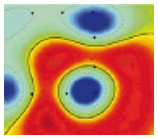
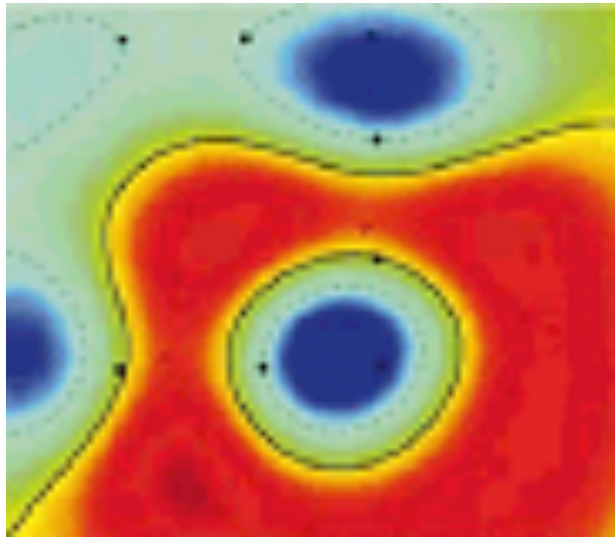


WiSe 2024/25

Deep Learning 1

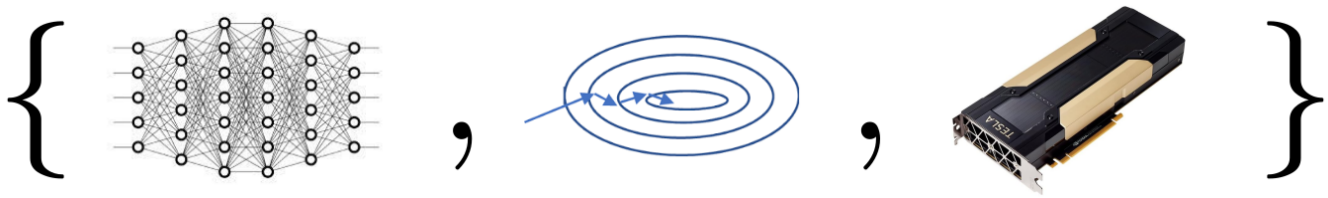
Technische Universität Berlin



Lecture 5

Overfitting & Robustness (1)

Recap Lectures 1–4



Lectures 1–4:

- With flexible neural network architectures, powerful optimization techniques, and fast machines, we have means of producing functions that can accurately fit large amount of highly nonlinear data.

Question:

- Do the learned neural networks generalize to new data, e.g. will it be able to correctly classify *new* images?

The data on which we train the model also matter!

A Bit of Theory

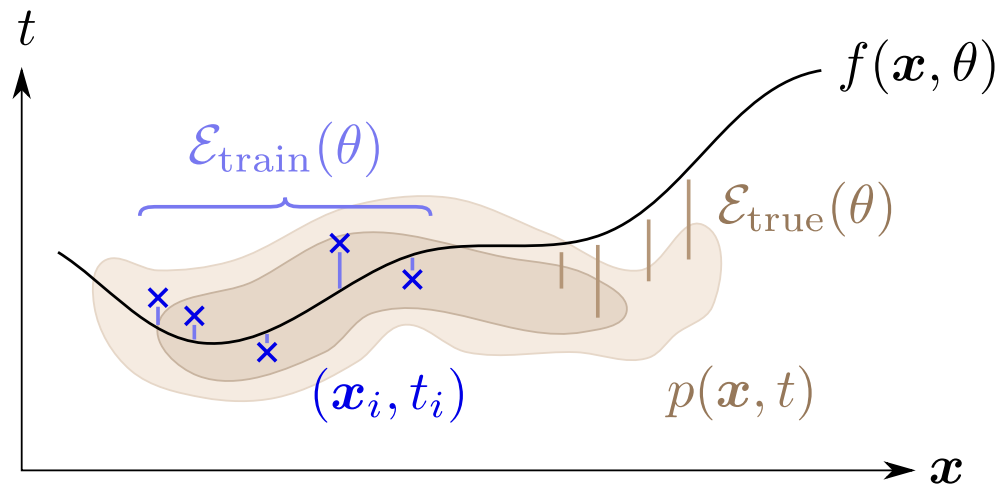
So far, we have only considered the error we use to optimize the model (aka. the training error):

$$\mathcal{E}_{\text{train}}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(f(\boldsymbol{x}_i, \theta) - t_i)^2$$

In practice, what we really care about is the *true* error:

$$\mathcal{E}_{\text{true}}(\theta) = \int (f(\boldsymbol{x}, \theta) - t)^2 \, p(\boldsymbol{x}, t) \, d\boldsymbol{x}dt$$

where $p(\boldsymbol{x}, t)$ is the *true* probability distribution from which the data is coming at test time. The true error is much harder to minimize, because we don't know $p(\boldsymbol{x}, t)$.

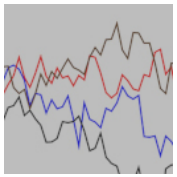


Characterizing Datasets

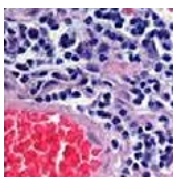
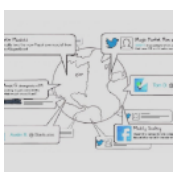
Factors that makes the available dataset $\mathcal{D}$ and the true distribution $p(\boldsymbol{x}, t)$ diverge:

- The fact the dataset is composed of few data points drawn randomly from the underlying data distribution (**finite data**).
- The fact that the dataset may overrepresent certain parts of the underlying distribution, e.g. people of a certain age group (**dataset bias**).
- The fact that the dataset may have been generated from an underlying distribution $p_{\text{old}}(\boldsymbol{x}, t)$ that is now obsolete (**distribution shift**).

Practical Examples

	Data types	Properties
	Image/text data	thousands of images per class, aggregated from many sources. Some image compositions may be overrepresented (**dataset bias / spurious correlations**).
	Sensor data	Potentially very large datasets, but sensors may become decalibrated over time (**distribution shift**).
	Games (GO, chess, etc.)	Infinite number of games states can be produced through computer-computer plays, but master-level plays being more expensive to generate, simple games may be overrepresented (**dataset bias**).

Practical Examples (cont.)

	Data types	Properties
	Simulated data (e.g. physics, car driving)	Theoretically infinite, but practically limited do the cost of running simulations. In practice, we only generate few instances (**finite data**). Problem of transferring from a simulated to a real-world environment (**distribution shift**).
	Medical data	limited number of patients due to rarity of a particular disease, or regulatory constraints (**finite data**, **dataset bias**). Aquisition devices may evolve over time (**distribution shift**).
	Social data	Large amount of data, but only recent data is relevant. Risk of not capturing the most recent trends (**distribution shift**).

