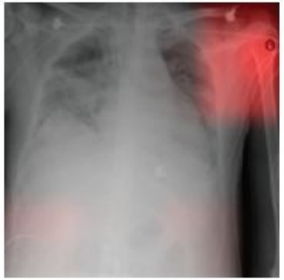
Shortcut learning in deep neural networks

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge & Felix A. Wichmann

Nature Machine Intelligence **2**, 665–673 (2020) | Cite this article



Article: Super Bowl 50

Paragraph: *"Peython Manning became the first quarterback ever to lead two different teams to multiple Super Bowls. He is also the oldest quarterback ever to play in a Super Bowl at age 39. The past record was held by John Elway, who led the Broncos to victory in Super Bowl XXXIII at age 38 and is currently Denver's Executive Vice President of Football Operations and General Manager. Quarterback Jeff Dean had a jersey number 37 in Champ Bowl XXXIV."*

Question: *"What is the name of the quarterback who was 38 in Super Bowl XXXIII?"*

Original Prediction: John Elway

Prediction under adversary: Jeff Dean

Task for DNN	Caption image	Recognise object	Recognise pneumonia	Answer question
Problem	Describes green hillside as grazing sheep	Hallucinates teapot if certain patterns are present	Fails on scans from new hospitals	Changes answer if irrelevant information is added
Shortcut	Uses background to recognise primary object	Uses features irrecognisable to humans	Looks at hospital token, not lung	Only looks at last sentence and ignores context

https://www.nature.com/articles/s42256-020-00257-z

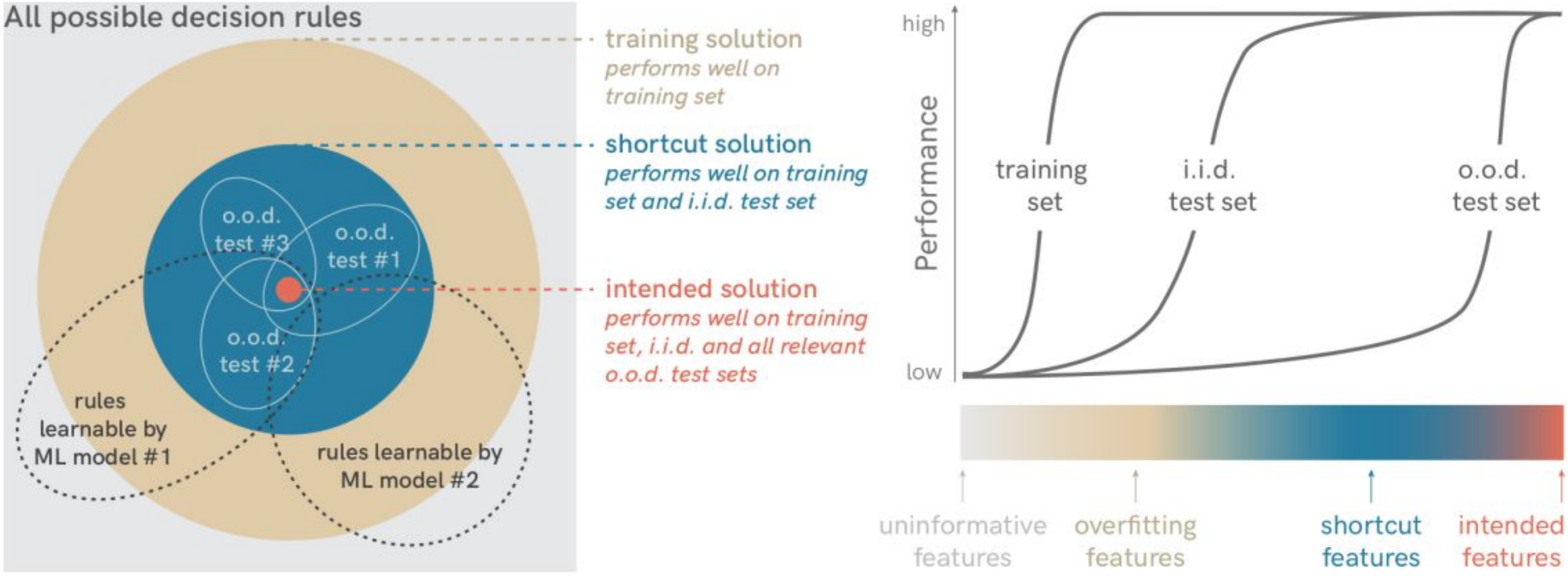


Figure 3. Taxonomy of decision rules. Among the set of all possible rules, only some solve the training data. Among the solutions that solve the training data, only some generalise to an i.i.d. test set. Among those solutions, shortcuts fail to generalise to different data (o.o.d. test sets), but the intended solution does generalise.

https://www.nature.com/articles/s42256-020-00257-z

Figure 4. Both human and machine vision generalise, but they generalise very differently. Left: image pairs that belong to the same category for humans, but not for DNNs. Right: images pairs assigned to the same category by a variety of DNNs, but not by humans.

https://www.nature.com/articles/s42256-020-00257-z

Data-centric AI

- a buzzword that addresses an important phenomenon: most time spent on AI implementation goes on data gathering and preparation
- ML theory tends to focus on choosing models and their hyperparameters
- in some cases, improved performance might be easier to achieve by working on the dataset

