

Neural Networks and Deep Learning

Lecture 4: Convolutional neural networks (CNN) and transfer learning





Prof. dr. Daniel Traian PELE

Bucharest University of Economic Studies (ASE), Department of Statistics and Econometrics
Romanian Academy, Institute for Economic Forecasting (IPE)
IDA Institute Digital Assets ▪ AI4EFin ▪ MSCA Digital Finance

Learning objectives

- Write a convolutional layer as a formula: kernel, padding, stride, channels, parameters and receptive field.
- Explain why a convolutional neural network (CNN) needs far fewer parameters than a multilayer perceptron (MLP) on images, and verify it on real data.
- Read the classic architectures, from LeNet to ResNet, and the residual connection that makes deep networks trainable.
- Use transfer learning (feature extraction and fine-tuning) when labelled images are expensive, and measure what it saves.
- Explain a prediction with a gradient-weighted class activation map (CAM) and turn image predictions into business numbers.

Abbreviations

AI	artificial intelligence
ASE	Bucharest University of Economic Studies
BN	batch normalisation
CAM	class activation map
CIFAR	Canadian Institute for Advanced Research (CIFAR-10 image set)
CNN	convolutional neural network
CPU	central processing unit
EU	European Union
GAP	global average pooling
GPT	generative pre-trained transformer
GPU	graphics processing unit
ICBE	International Conference on Business Excellence
IPE	Institute for Economic Forecasting, Romanian Academy
MLP	multilayer perceptron
MNIST	Modified National Institute of Standards and Technology (digit images)
MPS	Metal Performance Shaders (Apple GPU back end of PyTorch)
ReLU	rectified linear unit
RGB	red, green, blue (colour channels)
SGD	stochastic gradient descent
VGG	Visual Geometry Group, University of Oxford
VLM	vision–language model

The data: three real image collections

- The Modified National Institute of Standards and Technology (MNIST) digits are replaced in Fashion-MNIST by clothing photos; the Canadian Institute for Advanced Research (CIFAR) released CIFAR-10.

Data set	Content	Split (train/valid/test)	Business use
Fashion-MNIST	Zalando catalogue photos, 10 product types, 28×28 grey	50 000 / 10 000 / 10 000	automatic product tagging
CIFAR-10	photographs, 10 object classes, 32×32 colour	45 000 / 5 000 / 10 000	benchmark for architectures
EuroSAT	Sentinel-2 satellite tiles of Europe, 10 land-use classes, 64×64 colour	16 200 / 5 400 / 5 400	agriculture, insurance, real estate

- Sources: Zalando Research (Xiao et al., 2017); Krizhevsky (2009); Helber et al. (2019), Copernicus Sentinel-2 data; all downloaded by the course scripts (data/SOURCES.md).
- Validation parts are used for early stopping and tuning; every test set is used once, at the end.
- Pre-trained weights: ResNet-18 trained on ImageNet (torchvision); training on an Apple graphics processing unit (GPU) through Metal Performance Shaders (MPS), seed 20261108.

(data/SOURCES.md)

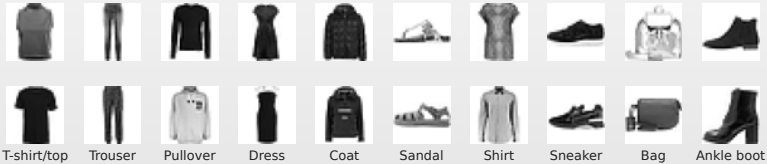
Outline

- Images as data and the motivation for convolutions
- The convolutional layer
- Case study: classification of product photos with softmax regression and a CNN
- Architectures: from LeNet to ResNet
- Transfer learning: land use from satellite images

A product photo is a matrix of numbers

- **In one sentence:** each image is a 28×28 array of grey levels; a model must map it to one of 10 product types.

$$\mathbf{X} \in [0,1]^{28\times 28}, \qquad \hat{\mathbf{p}} = f_\theta(\mathbf{X}) \in \Delta^9, \qquad \hat{y} = \arg\max_k \hat{p}_k$$



Source: Fashion-MNIST, Zalando Research (Xiao et al., 2017). Data: 60,000 training images, 28×28 grey levels.

- Fashion-MNIST: 50 000 training, 10 000 validation and 10 000 test images from the Zalando catalogue.

- **In practice:** an online retailer tags thousands of new product photos a day
  - A classifier sets the category, a person checks the doubtful ones.

(Xiao et al., 2017)

NN_L4_fmnist_samples

A dense layer on pixels wastes parameters and ignores space

- **In one sentence:** an MLP treats pixel (1,1) and pixel (1,2) as unrelated inputs and needs one weight per pixel and unit.

$$\mathbf{h} = \sigma(\mathbf{W}\,\mathrm{vec}(\mathbf{X}) + \mathbf{b}), \quad \mathbf{W} \in \mathbb{R}^{m \times HWC}$$

- **where**
  - H, W height and width
  - C channels
  - m hidden units
  - A 224×224 colour image and $m = 1000$ give $224 \cdot 224 \cdot 3 \cdot 1000 \approx 1.5 \cdot 10^8$ weights in the first layer alone

- **Three ideas of a CNN**
  - Local connectivity: a unit sees a small patch
  - Weight sharing: the same filter at every position
  - Equivariance: shifting the input shifts the feature map, $f(T_\delta \mathbf{X}) = T_\delta f(\mathbf{X})$
- On our data: MLP 235 146 parameters, CNN 140 778, 1.67 times fewer.

