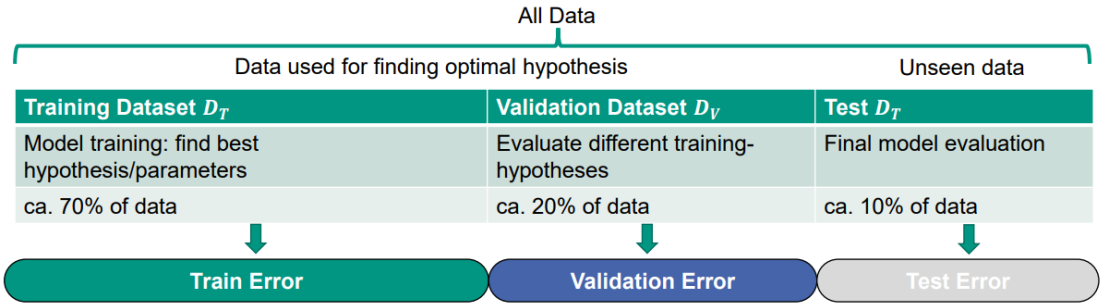
Inductive Lernen

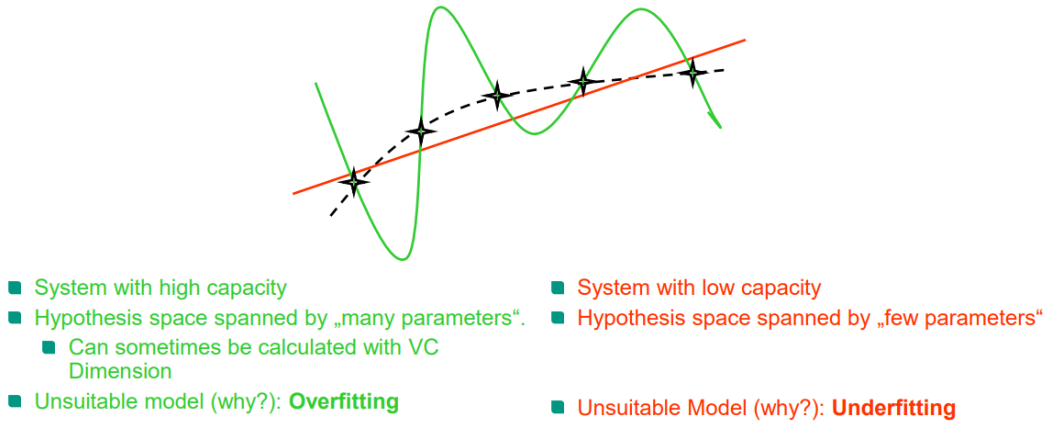
- Consistent：No negative examples are classified positive
  Complete：All positive examples are classified positive
- Inductive reasoning is a method of reasoning in which a general principle is derived from a body of specific observations.
- Deductive reasoning is a method of reasoning that starts with a general principle and and examines the possibility to reach a specific, logical conclusion.
- Inductive bias：Certain hypotheses are preferred over other hypotheses in the hypothesis space

Lerntheorie

- Occam‘s razor：Entities should not be multiplied beyond necessity/Oftentimes interpreted as: “other things being equal, simpler explanations are generally better than more complex ones"
- Empirical Risk / Empirical Error:

$$\hat{\mathcal{L}}_{\mathrm{D}}(h_\theta) = \mathbb{E}_{(x,y)\sim\hat{p}}[\ell(h_\theta(x), y)] = \frac{1}{|\mathrm{D}|}\sum_{(x,y)\in D}\ell(h_\theta(\mathrm{x}), y)$$

- Types of supervised Learning：Classification，Regression， Concept
- 

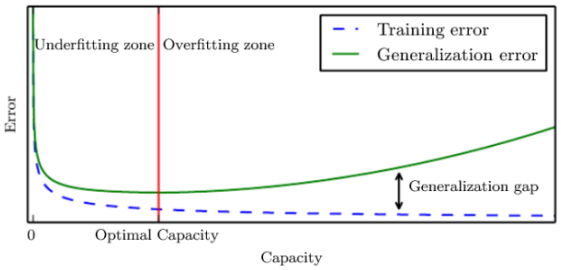
- Drei Kriterien hängt der Lernerfolg einer Lernmaschine ab: model complexity, amount of training data, generalization ability
- 