Outline

The Problem of Finite Data

- The problem of overfitting
- Mitigating overfitting

Dataset Bias 1: Imbalance Between Subgroups

- Data from Multiple Domains
- Building a 'Domain' Invariant Classifier

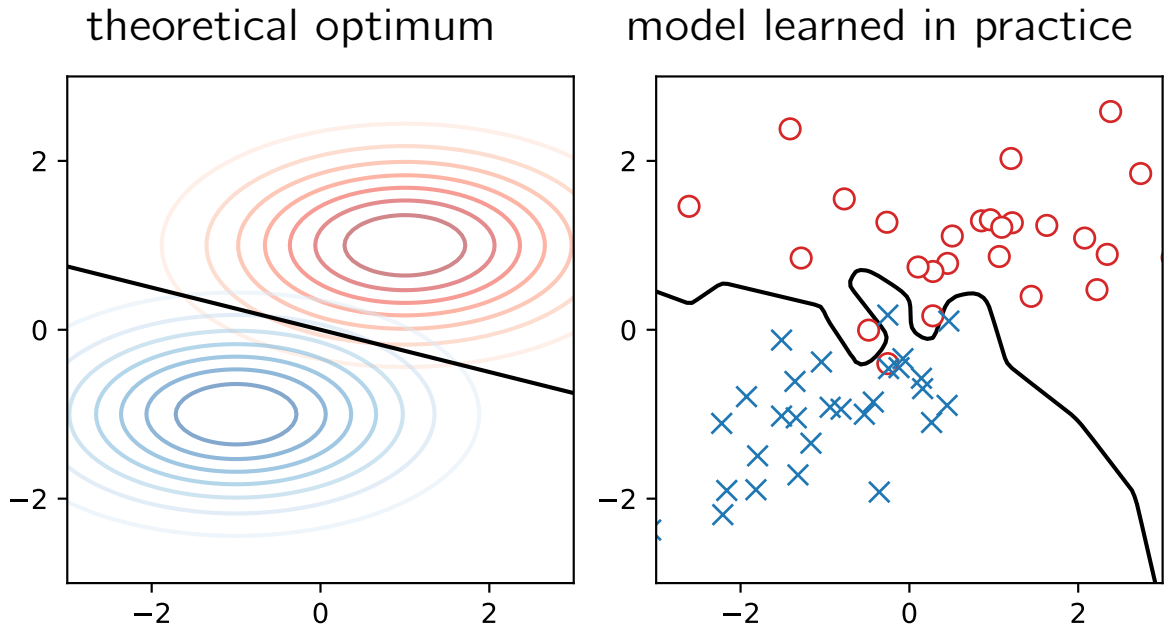
Dataset Bias 2: Spurious Correlations

- Examples of Spurious Correlations
- Detecting and Mitigating Spurious Correlations

Part 1

The Problem of Limited Data

Finite Data and Overfitting

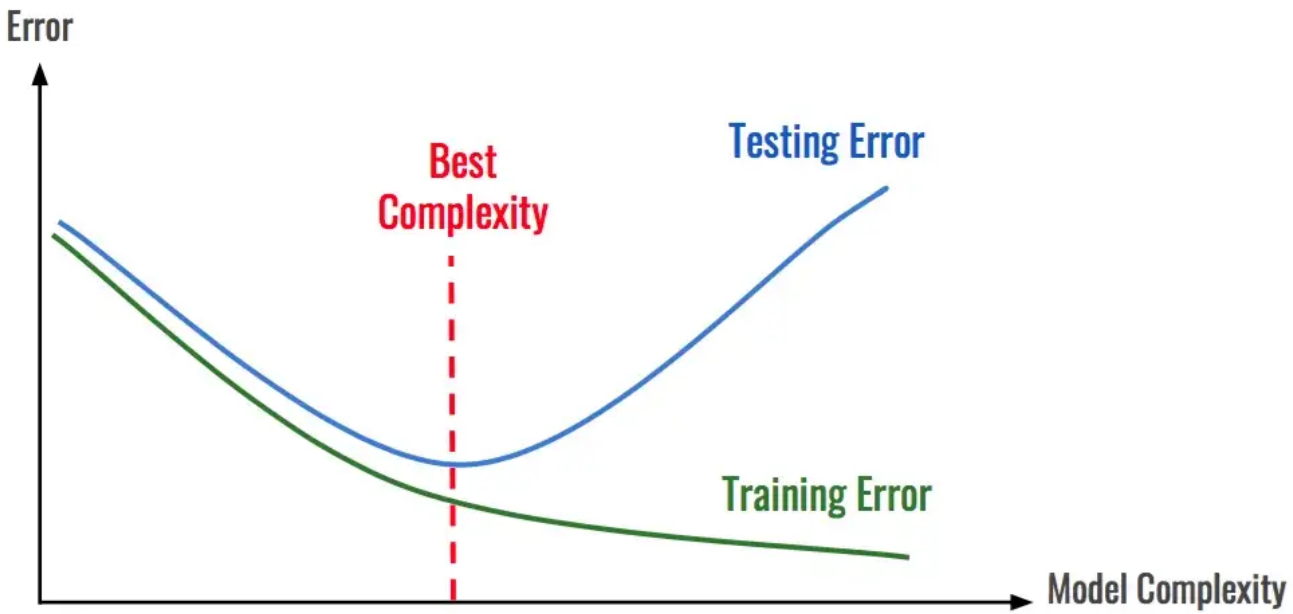


- Assume each data point $\boldsymbol{x} \in \mathbb{R}^d$ and its label $y \in \{0, 1\}$ is generated iid. from two Gaussian distributions.
- With limited data, one class or target value may be locally predominant 'by chance'. Learning these spurious variations is known as overfitting.
- An overfitted model predicts the training data perfectly but works poorly on new data.

Model Error and Model Complexity

William of Ockham (1287-1347)
Linked model complexity to how suitable the model is for explaning phenomena. “Entia non sunt multiplicanda praeter necessitatem”

Vladimir Vapnik
Showed a formal relation between model complexity (measured as the VC-dimension) and the error of a classifier.



Complexity and Generalization Error

Generalization Bound [Vapnik]

- Assume the training data is drawn iid. from the true data distribution.
- Let h denote the VC-dimension, a measure of complexity (or flexibility) of the considered class of models (e.g. one-layer networks).
- Let θ be the parameter of the learned model.
- The difference between the true error $\mathcal{E}_{\text{true}}(\theta)$ and the training error $\mathcal{E}_{\text{train}}(\theta)$ is upper-bounded as:

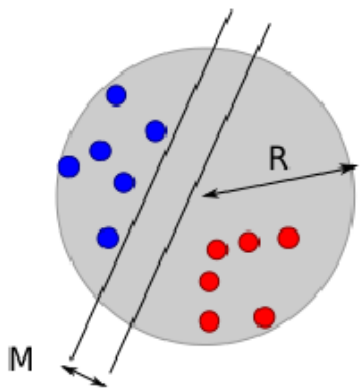
$$\mathcal{E}_{\text{true}}(\theta) - \mathcal{E}_{\text{train}}(\theta) \leq \sqrt{\frac{h(\log \frac{2N}{h} + 1) - \log(\eta/4)}{N}}$$

Factors that reduce the gap between training and true error:

- Lowering the VC-dimension h (i.e. choosing a less flexible class of models).
- Increasing the number of training points N.