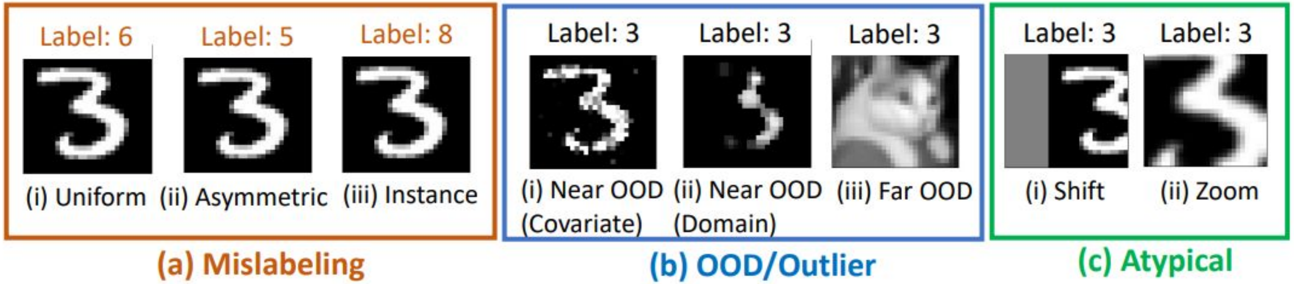
- data-centric approach typically boils down to outlier removal, detection of mislabeled samples, guiding additional data gathering (active training), data augmentation and EDA

CIFAR-10 given label:
cat

https://labelerrors.com/

Example difficulty

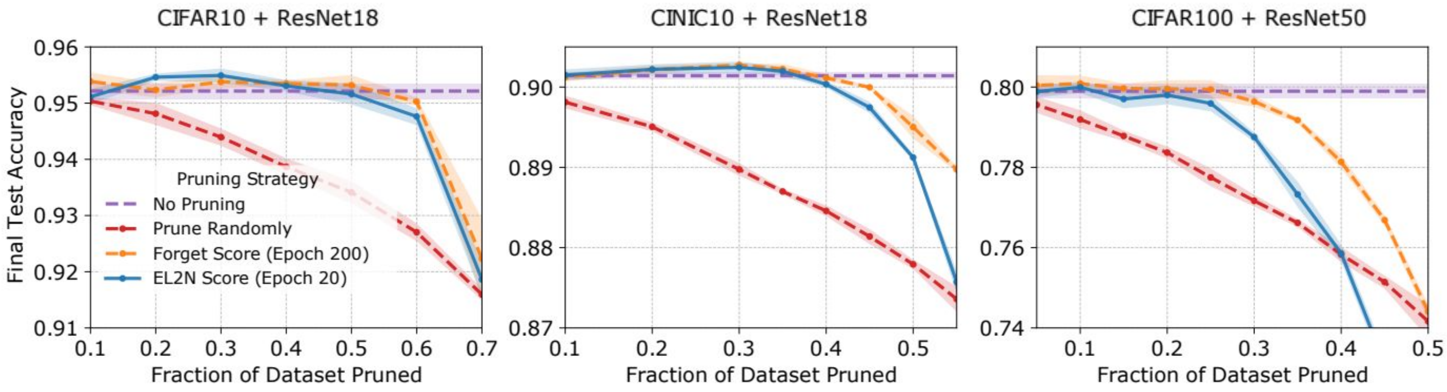
- a trained model can be used in EDA
- samples can be ranked by their "hardness"
- three types of "hardness" are defined in the paper below
- most difficult examples often significantly affect learning
- a substantial fraction of easy examples can be discarded



Dissecting Sample Hardness: A Fine-Grained Analysis of Hardness Characterization Methods for Data-Centric AI (2024)

Some training examples can be discarded

- balancing the dataset goes beyond classes: some examples are so similar or misleading that they don't contribute to the test performance
- two importance measurements are used below (samples that weren't forgotten during training and those that barely affect loss gradient)
- can lead to an improvement, especially when label noise is present

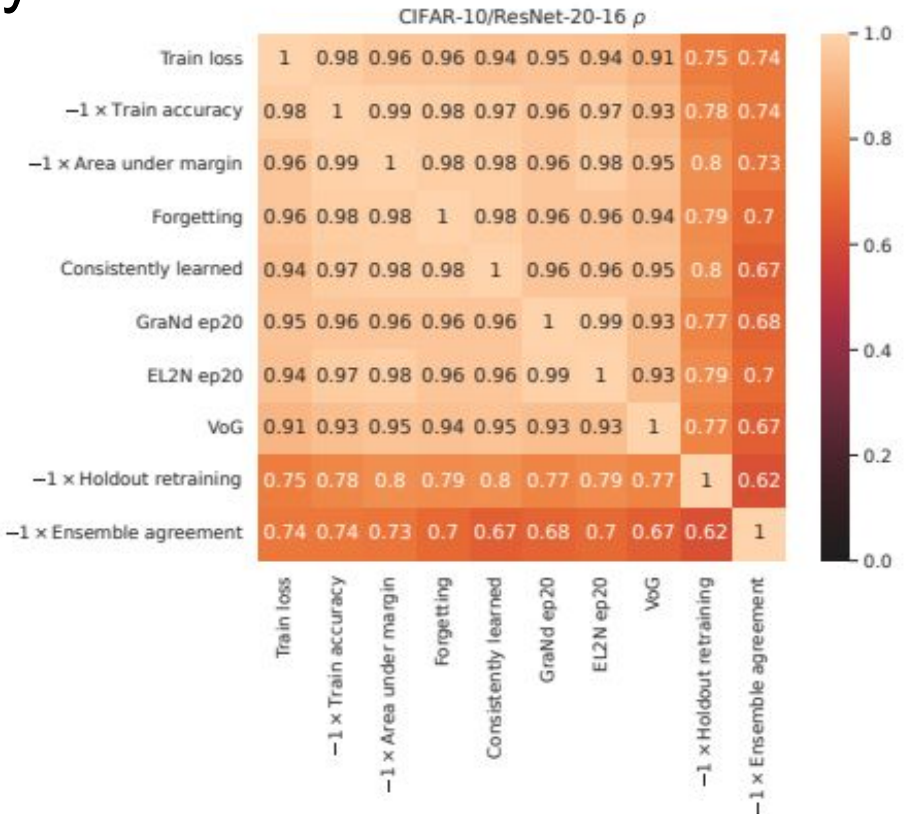


Deep Learning on a Data Diet: Finding Important Examples Early in Training (2023)

