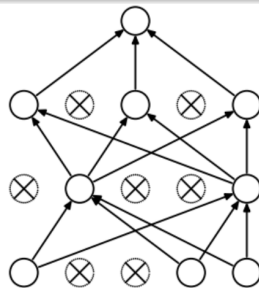


## Dropout

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



(a) Standard Neural Net

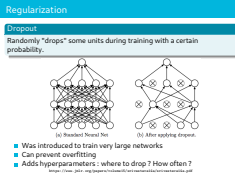


(b) After applying dropout.

- 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



Dropout is a technique used only for training so that neurons will not learn redundant information of data, as well as not relying on some specific features as they might be randomly dropped out

## 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  $\sigma = \text{Var}(X)$
- Compute  $\hat{X} = \frac{X-m}{\sigma} * \gamma + \beta$
- $m$  and  $\sigma$  are continuously updated across batches using running statistics, and  $\gamma$  and  $\beta$  are learnable parameters (by default set to 1 and 0, respectively)

## Notes

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

## Regularization

- Consider a batch  $X$
- Calculate  $m = E(X)$  and  $\sigma = \text{Var}(X)$
- Compute  $\hat{X} = \frac{X-m}{\sigma} * \gamma + \beta$
- $m$  and  $\sigma$  are continuously updated across batches using running statistics, and  $\gamma$  and  $\beta$  are learnable parameters (by default set to 1 and 0, respectively)

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

Batch Normalization serves to speed up convergence, and also allows the use of higher learning rates without risk of divergence

# Data Augmentation using image transformations

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



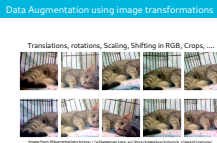
Image from Albumentations [https://albumentations.ai/docs/examples/pytorch\\_classification/](https://albumentations.ai/docs/examples/pytorch_classification/)

2024-02-09

Regularization, DA and SSL

└ Data Augmentation using image transformations

DA increases the amount of data that the model sees during training, it is only applied on the training set. Note that it's stochastic meaning that the model sees different augmented versions of the images in each epoch



# 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  $\tilde{x} = \lambda x_i + (1 - \lambda)x_j$
- $loss = \lambda CE(F(\tilde{x}), y_i) + (1 - \lambda)CE(F(\tilde{x}), y_j)$ , 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>

2024-02-09

## Regularization, DA and SSL

### └ 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  $\tilde{x} = \lambda x_i + (1 - \lambda)x_j$
- $loss = \lambda CE(F(\tilde{x}), y_i) + (1 - \lambda)CE(F(\tilde{x}), y_j)$ , 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 forces simple linear behavior in-between training samples, it is also robust against noisy labels

# Mixup, Cutout and Cutmix





| Image                   | ResNet-50   | Mixup [47]  | Cutout [3]  | CutMix  |
|-------------------------|---|---|---|---|
|                         |  |  |  |  |
| Label                   | Dog 1.0   | Dog 0.5<br>Cat 0.5  | Dog 1.0   | Dog 0.6<br>Cat 0.4  |
| ImageNet<br>Cls (%)     | 76.3<br>(+0.0)  | 77.4<br>(+1.1)  | 77.1<br>(+0.8)  | <b>78.6</b><br>(+2.3)   |
| ImageNet<br>Loc (%)     | 46.3<br>(+0.0)  | 45.8<br>(-0.5)  | 46.7<br>(+0.4)  | <b>47.3</b><br>(+1.0)   |
| Pascal VOC<br>Det (mAP) | 75.6<br>(+0.0)  | 73.9<br>(-1.7)  | 75.1<br>(-0.5)  | <b>76.7</b><br>(+1.1)   |

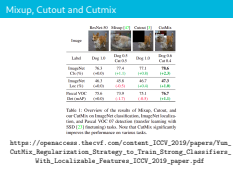
Table 1: Overview of the results of Mixup, Cutout, and our CutMix on ImageNet classification, ImageNet localization, and Pascal VOC 07 detection (transfer learning with SSD [23] finetuning) tasks. Note that CutMix significantly improves the performance on various tasks.

[https://openaccess.thecvf.com/content\\_ICCV\\_2019/papers/Yun\\_CutMix\\_Regularization\\_Strategy\\_to\\_Train\\_Strong\\_Classifiers\\_With\\_Localizable\\_Features\\_ICCV\\_2019\\_paper.pdf](https://openaccess.thecvf.com/content_ICCV_2019/papers/Yun_CutMix_Regularization_Strategy_to_Train_Strong_Classifiers_With_Localizable_Features_ICCV_2019_paper.pdf)

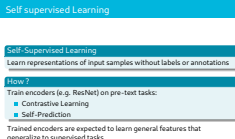
2024-02-09 Regularization, DA and SSL

└ Mixup, Cutout and Cutmix

Mixup boosts performance for classification (CLS) problems but degrades results for localization (LOCC) and objet detection (OO) problems. Cutout improves results for both CLS and LOC tasks but degrades performance for OO. CutMix improves results for all three tasks



## 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:

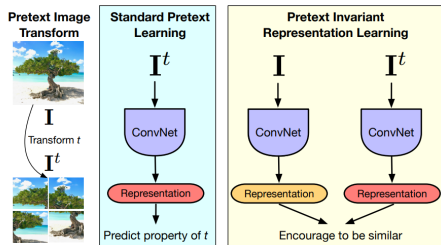
- Contrastive Learning
- Self-Prediction