Characterizing Complexity (One-Layer Networks)

$$f(\boldsymbol{x}, (\boldsymbol{w}, b)) = \text{sign}(\boldsymbol{w}^\top \boldsymbol{x} + b)$$



$$h = \min\left\{d + 1, 4\frac{R^2}{M^2} + 1\right\}$$

dimensions margin

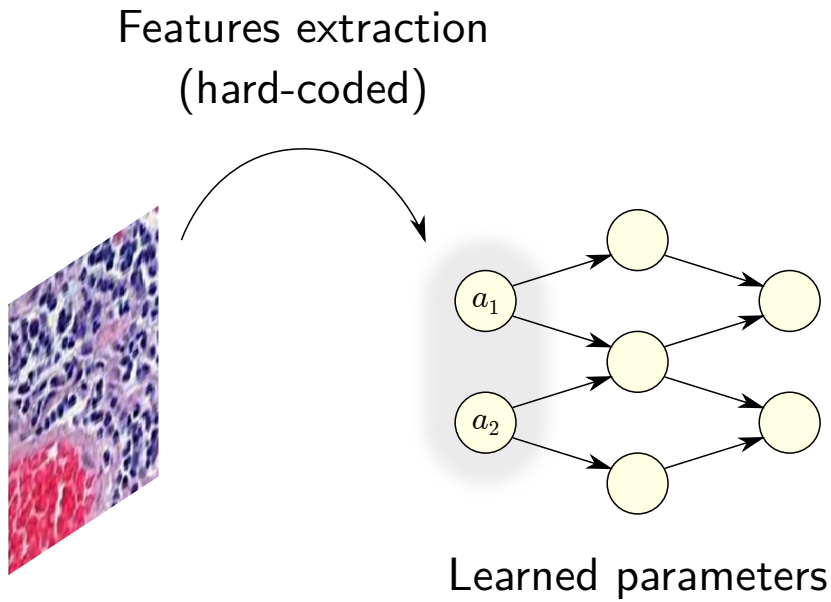
Interpretation:

- Model complexity can be restrained if the input data is low-dimensional **or** if the model builds a large margin (i.e. has low sensitivity).

Question:

- Can we build similar concepts for deep neural networks?

Reducing Complexity via Low Dimensionality



Approach:

- First, generate a low-dimensional representation by extracting a few features from the high-dimensional input data (either hand-designed, or automatically generated using methods such as PCA).
- Then, learn a neural network on the resulting low-dimensional data.

Reducing Complexity via Low Dimensionality

Observations:

- Building low-dimensional representations can be useful when predicting noisy high-dimensional data such as gene expression in biology ($d > 20000$)
- On other tasks such as *image recognition*, low dimensional representation can also delete class-relevant information (e.g. edges).

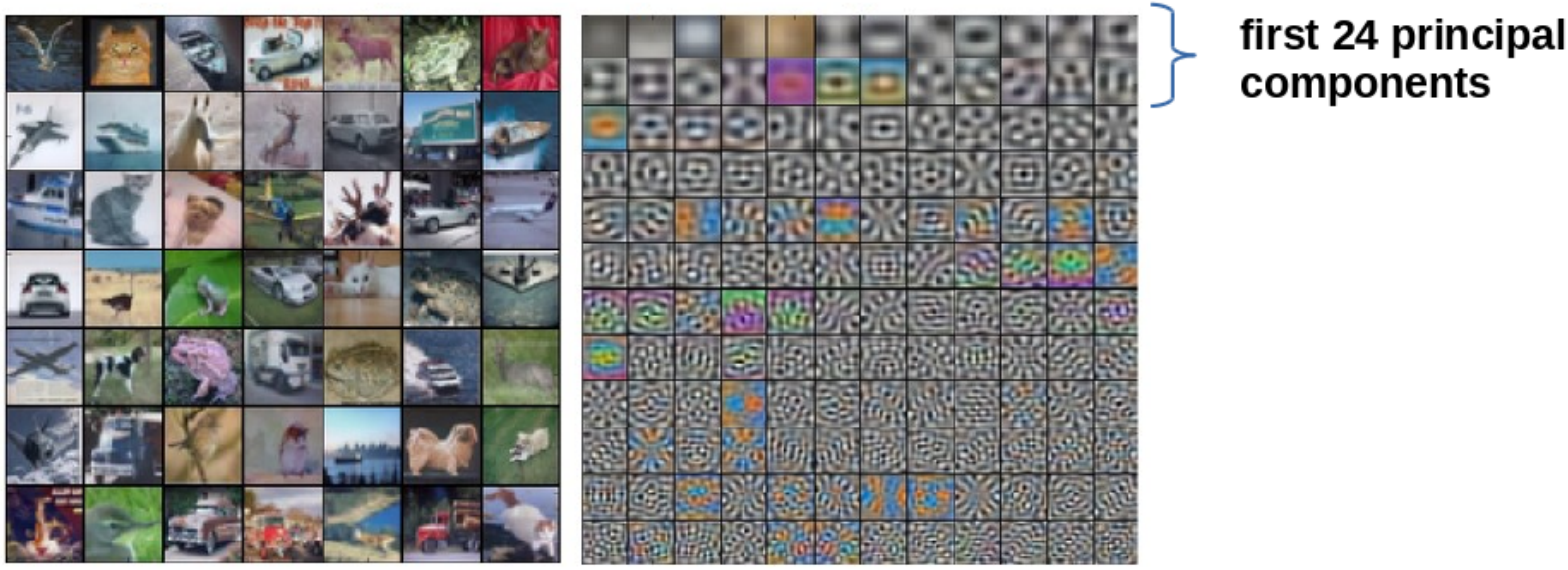
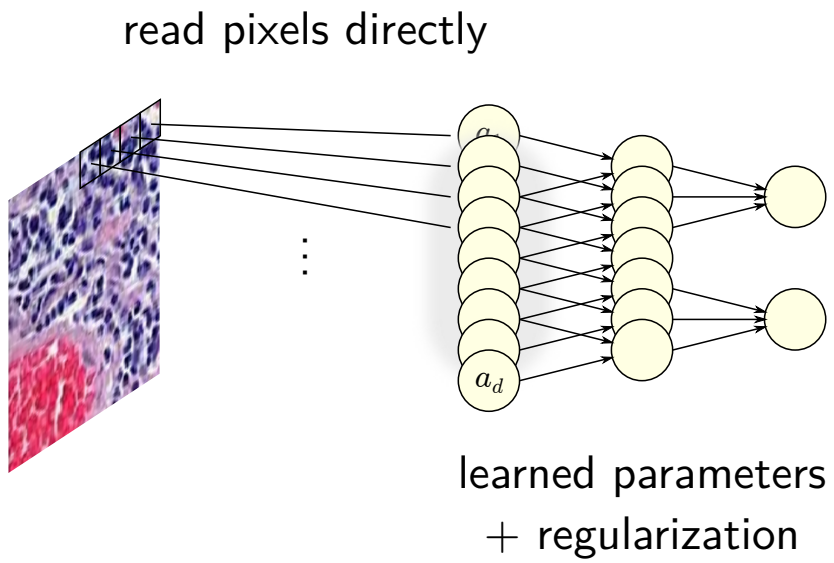


Image from Stanford CS class CS231n: Convolutional Neural Networks for Visual Recognition.