Measures of example difficulty

- lots of sample difficulty scores exist, most of them well-correlated
- examples: average training loss, difference between two highest output probabilities, variance of pixels' gradients, number of times a sample is forgotten when training
- most difficulty measures are highly sensitive to random initialization of the model; we need multiple training runs

Spearman (rank) correlation between difficulty scores



Dataset Difficulty and the Role of Inductive Bias (2024)

Different models find different samples difficult

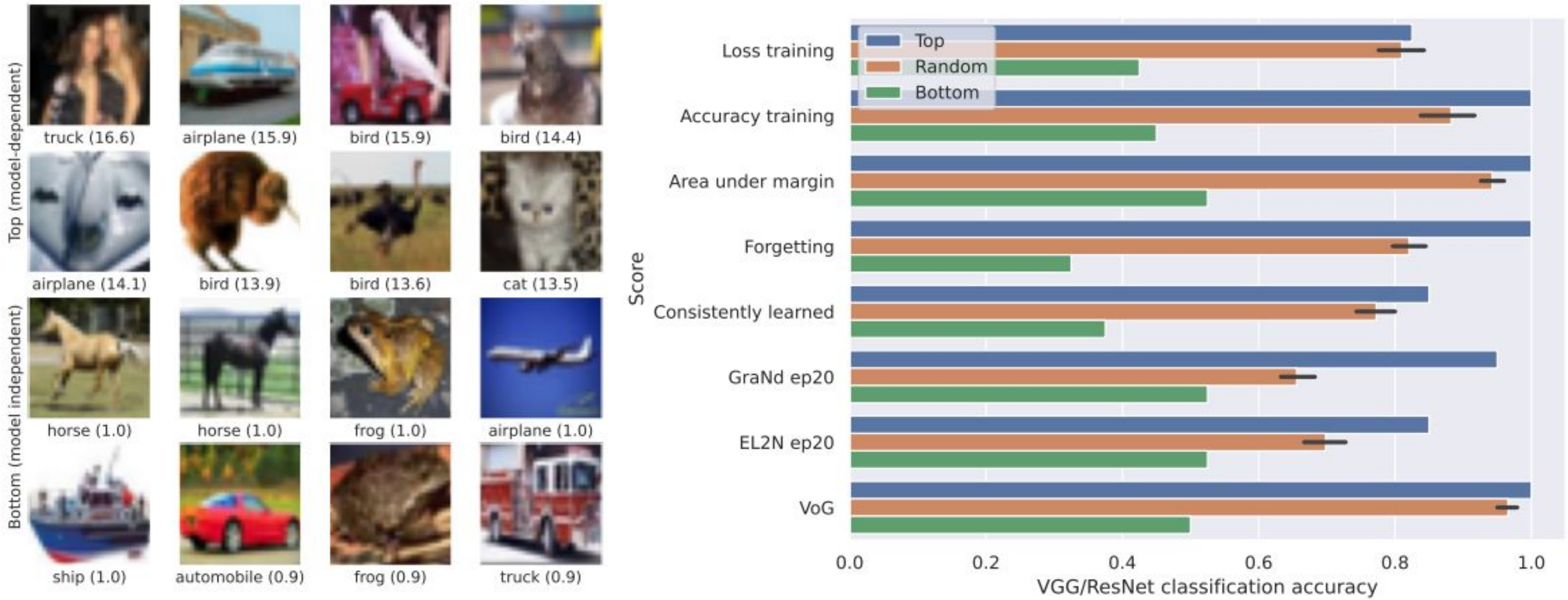


Figure 1: **Left:** Top 8 CIFAR-10 examples most/least sensitive to inductive biases (model width, depth, and architecture), ranked by statistical significance of changes to difficulty score for pairwise model comparisons (see Section 4.3 for more details). Numbers indicate mean $-\log P$ value of each example. **Right:** Distinguishing between VGG and ResNet using the top 8 examples (Left) as features for logistic regression.