- **In practice:** fewer parameters mean less data needed for the same accuracy, which is what labelled business images cost.

(LeCun et al., 1998; Bishop & Bishop, 2024, ch. 10)

Outline

- Images as data and the motivation for convolutions
- The convolutional layer
- Case study: classification of product photos with softmax regression and a CNN
- Architectures: from LeNet to ResNet
- Transfer learning: land use from satellite images

Convolution: the same small filter slides over the image

- **In one sentence:** each output value is a weighted sum of a $k \times k$ neighbourhood, with weights shared across positions.

$$Z_o(i,j) = b_o + \sum_{c=1}^{C_{\text{in}}} \sum_{u=0}^{k-1} \sum_{v=0}^{k-1} K_{o,c}(u,v)\, X_c(is+u-p,\ js+v-p)$$

- Feature map of filter o: $A_o = \text{ReLU}(Z_o)$, with the rectified linear unit (ReLU) $\max(0, z)$
  - K is learned by backpropagation.

- **where**
  - X_c input channel c
  - $K_{o,c}$ the $k \times k$ kernel from channel c to output map o
  - b_o bias
  - s stride
  - p zero padding

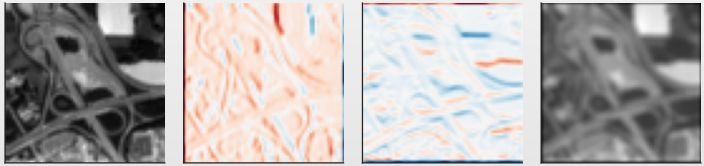
- **In practice:** a filter is a pattern detector that is tested at every position of the image at the cost of $k^2 C_{\text{in}}$ weights.

(Goodfellow et al., 2016, ch. 9)

Hand-made filters already find roads and field edges

- **In one sentence:** a Sobel kernel responds where the brightness changes along one direction; a mean kernel blurs.

$$K_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, \quad K_y = K_x^\top, \quad K_{\text{mean}} = \tfrac{1}{9}\mathbf{1}_{3\times 3}$$



Input (highway) Vertical edges Horizontal edges Mean 3×3 (blur)

Source: EuroSAT, Sentinel-2 images (Helber et al., 2019). Data: one 64 × 64 Sentinel-2 tile (class Highway), grey levels.

- Padding $p = 1$ keeps the 64×64 size
  - A CNN learns dozens of such kernels per layer instead of fixing them.

(own computation)

NN_L4_conv_edges

Output size, padding and stride

- **In one sentence:** three integers decide how large each feature map is.

$$H_{\text{out}} = \left\lfloor \frac{H_{\text{in}} + 2p - k}{s} \right\rfloor + 1$$

- **where**
  - H_{in} input height (same formula for width)
  - k kernel size
  - p padding
  - s stride

- **Worked examples**, $H_{\text{in}} = 64$, $k = 3$
  - $p = 1, s = 1$: $H_{\text{out}} = 64$ ("same")
  - $p = 0, s = 1$: $H_{\text{out}} = 62$ ("valid")
  - $p = 1, s = 2$: $H_{\text{out}} = 32$ (downsampling)
- Parameters of the layer: $(k^2 C_{\text{in}} + 1)\, C_{\text{out}}$, independent of the image size.

- **In practice:** the same network runs on any image size
  - Only the cost in operations, $H_{\text{out}} W_{\text{out}} k^2 C_{\text{in}} C_{\text{out}}$, grows.

(own computation)

Pooling summarises a neighbourhood; global pooling summarises the image

- **In one sentence:** max pooling keeps the strongest response in each window and makes the map smaller and slightly shift-invariant.

$$P_o(i,j) = \max_{0 \le u,v < 2} A_o(2i+u, 2j+v)$$

$$g_o = \frac{1}{HW} \sum_{i,j} A_o(i,j)$$

- **where**
  - P_o max pooling 2×2, stride 2 (halves height and width)
  - g_o global average pooling (GAP), one number per channel

- Our Fashion-MNIST CNN: maps $28 \times 28 \to 14 \times 14 \to 7 \times 7$, then GAP gives 128 numbers and one linear layer gives the 10 class scores.
- Gradient of max pooling: it flows only to the arg-max position of each window.
- GAP replaces large dense layers (thousands of weights) by an average (zero weights).

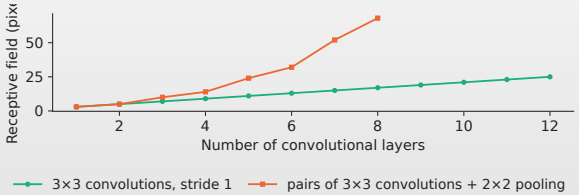
- **In practice:** GAP is the reason a modern CNN has few parameters in its head and can be explained with class activation maps.