Reducing Complexity by Reducing Sensitivity



Weight Decay [4]:

- Include in the objective a term that makes the weights tend to zero if they are not necessary for the prediction task.

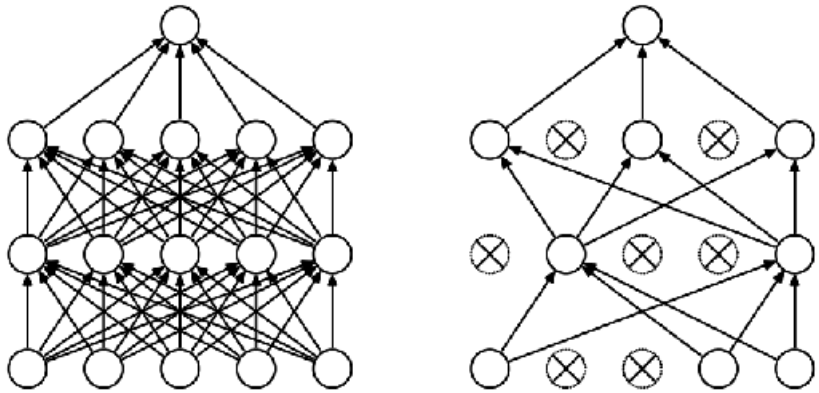
$$\mathcal{E}(\theta) = \sum_{i=1}^{N} (f(x_i, \theta) - t_i)^2 + \lambda \|\theta\|^2$$

- The higher the parameter λ, the more the exposure of the model to variations in the input domain is being reduced.

Reducing Complexity by Reducing Sensitivity

Dropout [7]:

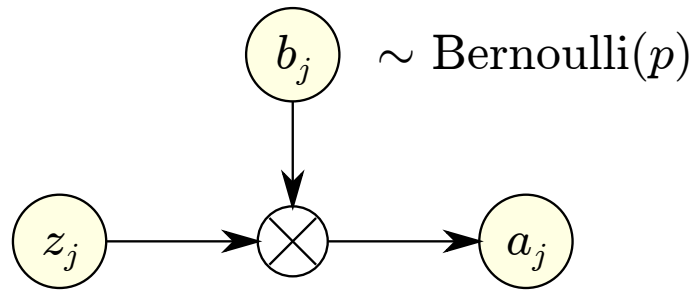
- Alternative to weight decay, which consists of adding artificial multiplicative noise to the input and intermediate neurons, and training the model subject to that noise.



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

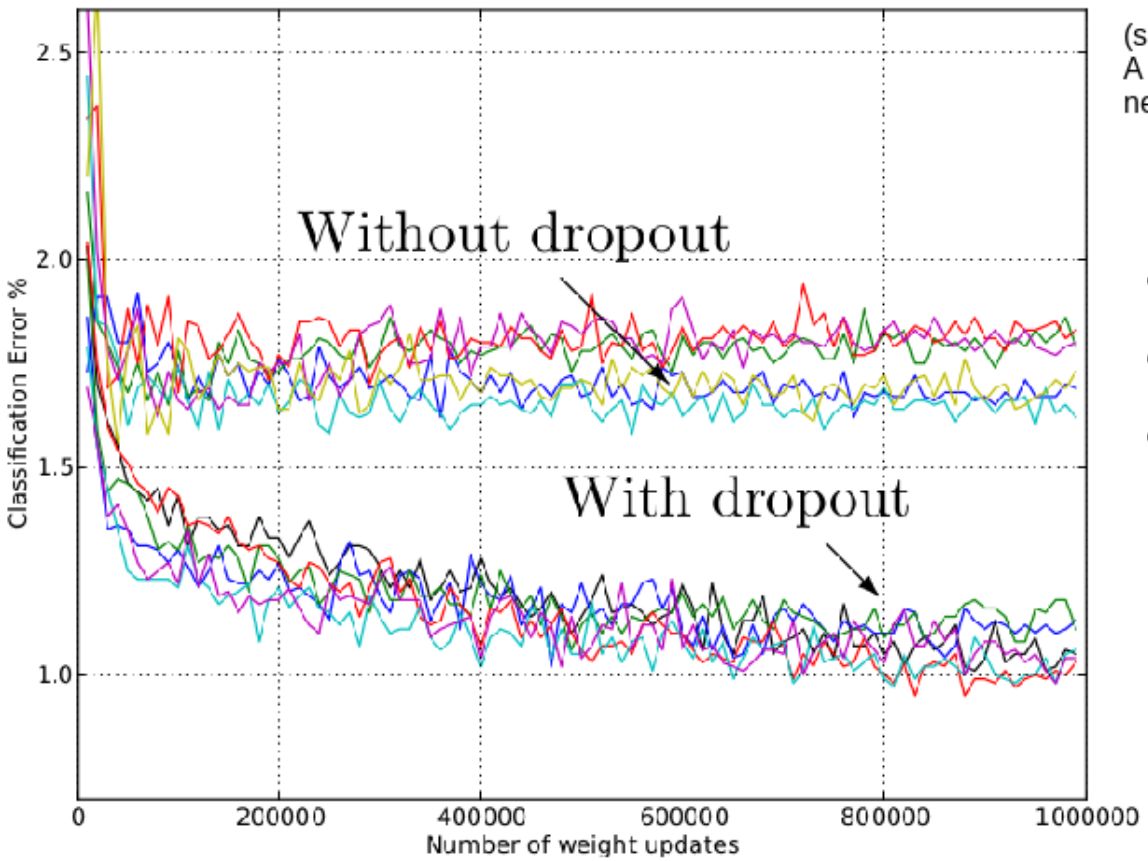
(Source: Srivastava et al. 2014: Dropout: A simple way to prevent neural networks from overfitting).

- This is achieved by inserting a "dropout" layer in the neural network, which multiplies each input (or activation) by a random variable $b_j \sim \text{Bernoulli}(p)$:

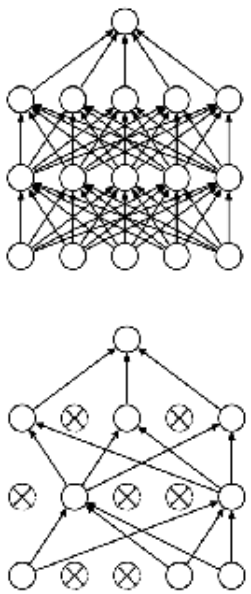


Reducing Complexity by Reducing Sensitivity

Effect of dropout on performance on the MNIST dataset



(source: Srivastava et al. 2014: Dropout: A simple way to prevent neural networks from overfitting).