Prediction depth

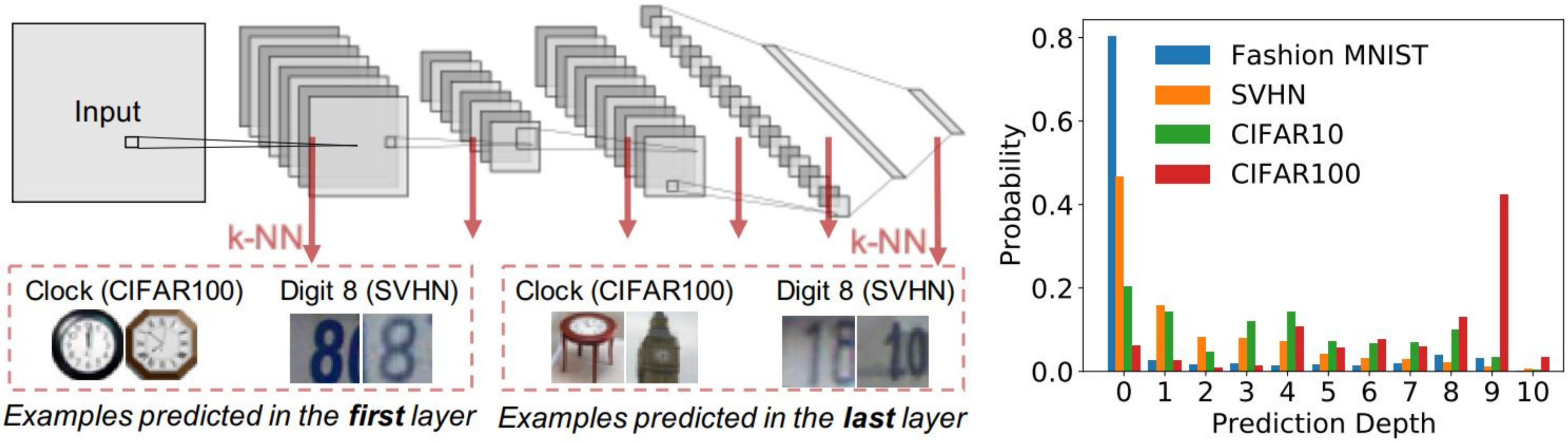


Figure 1: *Deep models use fewer layers to (effectively) determine the prediction for easy examples and more layers for hard examples.* **Left:** A cartoon illustrating the definition of prediction depth (given in Section 2.1).

Deep Learning Through the Lens of Example Difficulty (2021)

Iteration learned and prediction depth are correlated

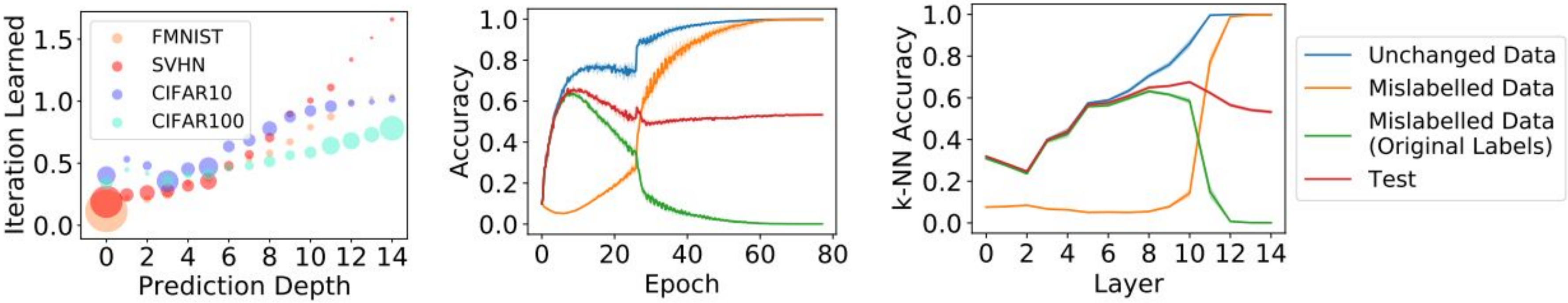


Figure 5: **Left:** *Data points with small prediction depths are on average learned before data points with higher prediction depths.* We train 250 VGG16 models for each dataset, using a 90:10% random train:validation split

Deep Learning Through the Lens of Example Difficulty (2021)

Using prediction depth to divide the dataset

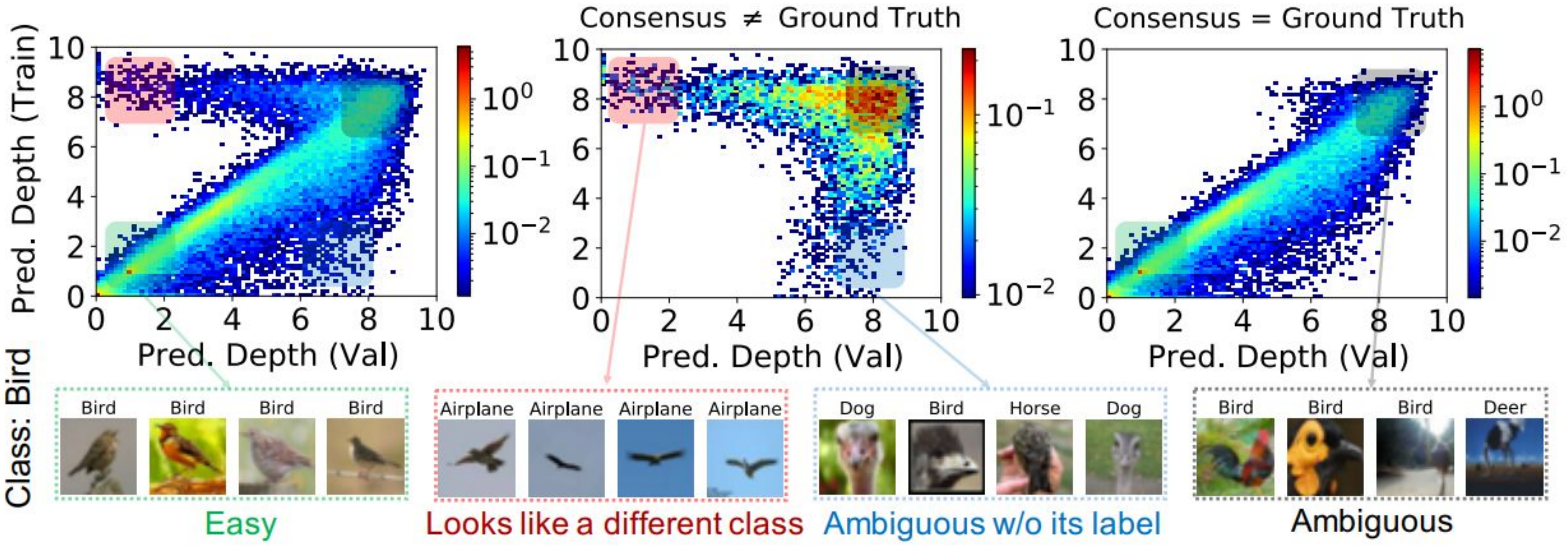


Figure 7: *The prediction depth can be the same, or very different for the same input when it occurs in the train and validation splits. Corners of this plot correspond to different forms of example difficulty.* (See Section 4 for discussion.)