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

SSL consists in learning a model that outputs useful representations of data without the use of label. This model can be used for transfer learning to other tasks. The model is trained in a supervised way with "labels" created from the data itself

# Self supervised Learning

## Contrastive Learning : Pretext-Invariant Representations Learning (PIRL)



<https://arxiv.org/pdf/1912.01991.pdf>

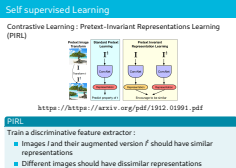
### PIRL

Train a discriminative feature extractor :

- Images  $I$  and their augmented version  $I^t$  should have similar representations
- Different images should have dissimilar representations

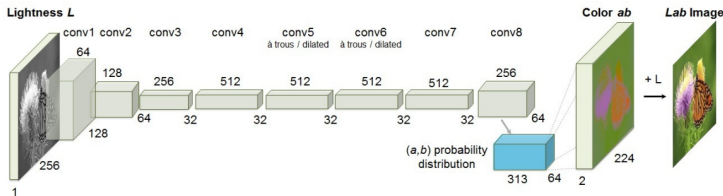
2024-02-09 Regularization, DA and SSL

└ Self supervised Learning



Contrastive learning is the state-of-the-art method for learning good representation of data, it consists in learning discriminative features by measuring similarities and dissimilarities between samples, it relies heavily on data augmentation

## Self-Prediction : Colorful Image Colorization



<https://arxiv.org/pdf/1603.08511.pdf>

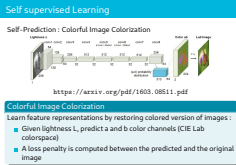
## Colorful Image Colorization

Learn feature representations by restoring colored version of images :

- Given lightness  $L$ , predict  $a$  and  $b$  color channels (CIE Lab colorspace)
- A loss penalty is computed between the predicted and the original image

## Self supervised Learning

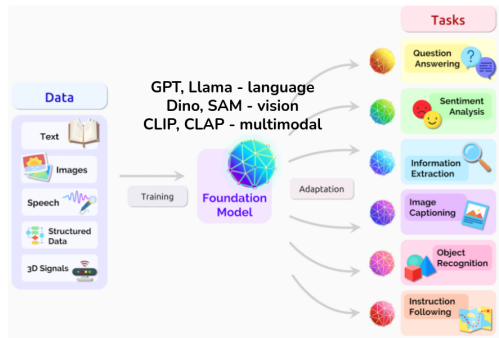
Colorizing images is an example of self-prediction tasks that learns representation of data by predicting a part of the data (colors in this case)





# Self supervised Learning

## SSL to pretrain foundation models



<https://arxiv.org/pdf/2108.07258.pdf>

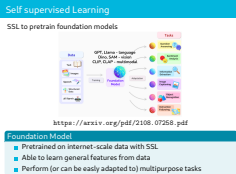
## Foundation Model

- Pretrained on internet-scale data with SSL
- Able to learn general features from data
- Perform (or can be easily adapted to) multipurpose tasks

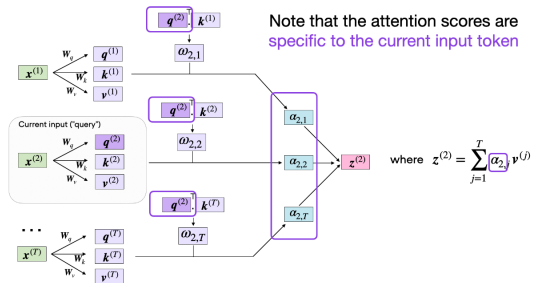
2024-02-09

Regularization, DA and SSL

└ Self supervised Learning



# Self-Attention



## Self-Attention in foundation models

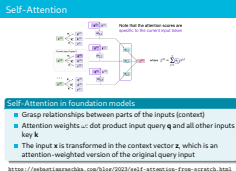
- Grasp relationships between parts of the inputs (context)
- Attention weights  $\omega$ : dot product input query  $\mathbf{q}$  and all other inputs key  $\mathbf{k}$
- The input  $\mathbf{x}$  is transformed in the context vector  $\mathbf{z}$ , which is an attention-weighted version of the original query input

<https://sebastianraschka.com/blog/2023/self-attention-from-scratch.html>

2024-02-09

Regularization, DA and SSL

└ Self-Attention

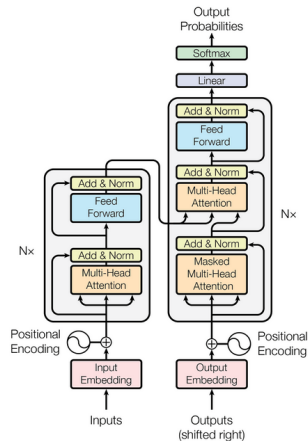


## Self-Attention

### Self-Attention and Transformers

- Self-Attention is found in the basic architecture of foundation models: **Transformers**
- No convolutions, inputs are transformed taking in account attention weights
- Best generalization in many domains, but need large scale data

<https://arxiv.org/pdf/1706.03762.pdf>



- Self-Attention is found in the basic architecture of foundation models: **Transformers**
- No convolutions, inputs are transformed taking in account attention weights
- Best generalization in many domains, but need large scale data

<https://arxiv.org/pdf/1706.03762.pdf>