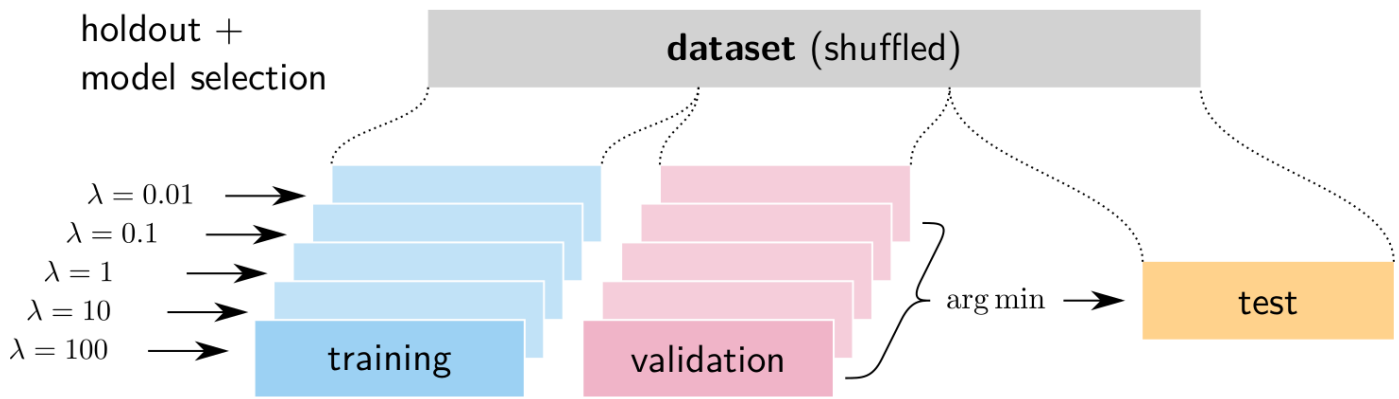
Note:

- On neural networks for image data, dropout tends to yield superior performance compared to simple weight decay.

Choosing a Model with Appropriate Complexity

Holdout Validation:

- Train multiple neural network models with different regularization parameters (e.g. λ), and retain the one that performs the best on some validation set disjoint from the training data.



Problem:

- Training a model for each parameter λ can be costly. One would potentially benefit from training a bigger model only once.

Accelerating Model Selection

Early Stopping Technique [6]:

- View the iterative procedure for training a neural network as generating a sequence of increasingly complex models $\theta_1, \ldots, \theta_T$.
- Monitor the validation error throughout training and keep a snapshot of the model when it had lowest validation error.

Early stopping:

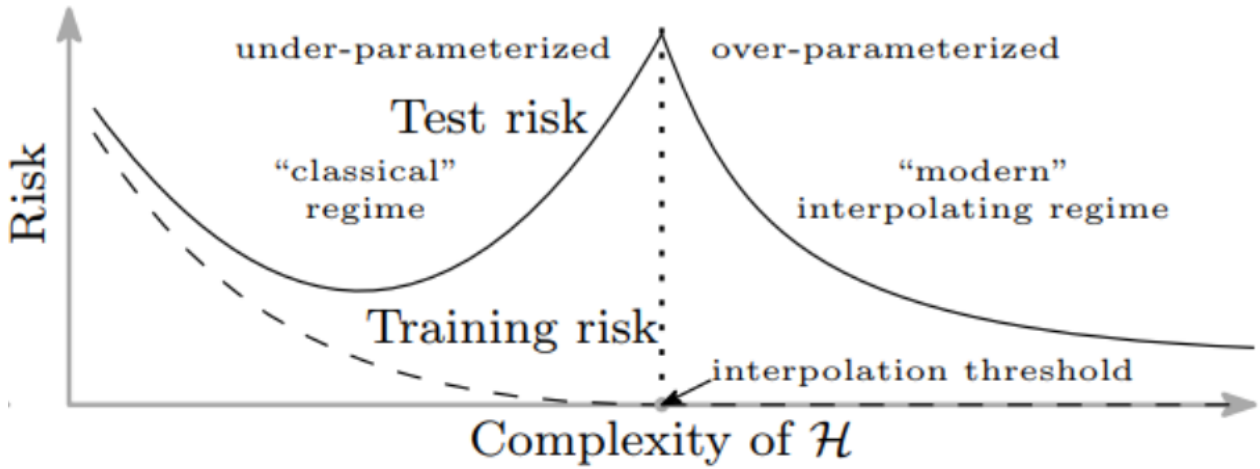
$\theta^\star = \texttt{None}$
$\mathcal{E}^\star = \infty$
for $t = 1 \ldots T$ **do**
  Run a few SGD steps, and collect the current parameter θ_t
  if $\mathcal{E}_{\text{val}}(\theta_t) < \mathcal{E}^\star$ **then**
    $\theta^\star \leftarrow \theta_t$
    $\mathcal{E}^\star \leftarrow \mathcal{E}_{\text{val}}(\theta)$
  end if
end for

Advantage:

- No need to train several models (e.g. with different λ's). Only one training run is needed!

Very Large Models

- When the model becomes very large there is an interesting '*double descent*' [2] phenomenon that occurs in the context of neural networks, where the generalization error starts to go down again as model complexity increases.
- This can be interpreted as some implicit averaging between the many components of the model (interpolating regime).



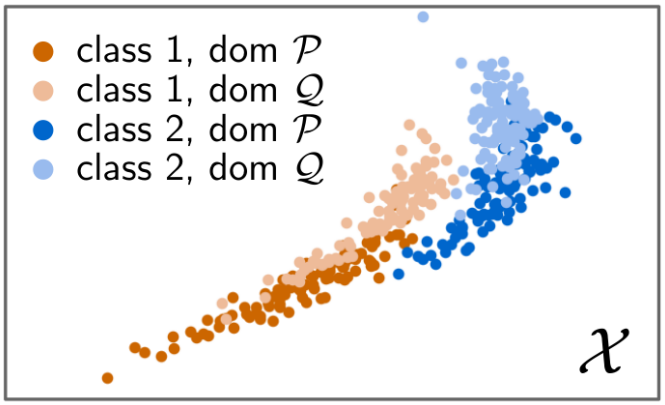
- Increasing model size to a great extent may contribute, without further regularization techniques to achieve lower test set error.

Part 2

Imbalances between Subgroups

Data from Multiple Domains

- The data might come from different domains ($\mathcal{P}$, $\mathcal{Q}$).
- Domains may e.g. correspond to different acquisition devices, or different ways they are configured/calibrated.
- One of the domains may be overrepresented in the available data, or the ML model may learn better on a given domain at the expense of another domain.