Deep Learning Through the Lens of Example Difficulty (2021)

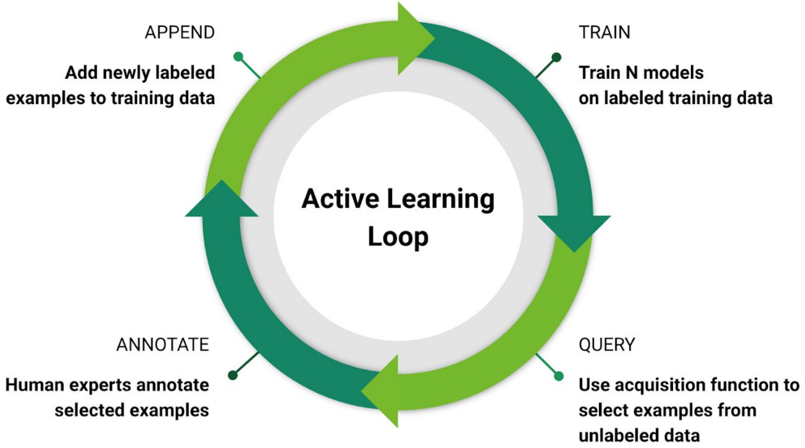
Code and experiments

https://github.com/JakubBilski/introduction-to-machine-learning/blob/main/2024/data_driven_ML.ipynb

https://api.wandb.ai/links/podcast-o-rybach-warsaw-university-of-technology/lu5teoet

Active Learning

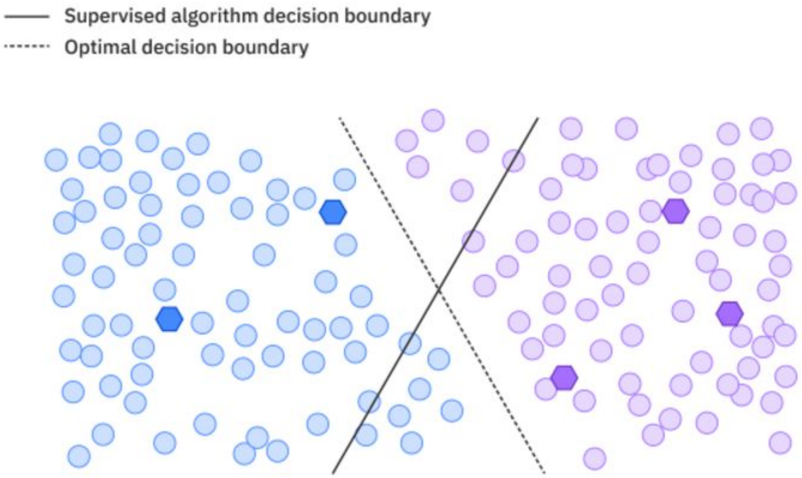
- starts with a small set of labeled data
- selects samples to be labeled next
- aims to cut down annotation costs
- selection methods:
  - uncertainty: most ambiguous samples
  - density-representative: centers of clusters
  - diversity-representative: most different samples
  - influence: tries to predict the impact on the performance
- see link below for a nice visualization



https://activelearner.fastforwardlabs.com/
https://www.cloudfactory.com/blog/active-learning-tackling-ml-data-annotation-challenges
A Survey on Deep Active Learning: Recent Advances and New Frontiers (2024)

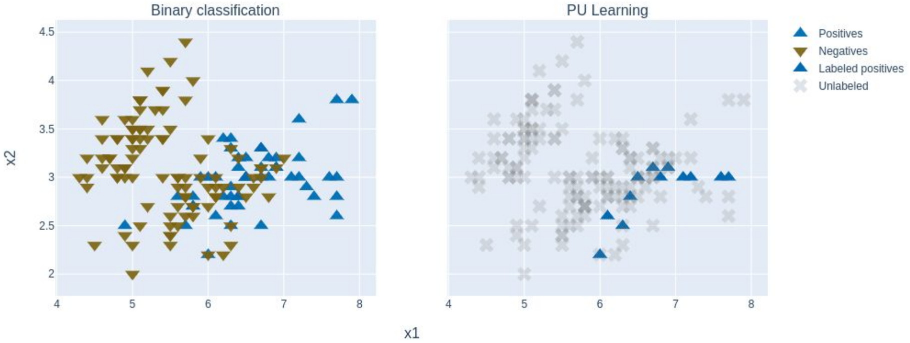
Semi-supervised Learning

- uses both labeled and unlabeled data
- relies on assumptions about the feature space
- smoothness assumption: points close to each other belong to the same class (transitivity can be used)
- low-density assumption: boundaries between classes are not densely populated
- manifold assumption: there's a low-dimensional view that allows to represent every class



https://www.ibm.com/topics/semi-supervised-learning

Positive Unlabeled Learning



- an extreme case of binary semi-supervised learning where only the positive class contains labeled examples
- probability view: s(x) = e(x) y(x)
  y(x): probability that x is positive
  e(x): probability that a positive x is labeled
  s(x): probability that x is labeled
- we know s(x), we're interested in y(x)

- assumptions must be made to make the problem feasible and separate e(x) and y(x)
- implementation might depend on the model's ability to return accurate probabilities

https://techblog.ing.pl/blog/podstawy-positive-unlabeled-learning

Calibration

- are values returned by the model accurate estimates of the actual probabilities?
- a forgotten problem from ML (e.g. binary trees are terribly calibrated) that's been recently gaining in popularity