(Bishop & Bishop, 2024, sec. 10.2)

Depth, pooling and the receptive field

- **In one sentence:** the receptive field is the patch of the input that can influence one unit.

$$r_l = r_{l-1} + (k_l - 1)\prod_{i<l} s_i, \qquad r_0 = 1$$



Source: own computation from the receptive-field recursion. Data: recursion of the receptive field, no data.

- Twelve 3×3 layers at stride 1 see only 25×25 pixels
  - Eight with 2×2 pooling after every pair see 68×68.

(own computation)

NN_L4_receptive_field

Backpropagation through a convolution is again a convolution

- **In one sentence:** because the kernel is shared, its gradient sums the contributions of every position.

$$\frac{\partial \mathcal{L}}{\partial K_{o,c}(u,v)} = \sum_{i,j} \delta_o(i,j)\, X_c(is+u-p,\, js+v-p)$$

$$\delta_o(i,j) = \frac{\partial \mathcal{L}}{\partial Z_o(i,j)}$$

- **where**
  - δ_o the error map of output o, computed by backpropagation from the layer above
  - For $s=1$, $\partial\mathcal{L}/\partial X_c = \sum_o \delta_o \star \mathrm{rot}_{180}(K_{o,c})$, a full convolution

- Weight sharing turns millions of position-specific gradients into one sum per weight.
- In code, autograd does it: `loss.backward()` on a `nn.Conv2d`.
- Derivation in the appendix (convolution as a matrix product).

- **In practice:** one forward and one backward pass cost about the same number of operations, so training time scales with the forward cost.

(Goodfellow et al., 2016, sec. 9.5)

Batch normalisation keeps every channel on the same scale

- **In one sentence:** each channel is standardised over the batch and the positions, then rescaled by two learned numbers.

$$\hat{a} = \frac{a - \mu_{\mathcal{B},o}}{\sqrt{\sigma^2_{\mathcal{B},o} + \epsilon}}, \qquad \mathrm{BN}(a) = \gamma_o \hat{a} + \beta_o$$

- **where**
  - $\mu_{\mathcal{B},o}, \sigma^2_{\mathcal{B},o}$ mean and variance of channel o over the batch $\mathcal{B}$ and all positions
  - γ_o, β_o learned
  - At test time running averages replace the batch statistics

- Every network of this lecture uses batch normalisation (BN) after each convolution: Conv $\rightarrow$ BN $\rightarrow$ ReLU.
- It allows larger learning rates
  - Our CIFAR runs use a one-cycle schedule up to 0.1 (chosen a priori).
- Remember `model.eval()` before prediction: otherwise BN uses the test batch.

- **In practice:** a model that scores well in the notebook but badly in production often forgot eval mode or the input scaling of training.

(Ioffe & Szegedy, 2015)

Outline

- Images as data and the motivation for convolutions
- The convolutional layer
- Case study: classification of product photos with softmax regression and a CNN
- Architectures: from LeNet to ResNet
- Transfer learning: land use from satellite images

Experimental protocol for the three models

- **Benchmark**: softmax (multinomial logistic) regression
  - $\hat{\mathbf{p}} = \text{softmax}(\mathbf{W}\text{vec}(\mathbf{X}) + \mathbf{b})$, 7 850 parameters
- **MLP**: 784–256–128–10, ReLU, dropout 0.2
  - 235 146 parameters
- **CNN**: 2 blocks [Conv3×3–BN–ReLU]×2 + max pool, Conv 128, GAP, linear
  - 140 778 parameters

- **Same training for all**
  - AdamW, batch 128, 12 epochs, one-cycle learning rate
  - Cross-entropy $\mathcal{L} = -\frac{1}{n}\sum_i \log \hat{p}_{i,y_i}$
  - Pixels standardised with the training mean 0.2859 and standard deviation 0.3530
- **Fair comparison**
  - Early stopping on the 10 000 validation images
  - 3 seeds per model; test images used once

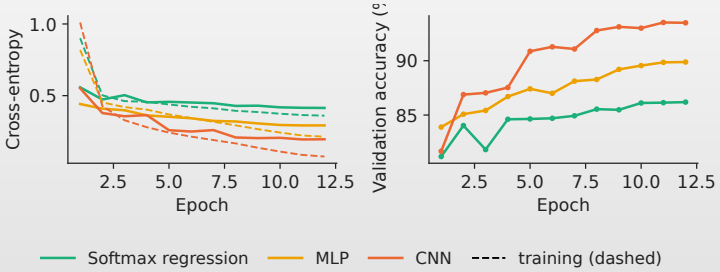
In practice: total training time of the nine runs: 299 seconds on the GPU of a laptop.

(own computation)

Experiment yourself: §3

The CNN learns faster and generalises better

- **In one sentence:** validation loss falls lowest for the CNN; the MLP fits the training data but stops improving on new images.



Source: Fashion-MNIST, Zalando Research (Xiao et al., 2017). Data: 50,000 training and 10,000 validation images; seed 20261108.