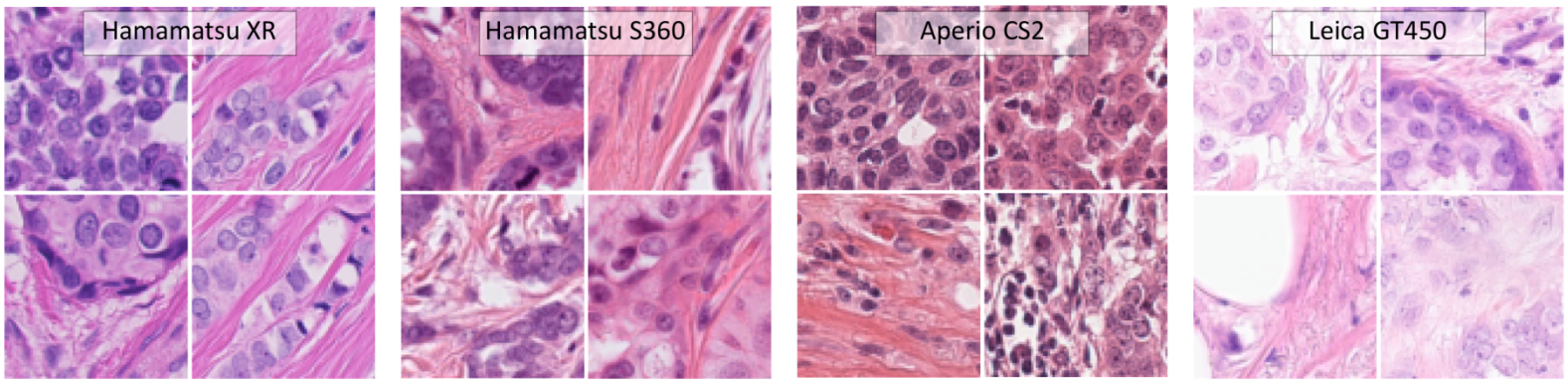
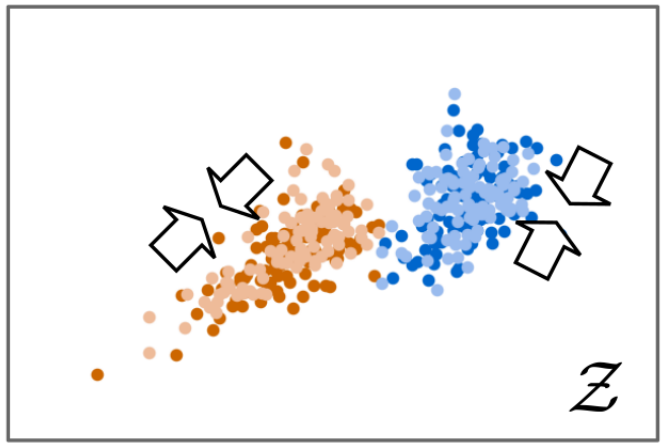


Image source: Aubreville et al. Quantifying the Scanner-Induced Domain Gap in Mitosis Detection. CoRR abs/2103.16515 (2021)

Addressing Multiple Domains



Simple Approach (one-layer networks):

- Denoting by $\mathcal{P}$ and $\mathcal{Q}$ the distribution of the two domains, regularize the ML model ($\boldsymbol{w}^\top \boldsymbol{x}$) so that both domains generate the same responses on average at the output:

$$\min_{\boldsymbol{w}} \quad \mathcal{E}(\boldsymbol{w}) + \lambda \cdot \left(\mathbb{E}_{\mathcal{P}}[\boldsymbol{w}^\top \boldsymbol{x}] - \mathbb{E}_{\mathcal{Q}}[\boldsymbol{w}^\top \boldsymbol{x}]\right)^2$$

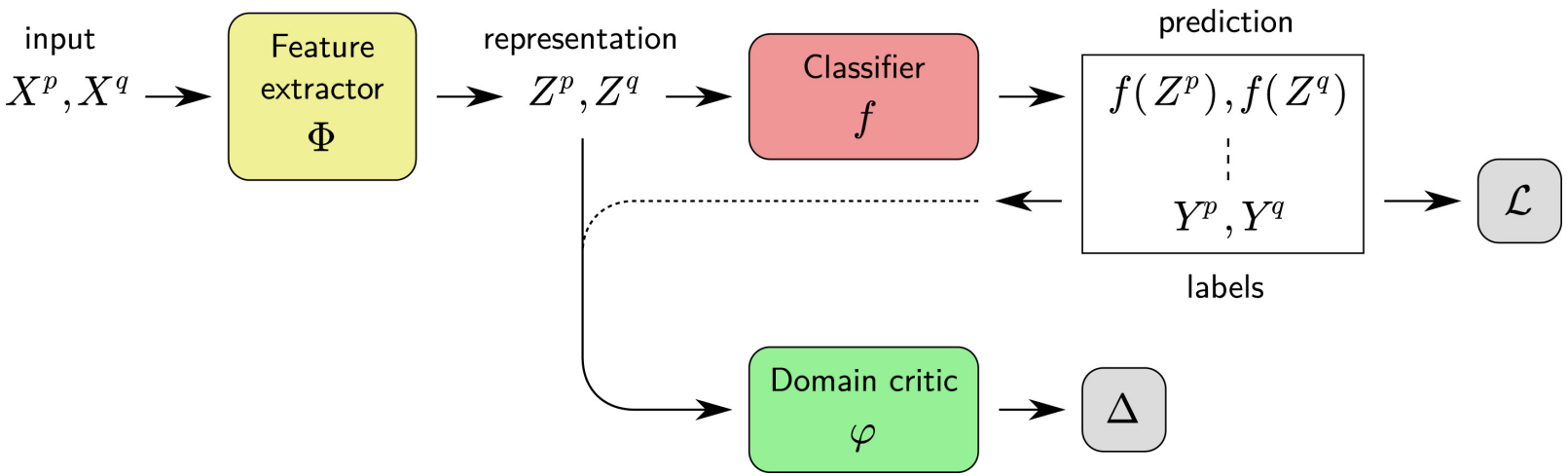
  (aka. moment matching). The approach can be enhanced to include higher-order moments such as variance, etc.

- In practice, more powerful tools exist to constrain distributions more finely in representation space, such as the Wasserstein distance.

Addressing Multiple Domains

More Advanced Approach [1]:

- Learn a auxiliary neural network (domain critic φ) that tries to classify the two domains. Learn the parameters of the feature extractor in a way that the domain critic φ is no longer able to distinguish between the two domains.



Addressing Multiple Domains

Example:

- Example of one particular class of the Office-Caltech dataset and the different domains from which the data is taken.

