

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.

2025-02-12

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.

2025-02-12

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 <ul style="list-style-type: none">■ 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.

2025-02-12

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 <ul style="list-style-type: none">■ 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.

2025-02-12

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): ℓ_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
- 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): ℓ_2 penalty term is added to the loss, limits the growth of model weights.

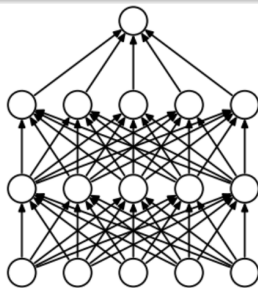
Has been shown to increase generalization and suppresses irrelevant model weights.

Ressources :

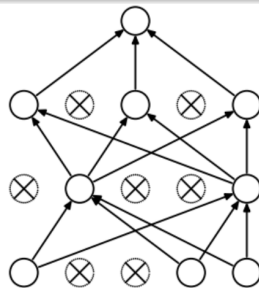
- <https://proceedings.neurips.cc/paper/1991/file/8eefcfd5990e441f0fb6f3fad709e21-Paper.pdf>
- https://ja.d2l.ai/chapter_deep-learning-basics/weight-decay.html
- Readily available in pytorch (optimizer options)

Dropout

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



(a) Standard Neural Net



(b) After applying dropout.

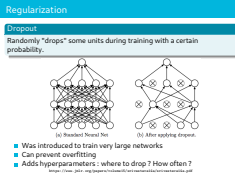
- Was introduced to train very large networks
- Can prevent overfitting
- Adds hyperparameters : where to drop ? How often ?

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

2025-02-12

Regularization, DA and SSL

└ 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 σ are continuously updated across batches using running statistics, and γ and β 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 σ are continuously updated across batches using running statistics, and γ and β 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,



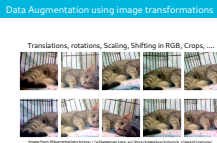
Image from Albumentations https://albumentations.ai/docs/examples/pytorch_classification/

2025-02-12

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>

2025-02-12

Regularization, DA and SSL

└ Mixup, Cutout and Cutmix

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





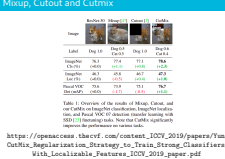
Image	ResNet-50	Mixup [47]	Cutout [3]	CutMix
				
Label	Dog 1.0	Dog 0.5 Cat 0.5	Dog 1.0	Dog 0.6 Cat 0.4
ImageNet Cls (%)	76.3 (+0.0)	77.4 (+1.1)	77.1 (+0.8)	78.6 (+2.3)
ImageNet Loc (%)	46.3 (+0.0)	45.8 (-0.5)	46.7 (+0.4)	47.3 (+1.0)
Pascal VOC Det (mAP)	75.6 (+0.0)	73.9 (-1.7)	75.1 (-0.5)	76.7 (+1.1)

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

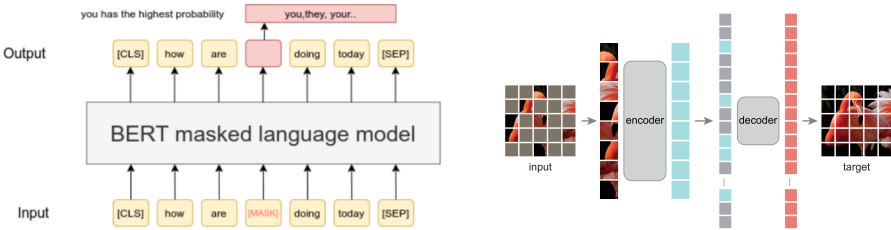
└ Mixup, Cutout and Cutmix

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



Types of SSL approaches

- **Masked Input Modeling:** Predicting missing part of the input

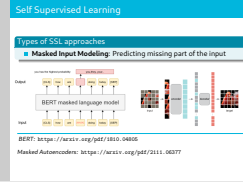


BERT: <https://arxiv.org/pdf/1810.04805>

Masked Autoencoders: <https://arxiv.org/pdf/2111.06377>

2025-02-12 Regularization, DA and SSL

└ Self Supervised Learning



Types of SSL approaches

- **Masked Input Modeling:** Predicting missing part of the input
- **Contrastive Learning:** Pulling together similar representations and pushing apart dissimilar ones