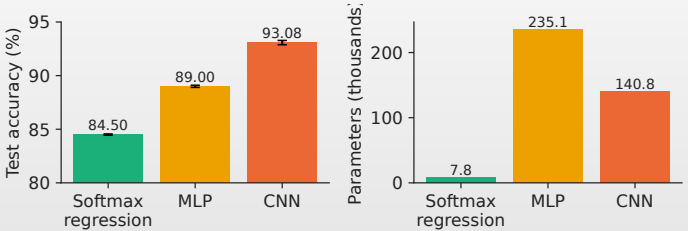
- Best validation epoch (seed 20261108): CNN 11, MLP 12, softmax 12, out of 12.

(own computation)

NN_L4_fmnist_curves

Accuracy and number of parameters of the three models

- **In one sentence:** weight sharing buys accuracy and saves parameters at the same time.



Source: Fashion-MNIST, Zalando Research (Xiao et al., 2017). Data: 10,000 test images; mean and standard deviation over 3 seeds.

- Test accuracy (mean $\pm$ sd, 3 seeds): softmax 84.50 ± 0.06 %, MLP 89.00 ± 0.10 %, CNN 93.08 ± 0.20 %.
- The CNN removes 37.1 % of the MLP's errors with 1.67 times fewer parameters.

(own computation)

NN_L4_fmnist_compare

McNemar's test of the accuracy gain on the same test images

- **In one sentence:** two classifiers scored on the same images differ only through the images on which exactly one of them is right.

$$\chi^2 = \frac{(|b - c| - 1)^2}{b + c} \overset{H_0}{\sim} \chi^2_1$$

- **where**
  - b images right for the CNN only
  - c right for the MLP only
  - H_0: both have the same error rate
  - 5 % critical value 3.84

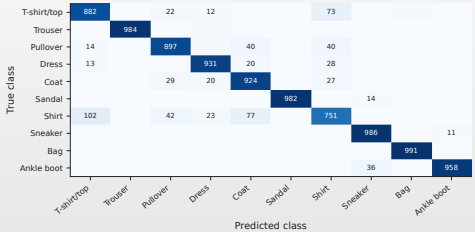
- Seed 20261108: $b = 614$, $c = 226$, $\chi^2 = 178.3$: the null is rejected.
- Across the 3 seeds, the paired t statistic of the accuracy gap is 31.38.
- Seeds measure the training noise
  - McNemar measures the test-set noise
  - Report both.

- **In practice:** before replacing a production model, show that the gain survives a test on the same cases, not only a higher average.

(McNemar, 1947)

Confusion matrix: errors between products of similar shape

- **In one sentence:** most mistakes are between upper-body garments of similar shape.



Source: Fashion-MNIST, Zalando Research (Xiao et al., 2017). Data: 10,000 test images, CNN seed 20261108.

- Weakest class: Shirt (recall 75.1 %)
  - Largest confusion: Shirt taken for T-shirt/top, 102 images.

- **In practice:** route only low-confidence photos of the weak classes to a human: the error cost falls where it is highest.

(own computation)

NN_L4_fmnist_confusion

First-layer filters are edge and contrast detectors

- **In one sentence:** nobody told the network to look for edges; gradient descent found them.



Source: Fashion-MNIST, Zalando Research (Xiao et al., 2017). Data: first convolutional layer of the CNN (32 filters of 3 x 3), seed 20261108.

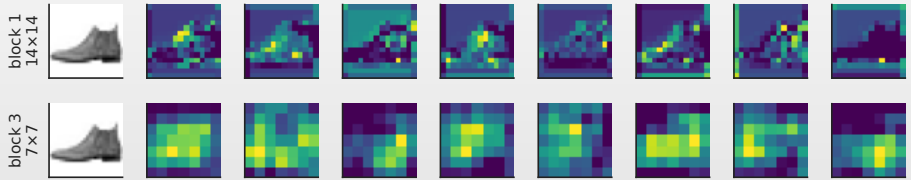
- Red = positive weight, blue = negative: many kernels are oriented differences, like the Sobel kernels set by hand.
- Deeper layers combine them into parts (collars, soles, handles) (Zeiler & Fergus, 2014).

(own computation; Zeiler & Fergus, 2014)

NN_L4_fmnist_filters

Feature maps at increasing depth

- **In one sentence:** early maps keep the outline of the boot; deep maps are small but say where the informative parts are.



Source: Fashion-MNIST, Zalando Research (Xiao et al., 2017). Data: one test image (ankle boot); 8 most active channels per block.

- Block 1: 32 maps of 14×14
  - Block 3: 128 maps of 7×7, then averaged by GAP.

(own computation)

NN_L4_fmnist_featuremaps

Outline

- Images as data and the motivation for convolutions
- The convolutional layer
- Case study: classification of product photos with softmax regression and a CNN
- Architectures: from LeNet to ResNet
- Transfer learning: land use from satellite images

Image-recognition architectures, 1998–2021

Network	Year	Key idea
LeNet-5	1998	convolution + subsampling for handwritten digits (the MNIST benchmark); trained by backpropagation
AlexNet	2012	deep CNN on ImageNet trained on GPUs; ReLU, dropout, data augmentation
Visual Geometry Group (VGG)	2015	only 3×3 kernels: two stacked 3×3 see 5×5 with $18C^2$ instead of $25C^2$ weights
ResNet	2016	residual connections $\mathbf{y} = \mathbf{x} + F(\mathbf{x})$: networks of 100+ layers become trainable
Vision transformer	2021	image patches as tokens with attention (Lecture 7)

⊡ ResNet-18, used for transfer learning below, has 11 181 642 parameters (our count, with a 10-class head).