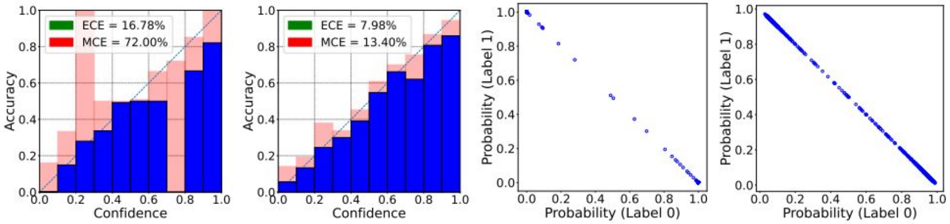
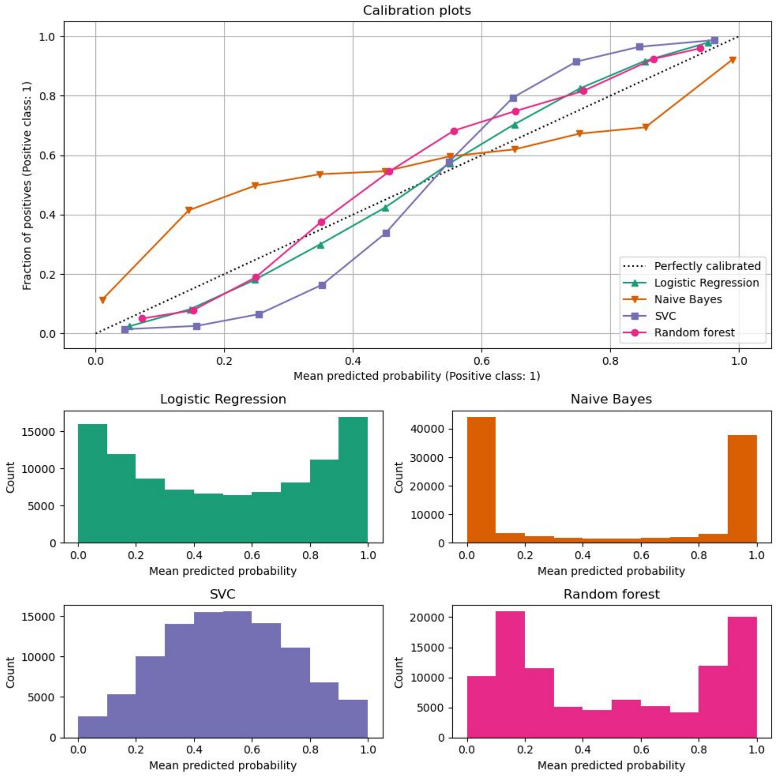


Figure 1: An illustration of model calibration. Uncalibrated model (left) that trained with standard cross-entropy loss and calibrated model (right) that trained with focal loss ($\gamma = 5$), have similar predictive performance on a binary classification task (accuracy is 83.8% and 83.4% respectively),



https://scikit-learn.org/1.5/modules/calibration.html
Calibration in Deep Learning: A Survey of the State-of-the-Art (2024)

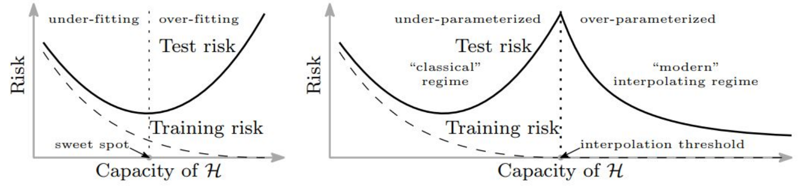
Bad calibration: example

should be more like 1% each; the game is already pretty much a draw

https://www.youtube.com/watch?v=mctyhQoK-tM

Deep neural networks are overparameterized

- … and yet, they generalize well
- intrinsic dimensions: a complexity measure for (model, dataset) pairs which estimates how many radom directions in the parameter space are enough to train the model
- for easier problems, it's possible to fit many classifiers into one network

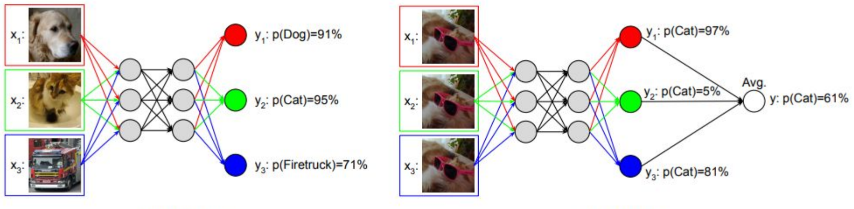


(a) (b)

Figure 1: **Curves for training risk (dashed line) and test risk (solid line).** (**a**) The classical *U-shaped risk curve* arising from the bias-variance trade-off. (**b**) The *double descent risk curve*, which incorporates the U-shaped risk curve (i.e., the "classical" regime) together with the observed behavior from using high capacity function classes (i.e., the "modern" interpolating regime), separated by the interpolation threshold. The predictors to the right of the interpolation threshold have zero training risk.

Dataset	**MNIST**		**MNIST (Shuf Pixels)**		**MNIST (Shuf Labels)**
Network Type	**FC**	**LeNet**	**FC**	**LeNet**	**FC**
Parameter Dim. D	199,210	44,426	199,210	44,426	959,610
Intrinsic Dim. d_{int90}	750	290	750	1,400	190,000

...	**CIFAR-10**		**ImageNet**	**Inverted Pendulum**	**Humanoid**	**Atari Pong**
...	**FC**	**LeNet**	**SqueezeNet**	**FC**	**FC**	**ConvNet**
...	656,810	62,006	1,248,424	562	166,673	1,005,974
...	9,000	2,900	> 500k	4	700	6,000