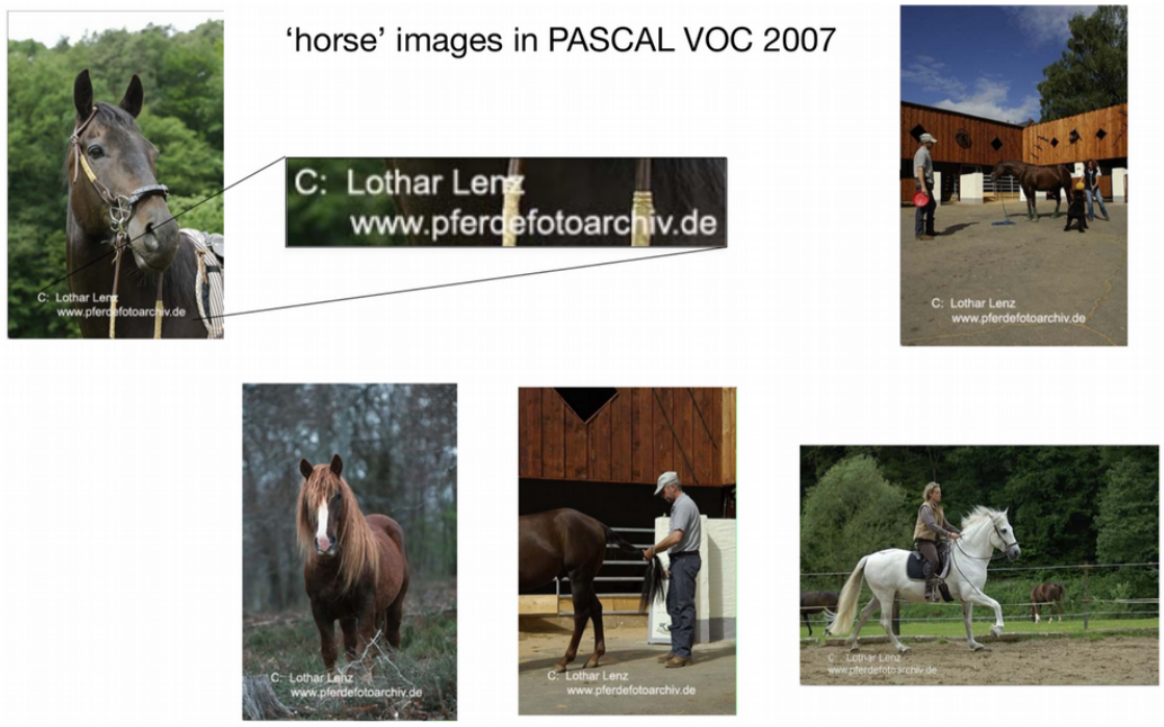
- Models equiped with a domain critic, although loosing performance on some domains, achieve better worst-case accuracy:

	Amazon	*Caltech*	*DSLR*	*Webcam*	Avg	Min
No critic	*90.63*	*84.27*	*93.75*	*98.31*	91.74	84.27
Marginal critic	*91.32*	*85.46*	*92.71*	*96.07*	91.39	85.46
Joint critic (Ours)	*91.67*	*86.05*	*93.75*	*97.00*	**92.12**	**86.05**

Part 3

Spurious Correlations

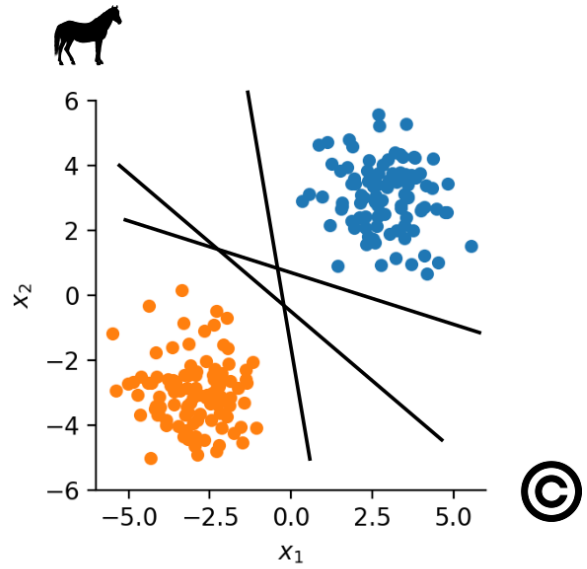
Spurious Correlations



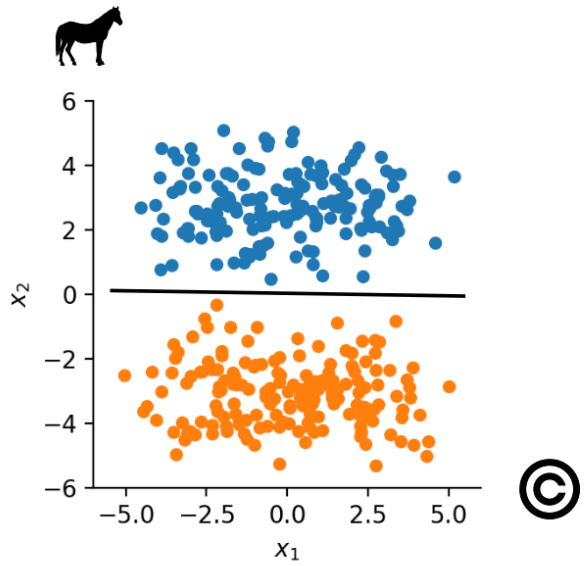
- Artefact of the distribution of available data ($\mathcal{P}$) where one or several task-irrelevant input variables are spuriously correlated to the task-relevant variables.
- Spurious correlations are very common on practical datasets, e.g. a copyright tag occurring only on images of a certain class; histopathological images of a certain class having been acquired with particular device and having as a result a different color profile, etc.

Spurious Correlations and the Clever Hans Effect

Available data ($\mathcal{P}$)



New data ($\mathcal{Q}$)



- A ML classifier is technically able to classify the available data ($\mathcal{P}$) using either the correct features or the spurious ones. The ML model doesn't know a priori which feature (the correct one or the spurious one) to use. A model that bases its decision strategy on the spurious feature is "*right for the wrong reasons*" and is also known as a *Clever Hans classifier*.
- A Clever Hans classifier may fail to function well on the new data ($\mathcal{Q}$) where the spurious correlation no longer exists, e.g. horses without copyright tags, or images of different class with copyright tags.

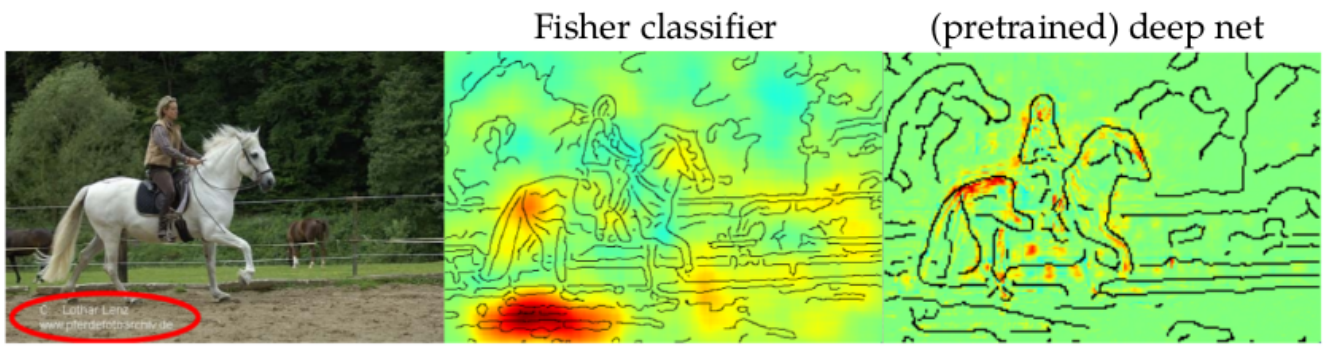
Spurious Correlations and the Clever Hans Effect

- Test set accuracy doesn't give much information on whether the model bases its decision on the correct features or exploits the spurious correlation.