(LeCun et al., 1998; Krizhevsky et al., 2012; Simonyan & Zisserman, 2015; He et al., 2016)

Residual connections and the identity path

- **In one sentence:** a block adds its output to its input, so the gradient always has a direct path back.

$$\mathbf{y} = \mathrm{ReLU}\left(\mathbf{x} + F(\mathbf{x}; \mathbf{W})\right), \qquad \frac{\partial \mathcal{L}}{\partial \mathbf{x}} = \frac{\partial \mathcal{L}}{\partial \mathbf{y}}\left(\mathbf{I} + \frac{\partial F}{\partial \mathbf{x}}\right)$$

- **where**
  - F two Conv–BN layers
  - When the shape changes, $\mathbf{x}$ passes through a 1×1 convolution with stride 2
  - The derivative is written for the pre-activation sum

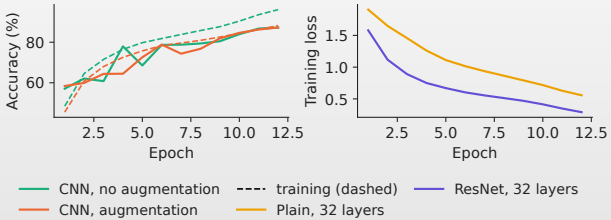
- Plain deep networks suffer from degradation: deeper ones have higher training error, which is an optimisation problem, not overfitting.
- If the best block is the identity, it suffices to set $F = 0$.
- On CIFAR-10 (next frame): same 32 layers, with and without the shortcut.

- **In practice:** residual blocks are the default in every modern vision and language model, including transformers.

(He et al., 2016)

Effects of data augmentation and residual connections

- **In one sentence:** left: the same CNN on Canadian Institute for Advanced Research (CIFAR-10) photos; right: 32-layer networks that differ only by the shortcut.



Source: CIFAR-10 (Krizhevsky, 2009). Data: 45,000 training and 5,000 validation images; seed 20261108.

- Test accuracy: CNN 86.56 % without and 86.86 % with augmentation (train–valid gap 8.9 vs 0.9 points).
- Plain-32 79.96 % vs ResNet-32 87.81 %
  - Final training loss 0.557 vs 0.291.

(own computation)

NN_L4_cifar_curves

Data augmentation as prior information

- **In one sentence:** we train on random transformations that should not change the label, which averages the loss over them.

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{T \sim \mathcal{T}} \Big[\ell\big(f_\theta(T(\mathbf{X}_i)), y_i\big) \Big]$$

- **where**
  - $\mathcal{T}$ the set of label-preserving transformations
  - CIFAR-10: random crops after 4-pixel padding and horizontal flips
  - EuroSAT: flips and rotations by 90°

- Gain on CIFAR-10 for the same network and budget: +0.30 percentage points of test accuracy.
- Choose $\mathcal{T}$ by domain: a flipped digit 6 is not a 6, but a satellite tile has no "up".
- ResNet-32 per-class recall: best automobile 95.7 %, worst cat 71.9 %.

- **In practice:** augmentation is free extra data
  - It is the first lever to pull before buying more labels.

(own computation)

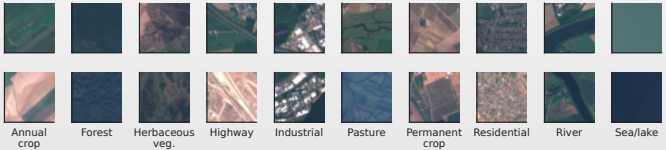
Outline

- Images as data and the motivation for convolutions
- The convolutional layer
- Case study: classification of product photos with softmax regression and a CNN
- Architectures: from LeNet to ResNet
- Transfer learning: land use from satellite images

Satellite tiles: land use for lenders, insurers and investors

- **In one sentence:** ten land-use classes from Sentinel-2 images of European Union (EU) countries; the question is how few labels we can afford.

$$\mathbf{X} \in [0,1]^{64\times64\times3}, \qquad y \in \{\text{crop}, \ldots, \text{sea/lake}\}$$



Source: EuroSAT, Sentinel-2 images (Helber et al., 2019). Data: 27,000 tiles of 64 x 64 pixels, 10 classes.

- 27 000 red, green, blue (RGB) tiles of 64×64 pixels
  - 2 000 to 3 000 per class.

- **In practice:** an agricultural lender checks that a farm loan finances crops
  - An insurer maps flood-exposed residential areas.