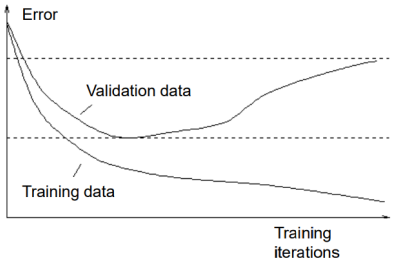
- Overfitting: : A hypothesis overfits the training examples, if some other hypothesis, that fits the training examples less well, actually performs better over the entire distribution of instances.

$$h \in H \text{ overfitting} \Leftrightarrow \exists h' \in H \text{ such that given } D_{Tr} \text{ and } D_V$$
$$\hat{\mathcal{L}}_{D_{Tr}}(h) < \hat{\mathcal{L}}_{D_{Tr}}(h') \wedge \hat{\mathcal{L}}_{D_V}(h) > \hat{\mathcal{L}}_{D_V}(h')$$

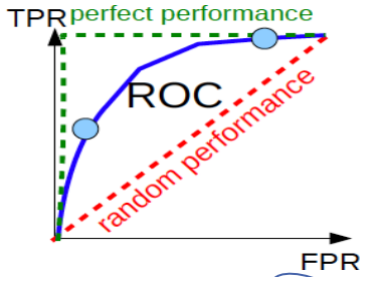
- Reasons for Overfitting: Model capacity is too large; Model is trained for too many

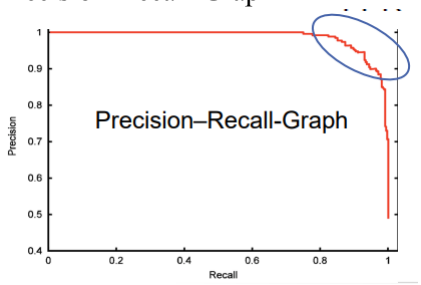
iterations.





- Solutions for Overfitting: Representative instances in training dataset (increase number and types of instances); Steer learning with validation error, e.g. early stopping; Correct choice and search for optimal hypothesis □θ; Decrease model capacity.
- Accuracy (as high as possible): $\text{Acc} = \frac{correct}{total} = \frac{TP + TN}{TP + FN + FP + TN}$
- False positive rate FPR, (as small as possible): $FPR = \frac{FP}{FP + TN}$
- Precision (as high as possible): $P = \frac{TP}{TP + FP}$
- Recall (as high as possible): $TPR = R = \frac{TP}{TP + FN}$
- Find Best Model of a Learning System: Approaches :TPR/FPR-Graph (Receiver-Operating Characteristic, ROC) Precision-Recall-Graph





- 定义(for linear classification):The VC-Dimension VC □$_\theta$ of H is equal to the maximum number of data points (from set S) which can be arbitrarily separated from H.
  如何计算 General: Hyperplane in R $n \Rightarrow n + 1$ separable values
  VC-Dimension: Measure of data complexity of learning
-

Unüberwachtes Lernen

- *Supervised Learning*：
  Data: Examples with input and desired output data (labeled data).
  Goal: Learn the relationship between input and output data in order to predict the correct output for new input data.
- *Unsupervised Learning*(Clustering &Dimensionality Reduction / Feature Extraction)：
  Data: Examples contain only input data (unlabeled data).
  Goal: Find the underlying structure in the data.
- *Reinforcement Learning :*
  Data: Experience from interaction with an environment and resulting reward.
  Goal: Learn a behavior that maximizes the reward in the long term.
- **K-Means**:
  Partition the training set into clusters $X_1, \dots X_k$ (with center points $\boldsymbol{c}_1, \dots, \boldsymbol{c}_k$) such that a minimum of distance between data and centroids is reached.

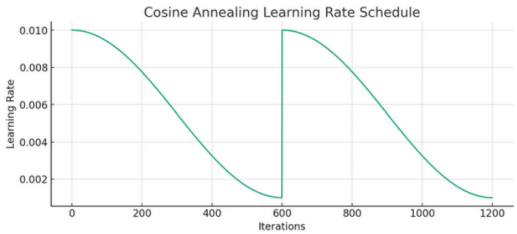
$$\mathcal{L} = \min \sum_{j=1}^{k} \sum_{\boldsymbol{x}_i \in X_j} \|\boldsymbol{x}_i - \boldsymbol{c}_j\|^2$$