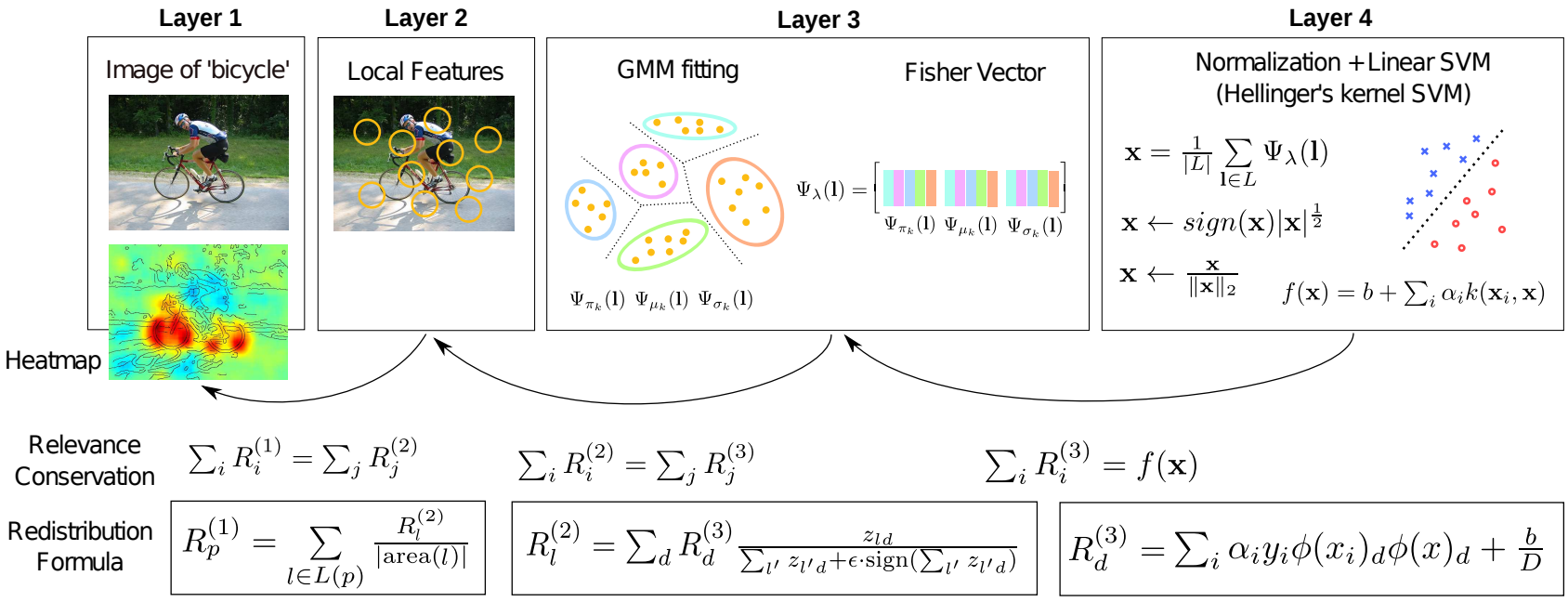
Fisher Vector Classifier vs. DeepNet pretrained on ImageNet

	aeroplane	bicycle	bird	boat	bottle	bus	car
Fisher	79.08%	66.44%	45.90%	70.88%	27.64%	69.67%	80.96%
DeepNet	88.08%	79.69%	80.77%	77.20%	35.48%	72.71%	86.30%
	cat	chair	cow	diningtable	dog	horse	motorbike
Fisher	59.92%	51.92%	47.60%	58.06%	42.28%	80.45%	69.34%
DeepNet	81.10%	51.04%	61.10%	64.62%	76.17%	81.60%	79.33%
	person	pottedplant	sheep	sofa	train	tvmonitor	mAP
Fisher	85.10%	28.62%	49.58%	49.31%	82.71%	54.33%	59.99%
DeepNet	92.43%	49.99%	74.04%	49.48%	87.07%	67.08%	72.12%

- Only an inspection of the decision structure by the user (e.g. using LRP heatmaps) enables the detection of the flaw in the model [5].



Generating LRP heatmaps



- Explanations are produced using a layer-wise redistribution process from the output of the model to the input features.
- Each layer can have its own redistribution scheme. The redistribution rules are designed in a way that maximizes explanation quality.

Mitigating Reliance on Spurious Correlations

Feature Selection / Unlearning:

- Retrain without the feature containing the artefact (e.g. crop images to avoid copyright tags).
- Actively look in the model for units (e.g. subsets of neurons) that respond to the artifact and remove such units from the model (e.g. [3]).

Dataset Design:

- Manually remove the artifact (e.g. copyright tags) from the classes that contain it, or alternatively, inject the artifact in every class (so that it cannot be used anymore for discriminating between classes).
- Stratify the dataset in a way that the spurious features are present in all classes in similar proportions.

Learn with Explanation Constraints

- Include an extra term in the objective that penalizes decision strategies that are based on unwanted features (preliminary revealed by an explanation technique).

Summary

Summary

- While deep learning can in principle fit very complex prediction functions, the way they perform in practice is in large part determined by the amount and quality of the data.
- Limited data may subject the ML model to **overfitting** and lead to lower performance on new data. Various methods exist to prevent overfitting (e.g. generating a low-dimensional input vector, or build a model with limited sensitivity to input such as weight decay or dropout).
- Another problem is **dataset bias**, where certain parts of the distribution are over/under-represented, or plagued with **spurious correlations**. Reliance of the model on spurious correlations can lead to low test performance, but this can be detected by Explainable AI approaches. A number of methods exist to reduce reliance on spurious correlations.

References

[1] L. Andéol, Y. Kawakami, Y. Wada, T. Kanamori, K.-R. Müller, and G. Montavon.
Learning domain invariant representations by joint wasserstein distance minimization.
Neural Networks, 167:233–243, 2023.

[2] M. Belkin, D. Hsu, S. Ma, and S. Mandal.
Reconciling modern machine-learning practice and the classical bias–variance trade-off.
PNAS, 116(32):15849–15854, 2019.

[3] P. Chormai, J. Herrmann, K. Müller, and G. Montavon.
Disentangled explanations of neural network predictions by finding relevant subspaces.
IEEE Trans. Pattern Anal. Mach. Intell., 46(11):7283–7299, 2024.

[4] A. Krogh and J. A. Hertz.
A simple weight decay can improve generalization.
In *NIPS*, pages 950–957, 1991.

[5] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller.
Unmasking clever hans predictors and assessing what machines really learn.
Nature Communications, 10(1), Mar. 2019.

[6] L. Prechelt.
Early stopping-but when?
In *Neural Networks: Tricks of the Trade*, volume 1524 of *LNCS*, pages 55–69. Springer, 1996.

[7] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov.
Dropout: a simple way to prevent neural networks from overfitting.
J. Mach. Learn. Res., 15(1):1929–1958, 2014.