(Helber et al., 2019)

NN_L4_eurosat_samples

Transfer learning: features learned on a million photos

- **In one sentence:** the early layers of a network trained on ImageNet detect edges and textures that are useful for any image.

$$f_\theta(\mathbf{X}) = g_\psi\left(\phi_\omega(\mathbf{X})\right), \quad \omega^{(0)} = \omega_{\text{ImageNet}}$$

- **where**
  - ϕ_ω the convolutional body (512 features after GAP)
  - g_ψ a new linear head with 10 outputs

- **Feature extraction**
  - ω frozen; fit only ψ (here a logistic regression, C chosen on validation)
- **Fine-tuning**
  - All weights trained, smaller learning rate for the body ($3 \cdot 10^{-4}$) than for the head ($3 \cdot 10^{-3}$), chosen a priori
- **From scratch**
  - A 4-block CNN, 1 176 746 parameters, random start

- **In practice:** feature extraction is the cheapest: one pass over the images and a logistic regression.

(Yosinski et al., 2014)

Experimental protocol for the three transfer strategies

- **Split**, stratified by class
  - 16 200 train / 5 400 validation / 5 400 test tiles
- **Input**
  - Tiles resized to 112×112 for ResNet-18, scaled with the ImageNet mean and standard deviation per channel
  - $\tilde{x}_c = (x_c - \mu_c)/s_c$
- **Training**
  - AdamW, batch 64, one-cycle schedule; flips and 90° rotations
  - From scratch 18 epochs; fine-tuning 6 epochs
  - Model kept at its best validation accuracy
- **Cost**
  - 1225 seconds for the three main runs on a laptop GPU

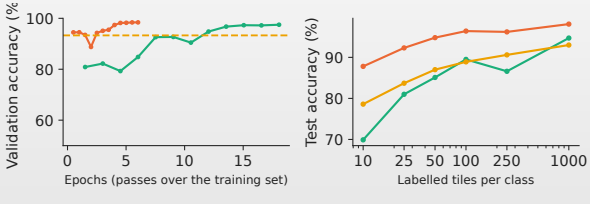
- **In practice:** fixing the test tiles before any modelling is what makes the three strategies comparable.

(own computation)

Experiment yourself: §5

Fine-tuning is the most accurate, and transfer matters most when labels are few

- **In one sentence:** left: learning with all labels; right: test accuracy when only n labelled tiles per class are available.



Source: EuroSAT, Sentinel-2 images (Helber et al., 2019). Data: 16,200 training, 5,400 validation, 5,400 test tiles; right: mean of 2 seeds.

- Test accuracy, all labels: from scratch 97.78 %, frozen + logistic 93.48 %, fine-tuned 98.76 %.
- With 10 tiles per class: scratch 69.9 %, frozen 78.6 %, fine-tuned 87.8 %.

(own computation)

NN_L4_transfer_curves

The label budget and the savings from transfer learning

Tiles/class	Scratch	Frozen	Fine-tuned
10	69.9	78.6	87.8
25	81.0	83.7	92.3
50	85.1	87.0	94.8
100	89.5	88.9	96.4
250	86.6	90.6	96.2
1000	94.7	93.0	98.1

- Test accuracy (%), mean of 2 random draws of the labelled tiles; same 5,400 test tiles.

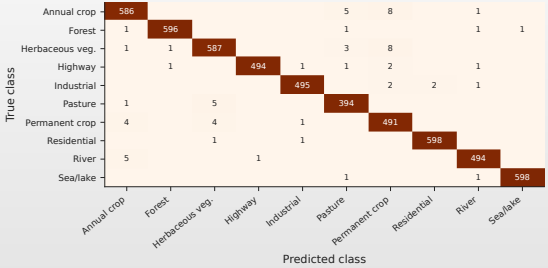
- With 100 labelled tiles in total, the pre-trained network already beats a CNN trained from scratch on the same labels by a wide margin.
- The scratch CNN needs the full budget to approach the fine-tuned network.
- The run of the whole table took 1857 seconds.

- **In practice:** labelling a tile costs an expert's minute
  - Transfer learning turns weeks of labelling into an afternoon.

(own computation)

Remaining errors: annual versus permanent crops

- **In one sentence:** the fine-tuned network mixes classes that look alike from space.



Source: EuroSAT, Sentinel-2 images (Helber et al., 2019). Data: 5,400 test tiles, fine-tuned ResNet-18, seed 20261108.

- 67 errors on 5 400 tiles
  - Largest: Annual crop taken for Permanent crop (8)
  - Macro-F1 98.72 %.

(own computation)

NN_L4_eurosat_confusion

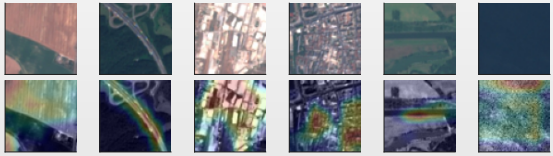
Grad-CAM: image regions behind the classification

- **In one sentence:** weight each feature map by how much the class score reacts to it, sum, and keep the positive part.

$$\alpha_o^c = \frac{1}{HW} \sum_{i,j} \frac{\partial z_c}{\partial A_o(i,j)}$$

$$M^c = \mathrm{ReLU}\left(\sum_o \alpha_o^c A_o \right)$$

- **where**
  - z_c the score (logit) of class c
  - A_o feature map o of the chosen layer
  - M^c the class activation map (CAM), upsampled to the tile



Source: EuroSAT, Sentinel-2 images (Helber et al., 2019). Data: six test tiles; Grad-CAM on the 7 x 7 maps of block 3 of the fine-tuned ResNet-18.

