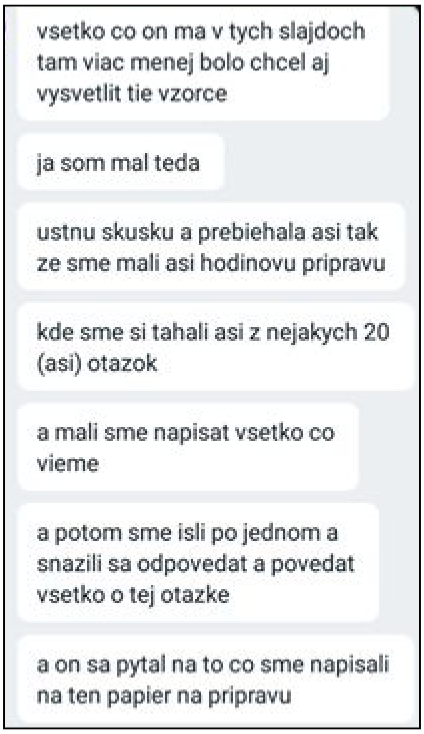
Všeobecné info

PODMIENKY ABSOLVOVANIA A SPÔSOB HODNOTENIA

- Projekty - **3x7 = 21 bodov**)
- Cvičenia - **10 bodov**.
- Skúška - **aspoň 7 bodov**.
- **Celkové hodnotenie:** A (50-46), B (45-41), C (40-36), D (35-31), E (30-**26**), Fx (25-0).

Info od týpka z matfyzu

** My máme písomnú skúšku*



1. spoj. perceptron so sig
a) schéma
b) vstupno-výstupný vzťah
c) odvodenie učiac pravidla (delta) po kroderet chyby
d) bias

2. BP
a) podstata alg
b) od čoho priamo závisí zmena váhy w_{ij} (vzťah)
c) vylepšenia alg a ich účel

3. Autoasociat (GI)
a) podstata (slovne, matem., graf)
b) akú úlohu má Gram-Schidt ortog

4.) PCA
a) schéma
b) Oja a vychetkit
c) čo vieme povedať o váhach po konvergencii v GHA (kmc neurónov)
d) ako je zabezpečené fungovanie pri viacerých neur. vo vrstve

5.) SOM
a) alg trénovania
b) dôvod zmeny okolia
c) čo SOM aproximuje

6) RBF
a) schéma a funkcie
b) RBF vs MLP učenie a zobrazenie neur
c) k-means ako a prečo
d) RLS v RBF?

7. SRN
a) schéma, aktiv. rovnice
b) aké interné reprezentácie
c) aký alg na trénovanie
d) architekt. bias

8. Stoch. NS
a) ako $P(s)$
b) koncepty
c) výhoda stoch vs. determ
d) princíp (optimalizácia) v odvodzaní učiaceho pravidla (BM, LBN,

Otázky na záverečnú skúšku - Neurónové siete

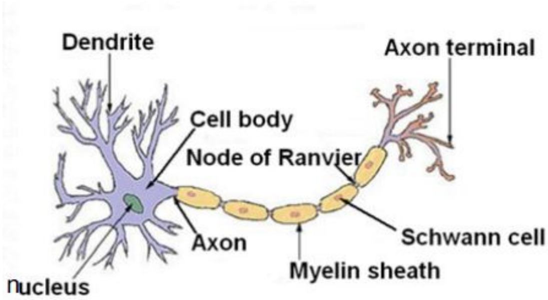
- Otázky som našiel na http://dai.fmph.uniba.sk/courses/NN/ns-otazky.pdf
- Vôbec to nemusia byť takéto otázky ale podobajú sa na to čo sme preberali