  Results are highly dependent on the number k and initialization of centroids c_j
  3 Fehler: Wrong number of initial clusters(Choose correct number of k);
  Potentially wrong cluster(Initialize the algorithm multiple times with different centroid starting points);
  Curse of dimensionality! In high-dimensional representations, all data is dissimilar à Makes it harder to find clusters.
  Adaptations(改进)：Fuzzy-k-means-Clustering：
  Softening: Each instance $\boldsymbol{x}_i$ contains "soft" probabilities for a membership of cluster X_j
  - $P(X_j|\boldsymbol{x}_i)$ "Probability measure of membership"
  - $P(X_j|\boldsymbol{x}_i) \sim 0$: Instance is far away from centroid
  - $P(X_j|\boldsymbol{x}_i) \sim 1$: Instance is close/inside the cluster
  - P ist normalized over all clusters X_j : $\sum_{X_j} P(X_j \mid \boldsymbol{x}_i) = 1$

  Regular k-means
  - Each instance belongs to exactly one cluster.
- DBSCAN：
  Algorithm is parametrized by *minPts* and a distance measure ε (Challenge: correctly define "density" with *minPts* and ε)
  Core Sample: A point is a core sample if at least *minPts* points are within distance ε
  Neighbor: A point is a neighbor if there are not *minPts* points within distance ε, but at least one point within distance ε is a core sample.
  Noise: There is no core sample within distance ε
  Vorteile: DBSCAN is robust against noise; DBSCAN can cluster different data densities; DBSCAN does not require a priori knowledge about number of clusters.
  Nachteile: DBSCAN non-deterministic; Still dependent of distance function; Fitting Selection of ε, *minPts* can be difficult (different densities)
- Autoencoder(是 Unsupervised Learning with Neural Networks，Principle: Network learns to reconstruct its input)

Neural Networks (非计算问题)

1. Optimierungsmethoden zweiter Ordnung sollten den Loss besser minimieren. Dennoch werden sie bei neuronalen Netzen nur sehr selten verwendet. Wieso? Nennen Sie zusätzlich eine Optimierungsmethode erster Ordnung, die eine Optimierungsmethode zweiter Ordnung approximiert. (++)
   The second order optimization method like Newton Method needs the Hessian matrix and invert it. This makes the calculation more difficult. Adam
2. ReLU Funktion und deren Ableitung; LeakyReLU Funktion und deren Ableitung;
3. Geben Sie die Formel und Ableitung für die Aktivierungsfunktion ReLU an. Welchen entscheidenden Nachteil hat ReLU? Welche alternative Aktivierungsfunktion versucht dies zu beheben?
   Dying ReLU; LeakyReLU
4. Was ist der Vorteil von ReLU gegenüber einer Sigmoid- oder Tanh-Aktivierungsfunktion? Die genannten Aktivierungsfunktionen sind nicht-linear, was würde passieren wenn Sie nur lineare Aktivierungsfunktionen für ein neuronales Netz verwenden würden?
   ReLU is very easy to calculate; no saturation behavior for x>0; most used activation function. Without nonlinear activation functions, neural networks of several layers collapse, always to a linear model.
5. Wozu wird Datenaugmentierung durchgeführt und was ist dabei zu beachten? Nennen Sie drei Beispiele wie man Bilddateien augmentieren kann.
   Reduce overfitting. The label must still be correct for the changed data. Brightness and Contrast; Rotation; Translation.
6. Warum werden meist dynamische Lernraten statt statischen Lernraten verwendet? Welchen Vorteil können zyklische Lernraten gegenüber monoton absinkenden Reduktionsverfahren haben? Skizzieren Sie den Verlauf einer zyklischen Lernrate mit Cosinus-Reduktion (cosine-annealing).
   It has the advantages of small and high learning rate. First using high learning rate to obtain good parameters quickly, then using small learning rate to find the minimun.



7. Backward Pass 的计算题(+++)
   Forward Pass 计算题(+) 看 Ubung 3
8. Welches Phänomen wird für Hypothese h in folgender Formel beschrieben, wobei D_L die Lerndaten, D_V die Validierungsdaten sind und $h, h' \in H$?

$$\hat{J}_{D_L}(h) < \hat{J}_{D_L}(h') \wedge \hat{J}_{D_V}(h) > \hat{J}_{D_V}(h')$$

9. Nennen Sie außerdem zwei Verfahren um das Phänomen abzuschwächen.
   Overfitting. Early stopping, Parameter regularization, Dropout, Data augmentation
10. Wie können sich zu groß oder zu klein gewählte Lernraten auf den Trainingsverlauf des Modells auswirken? (1P)
    High learning rates reduces loss fast, but don't find minimum. *Too high learning rates can even diverge training.* Small learning rates take a long time to find minimum. *Too small learning rates will get stuck in every local minimum and can't escape to find global minimum*
11. Ihr neuronales Netz verarbeitet Bilder, deren Pixel sie auf den Wertebereich [0,1]

skalieren. Geben Sie eine Parameterinitialisierung der Netzparameter an, welche auf diesen Bildern zum Dying ReLU Problem führt. (1P) Um dieses und ähnliche Probleme zu umgehen, wollen Sie eine andere, besser geeignete Initialisierungsmethode verwenden. Welche Komponente Ihres neuronalen Netzes sollte dabei Ihre Entscheidung am meisten beeinflussen? (0.5P)

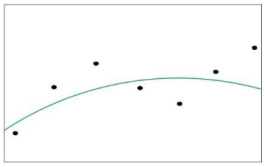
12. Inwiefern ändert sich das Gewichtsupdate wenn, statt einem Eingabevektor, ein Minibatch mit mehreren Eingabvektoren für das Training verwendet wird?