- **In practice:** an auditor can check that "industrial" is decided by the roofs, not by a cloud or a border artefact.

(Selvaraju et al., 2017; own computation)

NN_L4_gradcam

Land-use shares estimated from satellite tiles

- **In one sentence:** a portfolio question is about shares and totals, so errors that cancel matter less than errors that do not.

$$\hat{\pi}_k = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{\hat{y}_i = k\}, \qquad \mathbb{E}[\hat{\pi}_k] = \sum_j \pi_j \, P(\hat{y} = k \mid y = j)$$

- **where**
  - π_k true share of class k in a region
  - $P(\hat{y} = k \mid y = j)$ the confusion matrix, estimated on the test tiles

- Test tiles: share of crops (annual + permanent) 20.37 % true vs 20.54 % predicted.
- Largest error of any class share: 0.20 percentage points.
- Inverting the confusion matrix corrects the shares when the error rates are stable (domain shift breaks this).

- **In practice:** for an insurer or an agricultural lender the output is a share of hectares per region, with an error bar, not a label per tile.

(own computation)

Benefits and risks of transfer learning

- **Helps most**
  - Few labels (right panel of the transfer chart)
  - A source domain with similar low-level statistics (edges, textures)
  - A limited compute budget: frozen features cost one forward pass
- **Can mislead**
  - Domain shift: other season, other sensor, other country than the training tiles
  - Spurious cues learned from the source (watermarks, borders)
  - The test set must come from the population where the model will be used

- **In practice:** before deployment, label a small sample from the target region and re-measure
  - The fine-tuned model is cheap to update.

(Yosinski et al., 2014; Helber et al., 2019)

Convolutions for sequences and charts

- **In one sentence:** a one-dimensional convolution slides over time, the same way a two-dimensional one slides over space.

$$z_t = b + \sum_{c} \sum_{u=0}^{k-1} K_c(u)\, x_{c,\,t-u}$$

- **where**
  - $x_{c,t}$ input series c at time t (returns, volume, load)
  - Causal: only past values

- A one-dimensional CNN is a learned bank of moving-average filters
  - Lecture 5 compares it with recurrent networks.
- Charts of markets can be read as images by a vision–language model (VLM): research of our group, next section.
- Same rules: real data, a time-ordered split, a simple benchmark first.

- **In practice:** a risk desk can screen hundreds of chart panels per day with a model, and keep the analyst for the flagged ones.

(Bishop & Bishop, 2024, ch. 10)

Quiz 1: output size

- A 64×64 tile passes a 3×3 convolution with padding 1 and stride 2
  - How large is the output map?

Answer

- $\lfloor (64 + 2 - 3)/2 \rfloor + 1 = 32$, so 32×32.

Quiz 2: parameters

- Why does our CNN beat the MLP on Fashion-MNIST with fewer parameters?

Answer

- Weight sharing and local connectivity encode the structure of images
  - The CNN reaches 93.08 % with 140 778 parameters against 89.00 % with 235 146.

Quiz 3: depth

- A plain 32-layer network has a higher training loss than a residual one of the same depth
  - Is that overfitting?

Answer

- No: overfitting lowers the training loss
  - This is the degradation problem (0.557 vs 0.291), solved by the identity shortcut.

Quiz 4: a label budget

- You can label only 100 satellite tiles in total
  - Which strategy do you choose?

Answer

- A pre-trained network: with 10 tiles per class, fine-tuning gives 87.8 % and frozen features 78.6 %, against 69.9 % from scratch.

Quiz 5: explanation

- A Grad-CAM map of an “industrial” tile lights up a cloud
  - What do you do?

Answer

- Treat it as a warning of a spurious cue: inspect similar tiles, add cloudy examples or a cloud mask, and re-test on the target region.

Wrap-up

- Images as data and the motivation for convolutions
- The convolutional layer
- Case study: classification of product photos with softmax regression and a CNN
- Architectures: from LeNet to ResNet
- Transfer learning: land use from satellite images

- A convolution is a shared local filter
  - Padding, stride and pooling set the map size, depth and pooling the receptive field.
- On real images the CNN beats the MLP with fewer parameters
  - Residual blocks make depth trainable
  - Augmentation is free data.
- Transfer learning wins when labels are scarce.
  - Next: sequences and recurrent networks (Lecture 5).

Research from our group: vision models for finance

- **A multimodal vision–language framework for financial anomaly detection** (Jheng, Pele, Tak & Găman, 2026, PICBE), Best Paper Award at the International Conference on Business Excellence (ICBE) 2026
  - Market indicators projected onto a standardised multi-panel image; market regime classification treated as visual recognition
  - Zero-shot vision–language model (GPT-4.1, a generative pre-trained transformer) with rolling-quantile labels and structured natural-language rationales
  - Out-of-sample tests on the S&P 500: promising fidelity in detecting systemic anomalies, with interpretable explanations
- **Quantlet: the code snippet knowledge platform** (Găman, Winkel, Zuo, Härdle & Pele, 2026, Computational Statistics)
  - The platform on which every chart of this lecture is published as a reproducible Quantlet
- Findings as stated in the papers; numbers on the other slides come from our own computations.