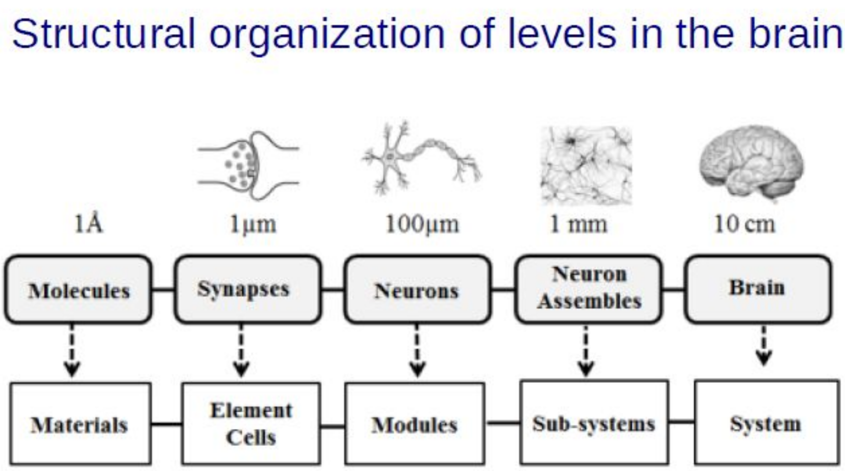
1. Stručná história konekcionizmu, vlastnosti biologického neurónu, model neurónu s prahovou logikou, implementácia Booleových funkcií. Paradigmy učenia a typy úloh pre NS.

Connectionism – theory of information processing, inspired by biology (the brain). It is based on Artificial Neural Networks (ANNs).

It has two goals:
- **theoretical foundations of cognitive science** (modeling of cognitive processes)
  - contrasting with symbolic approaches
  - features: parallelism, robustness, learning from experience,...
- **applications in practical problems**
  - tasks: pattern recognition, classification, associative memory, time series prediction, dimensionality reduction, data visualization, ...

Štruktúra biologického neurónu
V mozgu - ~ 10^{11} neurónov a ~ 10^{15} synáps



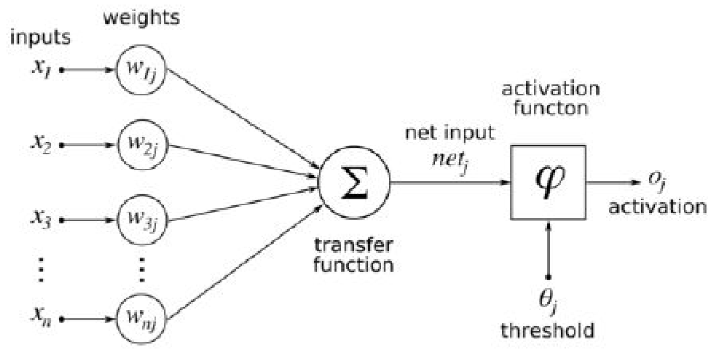


Typical artificial neuron model

1. receives signals from other neurons (or sensors)
2. processes (integrates) incoming signals
3. sends the processed signal to other neurons (or muscles)

Deterministic model

$o = f\,(\sum_i w_i x_i - \theta)$



Stochastic model $\quad P(o=1) = 1/(1+\exp(-net/T))$

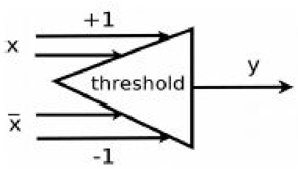
History of classical connectionism

- Aristoteles (400 BC) – introduced concepts of memory, and connectionism
- Spencer (1855) – separated psychology from philosophy, postulated that "neural states affect psychological states", knowledge is in connections.
- James (1890) – model of associative memory, "law of neural habit"
- Thorndike (1932) – distinguished sub-symbolic view on neural associations, formulated two laws of adaptation: "law of effect" and "law of exercise" (currently known as reinforcement in operant conditioning).
- McCulloch & Pitts (1943) – neural networks with threshold units
- Minsky (1967) extended their results to comprehensible form, and put them in the context of (formal) automata theory and theory of computation.