13. Warum wird bei Dropout die Ausgabe der Neuronen im Training mit der Dropoutwahrscheinlichkeit dividiert, bzw. in der Inferenz multipliziert?

14. Nennen Sie eine Eigenschaft, die jede Aktivierungsfunktion besitzen muss. *Nonliear*
15. Sie trainieren ein klassisches neuronales Netz mit Gewichtsregularisierung. Ab einem bestimmten Zeitpunkt ist die Ausgabe des Netzes konsistent mit dem Label und Sie erhalten einen Crossentropyloss mit Wert 0. Wenn sie nun das Netz weiter trainieren lassen: Verändern sich die trainierbaren Gewichte des Netzes weiter? Erläutern Sie Ihre Antwort.

16. Geben Sie die quadratische Fehlerfunktion E des Gradientenabstiegs an sowie die Formel der iterativen Gewichtsoptimierung $\Delta\vec{w}$ in Abhängigkeit von E. Benennen Sie die verwendeten Variablen.(++)

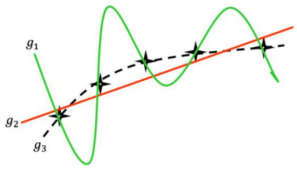
17. Nennen Sie zwei Probleme, die bezüglich der Form der Fehlerflächen beim Gradientenabstieg auftreten können. Geben Sie zwei Methoden an, mit denen diese Probleme jeweils vermieden werden können?(++)
    Stuck in local minimun : dynamic learning rate
    Vanishing gradient : decrease the layers, weight initialization
18. Was lässt sich über die VC-Dimension eines neuronalen Netzes sagen, das aus den untenstehenden Lerndaten (Punkte) die eingezeichnete Kurve approximiert? Wie muss die Topologie des Netzes angepasst werden um die Approximation zu verbessern.



VC-dimension is small

19. „Je höher die VC-Dimension, umso besser kann das Netz aus einem bestehenden Datensatz lernen, d.h. generalisieren.“ Ist diese Aussage wahr oder falsch, begründen Sie Ihre Entscheidung.(++)
    Falsch. Because sometimes high VC-dimension causes overfitting.
20. Was versteht man unter „residual learning“ und wodurch wird damit das Training verbessert?

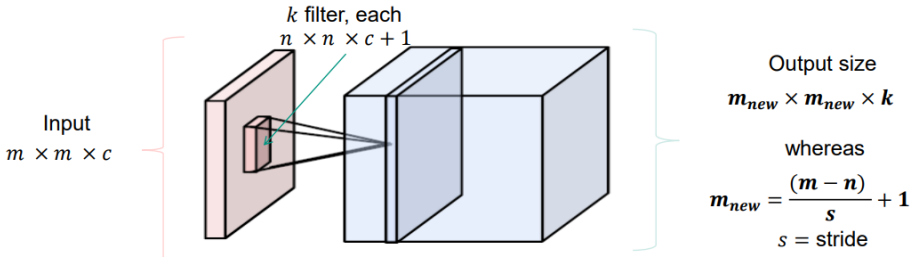
21. Was muss bei der Initialisierung der Gewichte eines Neuronalen Netzes beachtet werden? Nennen und erläutern Sie kurz eine gängige Initialisierungsmethode.
    Xavier or KaimingHe
22. 1. Nennen Sie die Fachbegriffe für das Fehlverhalten (zugrundeliegende Verhalten) von $g1$ und $g2$ und beschreiben Sie den Fehler in der Topologie des neuronalen Netzes, der zu diesem Verhalten führen kann. 2. Skizzieren Sie den typischen Verlauf des Trainings- und Testfehlers in Abhängigkeit der Trainingszyklen für $g1$ und $g2$.
    G1:overfitting, too many layers
    G2:underfitting, without activation fuction