(Bucharest University of Economic Studies; Romanian Academy, Institute for Economic Forecasting)

Further reading and resources

- **Bishop & Bishop (2024)**, ch. 10: convolutional networks, with the derivations of this lecture.
- **Goodfellow, Bengio & Courville (2016)**, ch. 9: convolution, pooling, variants.
- **He et al. (2016)**: the ResNet paper, short and readable.
- **This course**: code, data, notebooks — github.com/danpele/NEURAL_BIZ.
- **Experiment yourself**: the notebook reproduces the charts of this lecture — notebooks/.
- **Seminar 4**: transfer learning on EuroSAT with your own label budget.

References (1/2)

- Bishop, C. M. & Bishop, H. (2024). *Deep Learning: Foundations and Concepts*, ch. 10. Springer. doi:10.1007/978-3-031-45468-4
- Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning*, ch. 9. MIT Press. deeplearningbook.org
- He, K., Zhang, X., Ren, S. & Sun, J. (2016). Deep residual learning for image recognition. *CVPR*, 770–778. doi:10.1109/CVPR.2016.90
- Helber, P., Bischke, B., Dengel, A. & Borth, D. (2019). EuroSAT: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE JSTARS*, 12(7), 2217–2226. doi:10.1109/JSTARS.2019.2918242
- Ioffe, S. & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML*, PMLR 37, 448–456. arXiv:1502.03167
- Jheng, S.-L., Pele, D. T., Tak, R. & Găman, Ș. (2026). A multimodal vision-language framework for financial anomaly detection. *Proceedings of the International Conference on Business Excellence*, 20(1), 191–201. doi:10.2478/picbe-2026-0020
- Găman, Ș., Winkel, J., Zuo, X., Härdle, W. K. & Pele, D. T. (2026). Quantlet: the code snippet knowledge platform. *Computational Statistics*, 41(3), 66. doi:10.1007/s00180-026-01742-6
- Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. Technical report, University of Toronto. cs.toronto.edu
- Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *NIPS*, 25, 1097–1105. doi:10.1145/3065386

References (2/2)

- LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324. doi:10.1109/5.726791
- McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2), 153–157. doi:10.1007/BF02295996
- Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D. & Batra, D. (2017). Grad-CAM: Visual explanations from deep networks via gradient-based localization. *ICCV*, 618–626. doi:10.1109/ICCV.2017.74
- Simonyan, K. & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *ICLR*. arXiv:1409.1556
- Xiao, H., Rasul, K. & Vollgraf, R. (2017). Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. arXiv:1708.07747
- Yosinski, J., Clune, J., Bengio, Y. & Lipson, H. (2014). How transferable are features in deep neural networks? *NIPS*, 27, 3320–3328. arXiv:1411.1792
- Zeiler, M. D. & Fergus, R. (2014). Visualizing and understanding convolutional networks. *ECCV*, 818–833. doi:10.1007/978-3-319-10590-1_53

Appendix: convolution as a matrix product

- Stack every $k \times k \times C_{\text{in}}$ patch of the input as a row of $\mathbf{P} \in \mathbb{R}^{H_{\text{out}} W_{\text{out}} \times k^2 C_{\text{in}}}$ ("im2col").
- Stack the kernels as columns of $\mathbf{K} \in \mathbb{R}^{k^2 C_{\text{in}} \times C_{\text{out}}}$
  - Then $\mathbf{Z} = \mathbf{P}\mathbf{K} + \mathbf{1}\mathbf{b}^\top$.
- Gradients follow from the matrix rules of Lecture 2: $\partial \mathcal{L}/\partial \mathbf{K} = \mathbf{P}^\top \boldsymbol{\Delta}$ and $\partial \mathcal{L}/\partial \mathbf{P} = \boldsymbol{\Delta}\mathbf{K}^\top$, with $\boldsymbol{\Delta} = \partial \mathcal{L}/\partial \mathbf{Z}$.
- Adding back the rows of $\partial \mathcal{L}/\partial \mathbf{P}$ to their pixel positions ("col2im") gives $\partial \mathcal{L}/\partial \mathbf{X}$: the full convolution of the main text.
- A 1×1 convolution is the special case $k = 1$: a dense layer applied to the channels of every pixel.

Appendix: the CNN of the case study in PyTorch

```
block = lambda a, b:  [nn.Conv2d(a, b, 3, padding=1), nn.BatchNorm2d(b), nn.ReLU()]
cnn = nn.Sequential(*block(1, 32), *block(32, 32), nn.MaxPool2d(2),    # 28 -> 14
                    *block(32, 64), *block(64, 64), nn.MaxPool2d(2),    # 14 -> 7
                    *block(64, 128), nn.AdaptiveAvgPool2d(1), nn.Flatten(),
                    nn.Dropout(0.3), nn.Linear(128, 10))
resnet = torchvision.models.resnet18(weights='IMAGENET1K_V1')
resnet.fc = nn.Linear(512, 10)                # new head for the 10 land-use classes
```

- Full code: lectures/L4/analysis_L4.py and the notebook of this lecture.