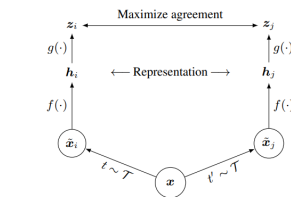
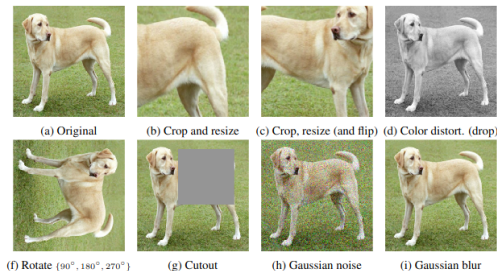
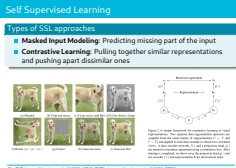


Figure 2. A simple framework for contrastive learning of visual representations. Two separate data augmentation operators are sampled from the same family of augmentations ($t \sim \mathcal{T}$ and $t' \sim \mathcal{T}$) and applied to each data example to obtain two correlated views. A base encoder 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 h for downstream tasks.

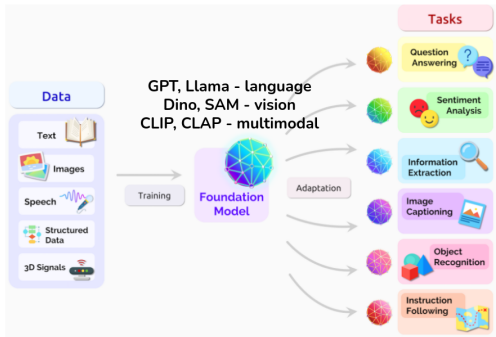
Self Supervised Learning

In self-supervised learning, a pretext task is an auxiliary task that is designed to generate labels from the data itself, without the need for human-labeled data. The goal of the pretext task is to help the model learn useful representations of the data by solving a problem that doesn't necessarily align with the final task the model will perform, but that forces the model to extract meaningful features from the data. Once the model has learned to solve the pretext task, the learned features (or representations) can be transferred to a downstream task, such as classification, detection, or other supervised tasks where labeled data is available.



Self Supervised Learning

SSL to pretrain foundation models



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

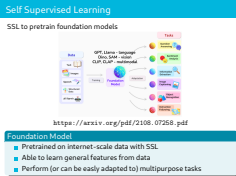
Foundation Model

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

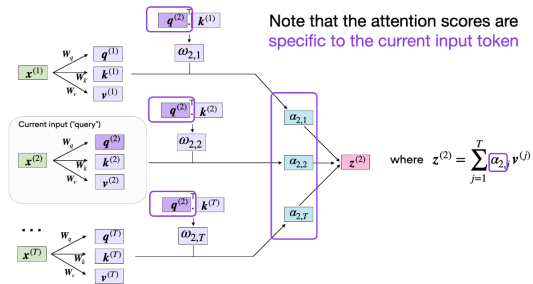
2025-02-12

Regularization, DA and SSL

└ Self Supervised Learning



Self-Attention



Self-Attention in foundation models

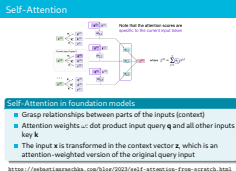
- Grasp relationships between parts of the inputs (context)
- Attention weights ω : dot product input query \mathbf{q} and all other inputs key \mathbf{k}
- The input \mathbf{x} is transformed in the context vector \mathbf{z} , which is an attention-weighted version of the original query input

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

2025-02-12

Regularization, DA and SSL

└ Self-Attention

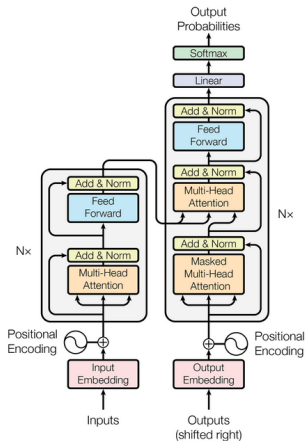


Self-Attention

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>