23. Wodurch treten verschwindende Gradienten (vanishing gradients) auf? Welche Gegenmaßnahme gibt es ohne die Anzahl der Neuronen zu verändern?
    Deep neural networks with many layers; too small initial weights
    Activation function; weight initialization such as Xavier or Kaiming He
24. Nennen Sie ein Optimierungsverfahren, das dem einfachen Gradientenabstieg überlegen ist und nennen Sie zwei Gründe wieso es besser ist.
    Adam, it has adaptable learning rate and considers moment.
25. Geben Sie die Funktionsterme von zwei häufig verwendeten nichtlinearen Aktivierungsfunktionen an. Für welches Verfahren im Kontext Neuronaler Netze wird die Ableitung der Aktivierungsfunktion benötigt?
    Backpropagation，Backward-Pass
26. Welcher Bagging-Ansatz für (tiefe) neuronale Netze wurde in der Vorlesung vorgestellt? Wie äußert sich die Verwendung dieser Methode jeweils in Training und Inferenz des neuronalen Netzes?
    Dropout，Randomly deactivate neurons (and their connections) during training with probability p.
27. Was muss bei der Initialisierung der Gewichte von Neuronen eines neuronalen Netzwerks beachtet werden? Was wird dadurch vermieden?
    Avoiding Vanishing/Exploding Gradients; Considering Activation Functions; Proper Scaling; Breaking Symmetry: Weights should be initialized randomly to break symmetry. Weight initialization helps in avoiding problems such as slow convergence, getting stuck in local minima, or even failing to learn at all.
28. Wie unterscheidet sich Stochastic Gradient Descent (bzw. Pattern Learning) vom „echten“ Gradientenabstieg? Was sind die jeweiligen Vorteile der beiden Verfahren?

CNN

1. 关于计算

- RGB-images are three-dimensional
- 

- 如果有 padding：m_{new}=（m-n+2p）/s +1
- k convolutional filter create k feature maps $w \times \square \times k$
- Max Pooling $\max\{a \in A\}$
- $softmax(z_i) = \frac{e^{z_i}}{\sum_k e^{z_k}}$
- Crossentropy Loss：

  $\mathcal{L}_{\text{CE}} = -\sum_{i=1}^{n} \text{label}_i \cdot \log(\text{prediction}_i)$

  Softmax is always applied beforehand to prediction

  $\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_k e^{z_k}}$

  Let's say we use a CNN for classification that contains a FC-layer with 3 neurons as final classification layer. The outputs of the neurons are: $\text{prediction} = [4 \quad 2 \quad 1]$

  The corresponding label in dataset:

  $\text{label} = [1 \quad 0 \quad 0]$

  Then Softmax calculates

  $[\frac{e^4}{e^4+e^2+e^1} \quad \frac{e^2}{e^4+e^2+e^1} \quad \frac{e^1}{e^4+e^2+e^1}]$

  $[0.844 \quad 0.114 \quad 0.042]$

  And the crossentropy loss is (We are using the natural logarithm ln)

  $\mathcal{L}_{\text{CE}} = -(1 \cdot \log(0.844) + 0 \cdot \log(0.114) + 0 \cdot \log(0.042)) = -\log(0.844) = 0.169$
- number parameters=numberFilters×(kernelWidth×kernelHeight×inputChannels+1)
- Calculate parameters of fully connected layer:

  $\text{number parameters} = (\text{input features} + 1) \times \text{number neurons}$

  Example:

  Let's say we want to classify images into 5 possible classes. The images have size 32×32 and are in RGB. For this classification we want to use 2 Convolutional Layer and one fully connected layer.

  - The first convolutional layer has 16 filters, stride 2 and a 5×5 kernel
  - The second convolutional layer has 32 filters, stride 1, a 3×3 kernel and padding to keep the same feature map resolution

  Calculate the number of trainable parameters!

  For the first convolutional layer we get:

  $16 \cdot (5 \cdot 5 \cdot 3 + 1) = 1216 \text{ Parameters}$

  For the second convolutional layer we get:

  $32 \cdot (3 \cdot 3 \cdot 16 + 1) = 4640 \text{ Parameters}$

Dimension of feature map after first convolutional layer:

$$14 \times 14 \times 16$$

Second convolution uses padding to keep the resolution identical but increases number of filters. So the dimension after the second convolutional layer is:

$$14 \times 14 \times 32$$

Now we can calculate the number of trainable parameters of the fully connected layer:

$$\text{number parameters} = (\text{input features} + 1) \times \text{number neurons}$$

which is:

$$(14 \cdot 14 \cdot 32 + 1) \cdot 5 = 31365$$

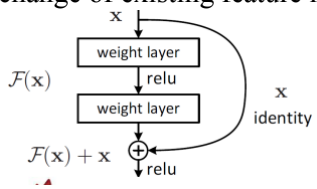
So we have

$1216 + 4640 + 31365$ = 37221 parameter.