First neural network

McCulloch & Pitts (1943) – neural networks with threshold logic units

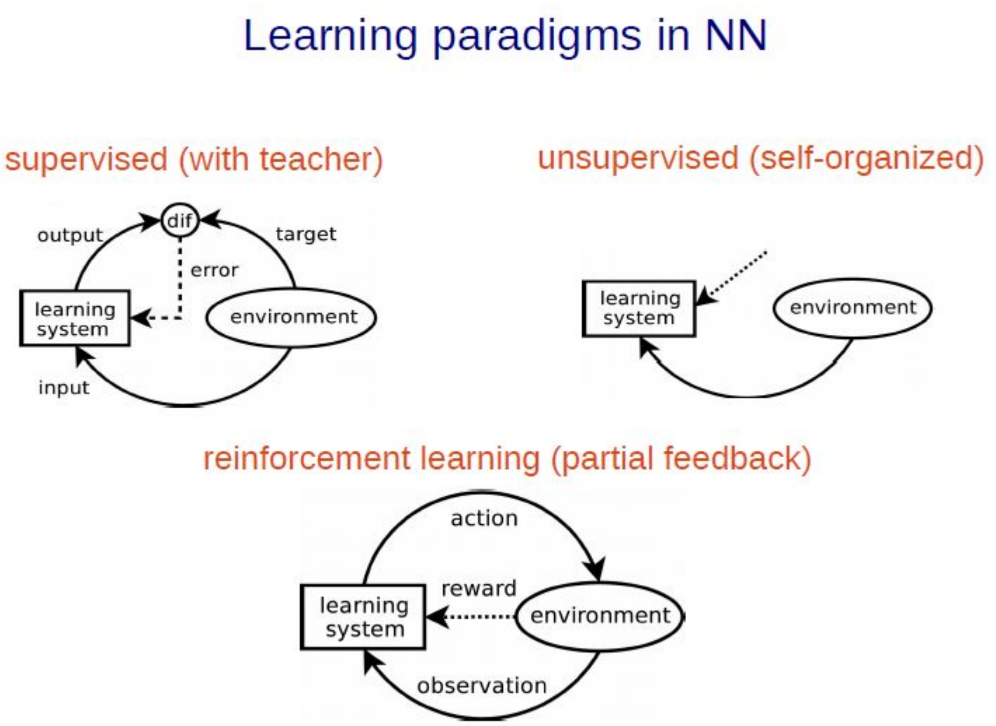
- Threshold Logic unit:



Y = 1, if sum(x) – sum(non x) ≥ threshold
0, otherwise.

- can simulate any linearly separable Boolean function:
- Fixed weights, positive and negated inputs

- Theorem: Any Boolean function f: {0,1}n {0,1} can be simulated by a two-layer NN with logical units.



Learning tasks

- Pattern association (auto-, hetero-)
- Pattern classification (within pattern recognition)
- Feature extraction (within PR or independently)
- Data compression
- Function approximation
- Control
- Filtering
- Prediction
- Signal generation (with recurrent networks)
- ...

2. Binárny perceptrón: pojem učenia s učiteľom, učiace pravidlo, algoritmus trénovania, deliaca nadrovina, klasifikácia vzorov, lineárna separovateľnosť, náčrt dôkazu konvergencie, definícia a príklad.

Summary of perceptron algorithm

Given: training data: input-target $\{\boldsymbol{x}, d\}$ pairs, unipolar perceptron
Initialization: randomize weights, set learning rate, $E = 0$.

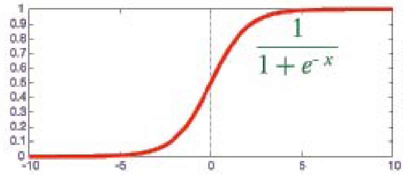
Training:

1. choose input $\boldsymbol{x}$, compute output y
2. evaluate error function $e(t) = ½\,(d - y)^2$, $E = E + e(t)$
3. adjust weights using delta rule (if $e(t) > 0$)
4. if all patterns used, then goto 5, else go to 1
5. if $E == 0$ (all patterns in the set classified correctly), then end
   else shuffle inputs, $E = 0$, go to 1

3. Spojitý perceptrón: Rôzne aktivačné funkcie perceptrónu, chybové funkcie a spôsob minimalizácie, učiace pravidlá, algoritmus trénovania perceptrónu. Súvis s Bayesovským klasifikátorom.

Continuous perceptron

- Nonlinear unit with activation function: $y = f(net) = 1 / (1+e^{-net})$
  Has nice properties:
  - boundedness
  - monotonicity
  - differentiability



sigmoid

- Quadratic error function: $E(\mathbf{w}) = ½ \Sigma_p(d^{(p)} - y^{(p)})^2$ [p ~ patterns]
- (unconstrained) minimization of the error function: necessary conditions $e(\mathbf{w}^*) \leq e(\mathbf{w})$ and $\nabla e(\mathbf{w}^*) = 0$, gradient operator
  $\nabla = [\partial/\partial w_1, \partial/\partial w_2, ...]^T$. Minimizing $E(\mathbf{w})$ leads to
- (stochastic, online) gradient descent learning:
  $$w_j(t+1) = w_j(t) + \alpha \ (d^{(p)} - y^{(p)}) f'(net) x_j = w_j(t) + \alpha\delta^{(p)}x_j$$
- (alternative) batch learning: (update after each epoch)
  $$w_j(t+1) = w_j(t) + \alpha\Sigma_p\delta^{(p)}x_j^{(p)}$$

4. Viacvrstvové dopredné neurónové siete: architektúra a aktivačné vzorce, odvodenie metódy učenia pomocou spätného šírenia chýb (BP) pre dvojvrstvovú doprednú NS, modifikácie BP, typy úloh pre použitie doprednej NS.