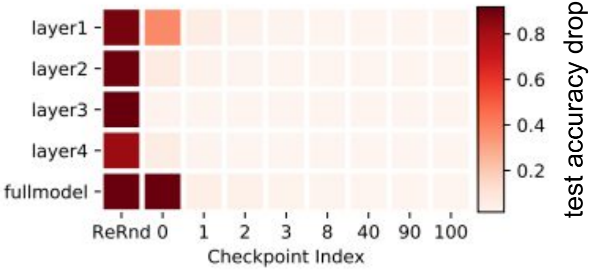


(a) Training (b) Testing

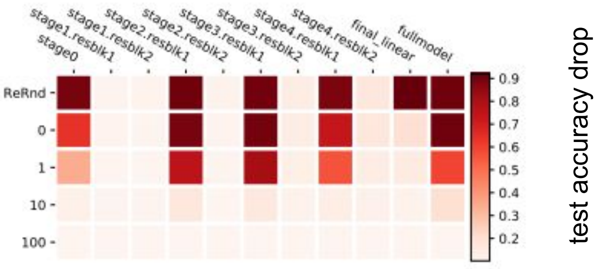
https://lilianweng.github.io/posts/2019-03-14-overfit/
Training Independent Subnetworks for Robust Prediction (2021)
Measuring the Intrinsic Dimension of Objective Landscapes (2018)
Reconciling modern machine learning practice and the bias-variance trade-off (2018)

Training is not evenly spread across layers

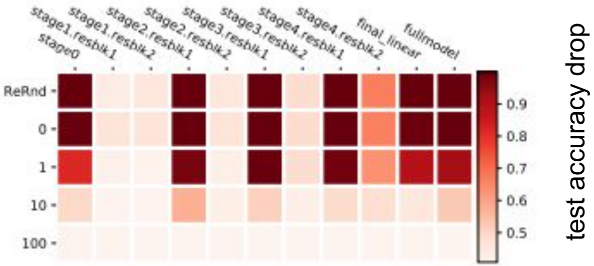
- reinitialization robustness: what happens if we set a trained model's layer to its historic state (e.g. 1st epoch) or randomize it ("ReRnd")
- only some layers are critical
- more difficult problem ≈ more critical layers



A small fully-connected network on MNIST



ResNet on CIFAR: residual blocks spread complexity



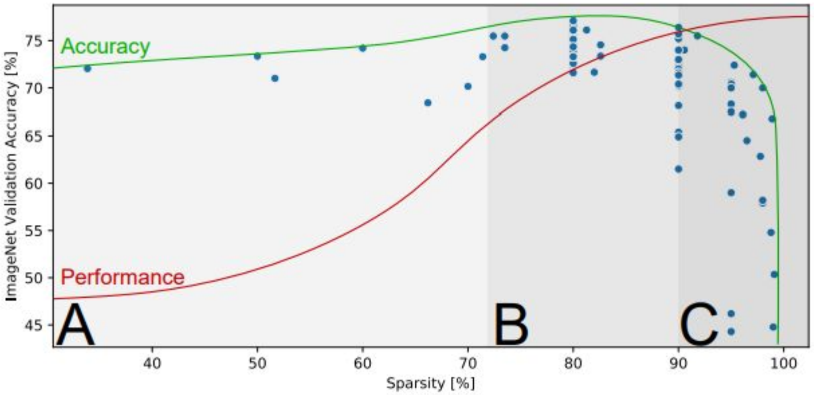
ResNet on ImageNet: more layers become critical

Are All Layers Created Equal? (2019)

Sparsity in neural networks

- most deep models are dense and over-parameterized
- because of that, they can memorize random patterns in the data (i.e. overfit)
- some parts of the network can be removed without a loss in accuracy
- this saves memory and time and might lead to better generalization

- example methods for model compression:
    - finding a smaller model (e.g. with NAS)
    - lowering precision/quantization
    - designing with sparser connections
    - pruning during or after training

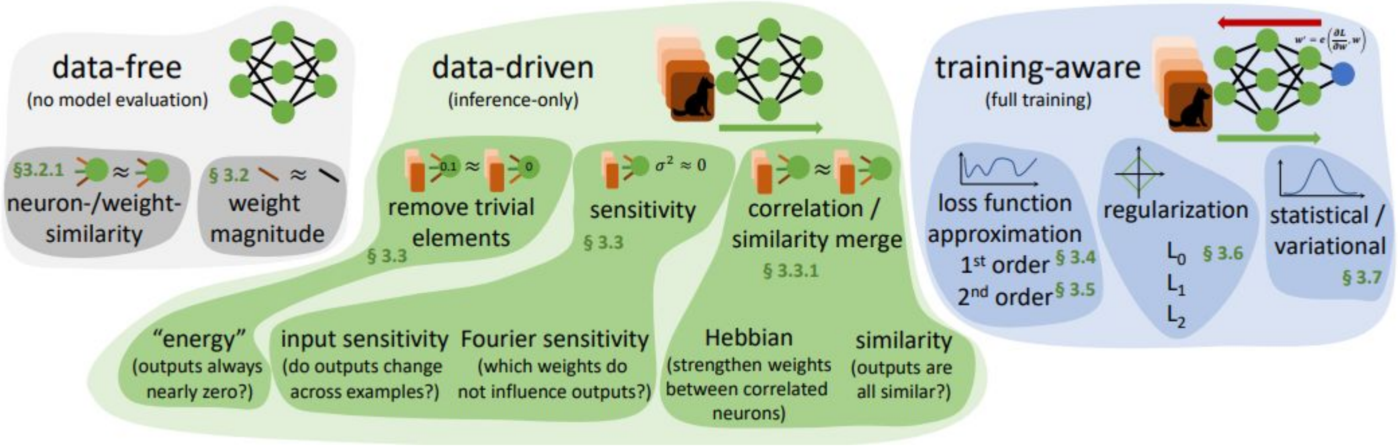


Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks (2021)