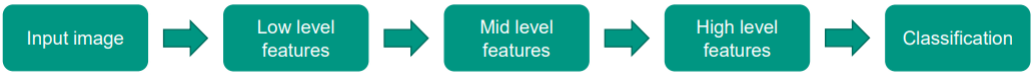
2. Bei CNNs ist die Größe der Kernel ein wichtiger Hyperparameter. 1. Welche Vorteile haben 3 × 3 Kernel gegenüber größeren z.B. 7 × 7 Kernel hinsichtlich des rezeptiven Felds? 2. Wie verhält sich das rezeptive Feld bei CNNs die nur 1 × 1 Kernels verwenden? 3. Die Ausgabe eines 3 × 3 Kernels hat eine kleinere Auflösung als seine Eingabe. Erklären Sie eine Art von Padding und eine dazu passende Größe, um dies zu verhindern.
   1. *This leads to more efficient trainability and often better results because the smaller kernels can introduce more nonlinearities into the learning processes.*
   2. *CNNs that exclusively use 1 × 1 kernels cannot capture spatial features*
   3. *Using padding like zero padding can keep resolution constant for 3x3 kernel with stride=1*
3. Wieso werden GPUs statt CPUs für das Training und Ausführen von CNNs verwendet?
   GPUs are optimized for computer graphics which are mostly matrix operations. All mathematical functions of CNNs/NNs can be represented as matrix operations
4. Heutzutage verwendet fast jedes neuronale Netz Skip-Connections. 1. In welcher Netzarchitektur wurden sie erstmalig vorgestellt? 2. Welches Problem wird durch die Verwendung von Skip-Connections umgangen? 3. Was ist die aktuelle Theorie weshalb Skip-Connections dieses Problem umgehen?
   1. *ResNet(2015)*
   2. *Skip connections fix performance problem. Worse performance with more layers.*
   3. *Probably improvement of the loss landscape*
5. Nennen Sie einen Grund SiLU statt ReLU für ein CNN zu verwenden
   More recently Swish/SiLU
   - $f(x) = x \cdot \sigma(x)$

   SiLu gets better result als ReLU for CNN.
6. Wofür werden entweder Pooling Layer oder Strided Conv. Layer verwendet? Nennen Sie für beide Verfahren jeweils ein Vorteil.
   For small image areas in feature map.
   Pooling does not need parameter.
   Stride can decrease feature map resolution.
7. Bei Computer Vision Aufgaben werden heutzutage keine herkömmlichen neuronalen Netze verwendet sondern CNNs. Nennen Sie dafür zwei Gründe.
   Neural networks do not work on variable size input.
   Neural networks are not translation invariant
8. Welche wesentliche Technik wurde in der ResNet-Architektur eingeführt? Erklären Sie diese kurz.
   Skip connection.

Identity parallel to each residual block; Add identity to output of block; Learn the change of existing feature maps



9. Für welche Arten von Daten werden CNNs typischerweise verwendet? *Images*
10. Warum besitzen CNNs weniger Parameter/Gewichte als vollvernetzte Neuronale Netze? *CNNs use convolution.*
11. Wie sehen die Filter von auf realen Bilddaten trainierten CNNs typischerweise aus? Gibt es Unterschiede je nach Position (Tiefe) des Filters im CNN? Beschreiben Sie diese.
    The filters in a CNN trained on real-world image data typically exhibit a hierarchical structure in terms of complexity and abstraction, depending on their depth within the network:



12. Was ist ein Vorteil von Fully Convolutional Networks gegenüber CNNs mit Fully-Connected Schichten am Ende und wofür werden diese typischerweise verwendet?\
    Fully Convolutional Networks can handle inputs of arbitrary size(different resolutions), while Fully-Connected layers need fixed resolution input.
13. Erklären Sie "Weight Sharing". Welche Vorteile ergeben sich hierdurch gegenüber klassischen Neuronalen Netzen?
    One weight of convolutional kernel is used multiple times for different inputs → Addition of gradients for different inputs
14. Nennen Sie zwei verschiedene Methoden zur Initialisierung von Gewichten. Was passiert, wenn alle Gewichte mit 1 initialisiert werden?
    Start training from scratch like Kaiming He, (Xavier)
    Take weights from already trained CNN (Transfer Learning)
    May cause vanishing gradient
15. Wie wird das Vanishing-Gradient Problem bei ResNet gelöst? Erläutern Sie dies kurz.
    *Use skip connection. Add the identity to output of block(*allowing gradients to flow through alternative paths.*)*
16. Welche Nachteile hat die Verwendung der Sigmoid Aktivierungsfunktion für tiefe neuronale Netze wie Resnet ohne „skip connections"?
    GPT4：Vanishing Gradient Problem; Not Zero-Centered Output；Computational Expense；Not Suitable for Deep Networks
17. Was ist der Zweck einer 1x1 Faltung, wie sie im Inception Modul vorkommt?
    GPT4: Dimensionality Reduction
18. Was versteht man unter Padding bei CNN? Nennen Sie drei in der Vorlesung genannten Möglichkeiten um die Eingabe eines CNN zu erweitern (Padding).
    Increase resolution of input feature map.
    Zero Padding; Reflect Padding; Circular Padding

Bayesian Learning

1. 计算题: Navie Bayes Classifier $v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_i P(a_i|v_j)$

   Bayesian Network $P(y_1, \ldots, y_n) = \prod_{i=1}^{n} P(y_i|\text{Predecessor}(Y_i))$

2. Was kann bei einem Bayes'schen Netz gelernt werden? Mit welcher Methode erfolgt dies, wenn die Struktur bekannt ist und Variablen nur teilweise beobachtbar sind?
   GPT4: Structure learning, Parameter learning.
   Gradient ascent; Expectation-Maximization Algorithm

3. Gegeben zwei Hypothesen i und j. Unter welcher Annahme lässt sich die Maximum a-Posteriori Hypothese zur Maximum Likelihood Hypothese vereinfachen?
   Goal: Find the hypothesis h of H with the highest probability given the observed data. This is the maximum a posteriori (MAP) hypothesis.

$$\begin{aligned} h_{MAP} &= \arg\max_{h \in H} P(h|D) \\ &= \arg\max_{h \in H} \frac{P(D|h)P(h)}{P(D)} && \text{Bayes} \\ &= \arg\max_{h \in H} P(D|h)P(h) && P(D) = const. \end{aligned}$$

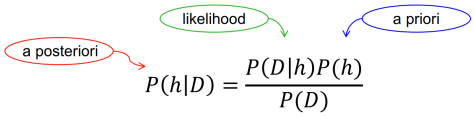
   Under the assumption $P(h_i) = P(h_j)$, we can simplify this to the maximum likelihood (ML) hypothesis:

$$h_{ML} = \arg\max_{h \in H} P(D|h)$$

4. Unter welcher Annahme lässt sich der optimale Bayes-Klassifikator zum Naiven BayesKlassifikator vereinfachen?
   Simplifying assumption (a_i conditionally independent):

$$P(a_1, a_2 \ldots a_n|v_j) = \prod_i P(a_i|v_j)$$

   Bayes' Theorem (for ML)



   - $\boldsymbol{P(h)}$: A priori probability, that h of H is valid, before observing data D.
   - $\boldsymbol{P(D)}$: Probability of observing data D, without knowledge about valid hypotheses.
   - $\boldsymbol{P(D|h)}$: Likelihood, probability of observing data D under the model/hypothesis h.
   - $\boldsymbol{P(h|D)}$: A posteriori probability, that h is valid given the observed data D and the prior knowledge about h

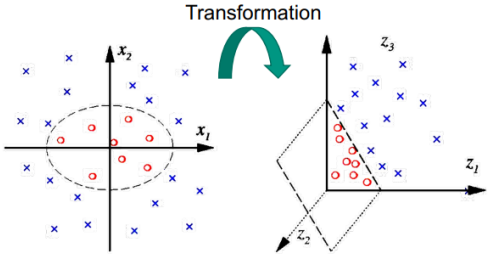
5.

SVM

1. Linear Support Vector Machine
   Problem: Classification – There are many possibilities to separate the two sets (red and blue), but what is the optimal solution for the problem?
   Intuition: Size of the margin determines the generalization capability
   Solution: Find the best separating line/hyperplane with maximum margin to the classes
2. Support Vectors(会画)：Data points of the training dataset that have a direct impact on the position of the hyperplane and would change it if removed. Data points closest to the hyperplane. Data points that are the "hardest" to classify
   - Hyperplane with maximum margin
     - Maximize $\frac{2}{|\vec{w}|}$ → Minimize $|\vec{w}|^2$
     - Under the **conditions**: $y_i(\vec{w}\,\vec{x_i} + b) \geq 1,\ i = 1 \ldots n$
3.
4. Soft Margin SVM：Allow a small number of wrong classifications
   Leads to a larger margin and higher generalization
5. Nonlinear Kernel Methods (SVM)
6. Idea：Transform the data into another (higher-dimensional) space where the data can be separated linearly and solve the problem there.
   In case of classification: Linear separation in the transformed space.