Pruning

- removing some connections
- can be implemented by zeroing out chosen weights
- many strategies for choosing targets

- in practice, real applications of pruning seem to be limited and target, if anything, mobile devices (where other improvements are also possible)



Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks (2021)

CNNs are sparse networks

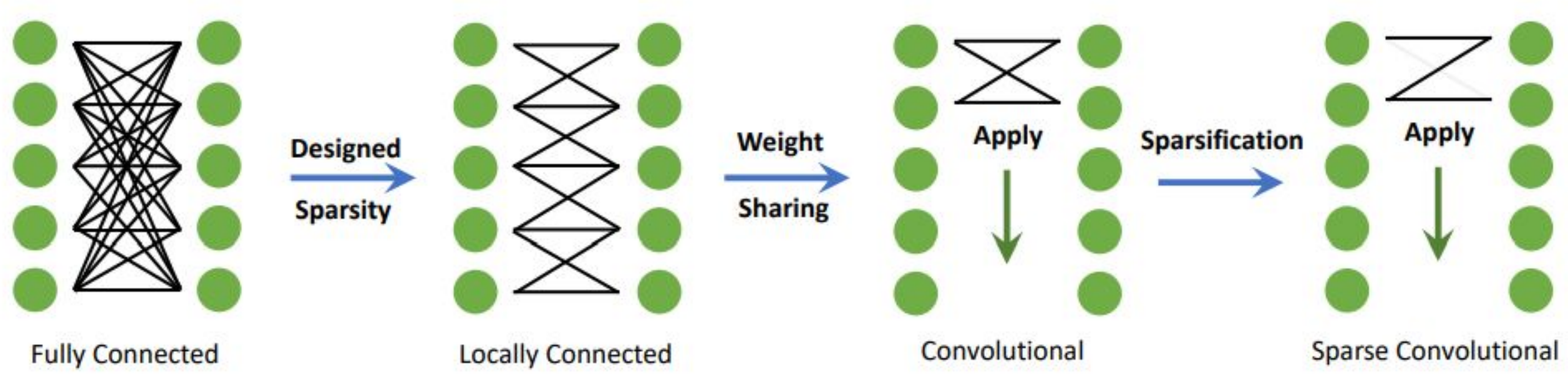
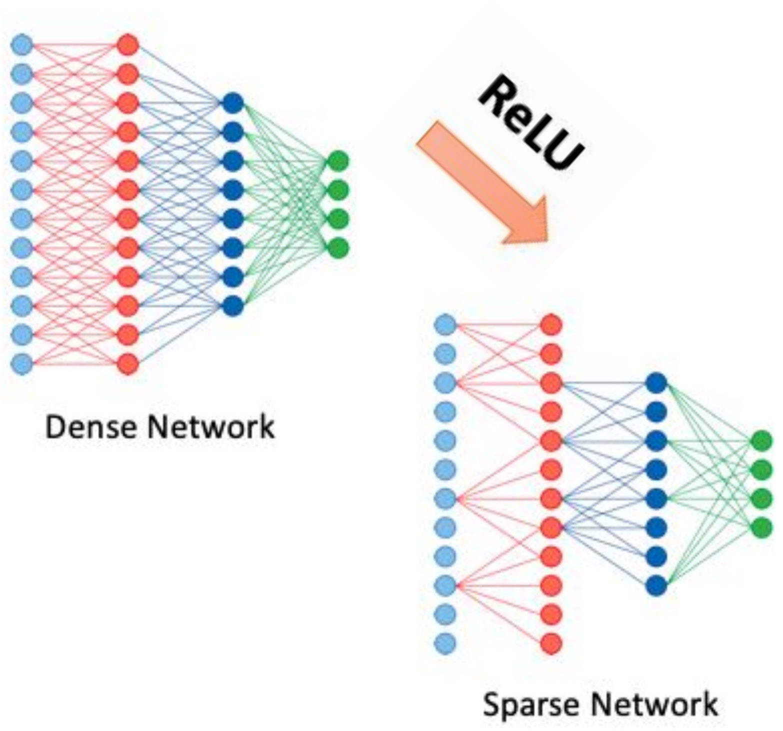


Fig. 3. Convolutional operators as sparse fully-connected operators for a single input and output channel.

Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks (2021)

ReLU creates sparse networks

- neurons with outputs zero are effectively removing their output connections for a particular sample
- gradient doesn’t flow through zeroed neurons; they don’t train
- a large part of the network can be inactive for individual samples
- this has been linked to generalization
- an uncommon situation where many neurons shut down for all samples is called a “dying ReLU”



https://cerebras.ai/chip/cerebras-wafer-scale-engine-why-we-need-big-chips-for-deep-learning/

Random convolutional filters

- feature extraction with random, non-trainable convolutional filters
- global pooling to get features from activations: avg, max, ppv (proportion of positive values)
- a generic classifier can be applied on those features to create a simple baseline model
- achieves state-of-the-art results in Time Series Classification

```python
class RandomConvFilters(nn.Module):
    def __init__(self, input_is_grayscale, num_filters):
        super(RandomConvFilters, self).__init__()
        self.conv1 = nn.Conv2d(
            1 if input_is_grayscale else 3,
            num_filters,
            3
        )
        self.conv1.requires_grad = False

    def forward(self, x):
        x = self.conv1(x)
        return (x>0).float().mean(dim=[2,3]) # proportion of positive values


class LogisticRegression(nn.Module):
    def __init__(self, num_inputs, num_classes):
        super(LogisticRegression, self).__init__()
        self.fc1 = nn.Linear(num_inputs, num_classes)

    def forward(self, x):
        x = self.fc1(x)
        return x
```

MINIROCKET: A Very Fast (Almost) Deterministic Transform for Time Series Classification (2020)