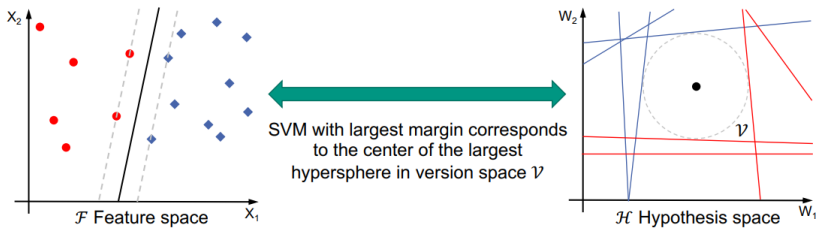
   Example: $\phi: \mathbb{R}^n \to \mathbb{R}^m$ often $m \gg n$
   - Monomial-transformation

   $\phi: \mathbb{R}^2 \to \mathbb{R}^3$
   $(x_1, x_2) \to (z_1, z_2, z_3) = (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$



7. Kernel-Trick&Kernel function
   Kernel function $K(\vec{x}, \vec{y})$: Defines a dot product / similarity measure in the transformed space without explicitly calculating it.

   Kernel-trick: Instead of calculating the complex nonlinear transformation $\phi(\,)$ and the scalar product, we use a kernel that does this implicitly, thus saving a lot of computational operations.
8. Version Space for SVM



9. SVM Extensions - Classification on k Classes
   - One – versus – All
     - *k* SVMs (one for each class)
     - Voting
   - One – versus – One
     - *k(k-1)/2* SVMs
     - Voting
   - Multi-class-affiliation
     - *k* SVMs (one for each class)
     - Voting
   - *k* – class SVM (according to Watkins)
     - One common optimization method
     - No voting

Decision Trees (Decision trees are a non-parametric supervised learning method)

1. Properties:
   Non-parametric: No assumptions about underlying distribution
   Interpretable:
2. Entropy: : Entropy is a measure of the homogeneity (in terms of class membership) of the current data S
   Entropy for K **Classes**: $H(S) = -\sum_{i=1}^{K} p(y_i)\log_2 p(y_i)$
   Entropy for **2 Classes**($\oplus$, $\ominus$): $H(S) = -p_{\oplus}\log_2 p_{\oplus} - p_{\ominus}\log_2 p_{\ominus}$
   - **High entropy**
     - Y is derived from an almost uniformly distributed probability density
     - Data sampled from this density are difficult to predict
   - **Low entropy**
     - Y is derived from a probability density with a high probability for one class, relative to the others
     - Data sampled from this density are highly predictable
3. Information Gain: $IG(S, A) = H(S) - \sum_{v \in V(A)} \frac{|S_v|}{|S|} H(S_v)$
   Objective of learning with decision trees: Select attributes that minimize entropy and maximize information gain. Classify learning examples with as few steps as possible → Tree with little depth
4. ID3 Algorithm:
   Top-Down: Build decision tree recursively from root node
   Greedy: Each time an attribute is selected, the attribute that maximizes the information gain is used to split the data.
   ID3 is prone to overfitting (Stops only when all instances have been classified perfectly, even if the data is noisy/wrong). Can be improved by bagging or random forest.
5. How to combat the high variance/overfitting properties of ID3?
   - **Maximum depth**: Growing the tree stops after a certain number of steps, even if not all data is perfectly classified.
   - **Minimum samples:** A node cannot be split if it contains less than a specified number of training examples
   - **Early stopping:** Stop growing the tree if the validation error increases
   - **Pruning**: Remove non-critical parts of the tree to reduce complexity

   Reduce overfitting with multiple trees:
   - **Bagging**
   - **Random Forests**
6. Pruning: Remove a node (subtree) and replace it with a leaf node of the most common class.
   Bottom up: Starting from the leaves, replace each node with a leaf node until the validation error starts to increase.
7. Bagging:
   Bootstrap: Create k datasets $D1$ to Dk from D via layback sampling
   Advantage: Reduce variance without increasing bias
   Disadvantage: Computational cost increases; Loss of interpretability
8. Random Forests: Advantage: Generally, much better results than single decision tree
   Problem of bagging: Models are highly correlated.
   Now: Choose random subset $s < d$ of attributes for each split. Create nodes using the best attribute for maximizing information gain in s.
   Why: To ensure, that the k learned trees are less correlated. Each tree uses a different subset of attributes.