Multi-layer perceptrons

- Generalization of simple perceptrons
- Features:
  - contains hidden-layer(s)
  - neurons have non-linear differentiable activation function
  - full connectivity b/w layers
- (supervised) error "back-propagation" learning algorithm introduced
- originated after 1985: Rumelhart & McClelland: *Parallel distributed processing* (described earlier by Werbos, 1974, 1982)
- theoretical analysis difficult
- response to earlier critique of perceptrons (Minsky & Papert, 1969)

Two-layer perceptron

- Inputs $\boldsymbol{x}$, weights $\boldsymbol{w}$, $\boldsymbol{v}$, outputs $\boldsymbol{y}$
- Nonlinear activation function f
- Unit activation:

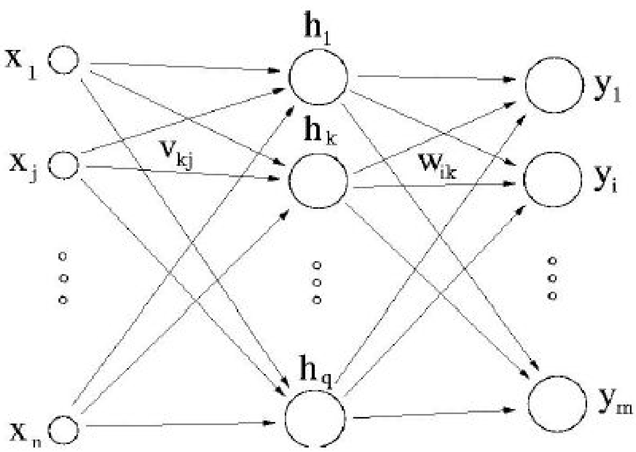
$$h_k = f\left(\sum_{j=1}^{n+1} v_{kj} x_j\right)$$

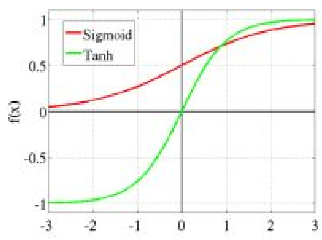
$$y_i = f\left(\sum_{k=1}^{q+1} w_{ik} h_k\right)$$

- Bias input: $x_{n+1} = h_{q+1} = -1$
- Activation function examples:

$$f(net) = \sigma(net) = \frac{1}{1+e^{-net}}$$

$$f(net) = \tanh(net) = \frac{e^{net} - e^{-net}}{e^{net} + e^{-net}} = \frac{2}{1+e^{-2net}} - 1$$

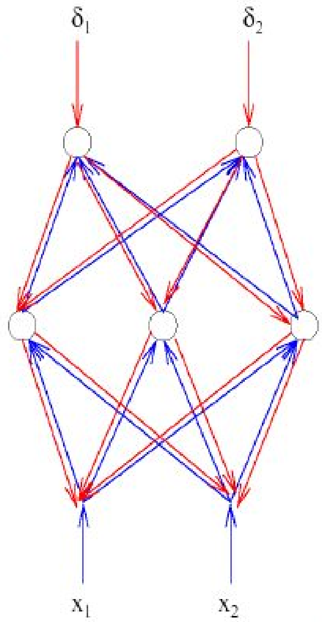




Summary of back-propagation algorithm

Given: training data: input-target $\{\boldsymbol{x}^{(p)}, \boldsymbol{d}^{(p)}\}$ pairs
Initialization: randomize weights, set learning parameters
Training:

1. choose input $\boldsymbol{x}^{(p)}$, compute outputs $\boldsymbol{y}^{(p)}$ (forward pass),
2. evaluate chosen error function $e(t)$, $E \leftarrow E + e(t)$
3. compute δ_i, δ_k (backward pass)
4. adjust weights Δw_{ik} and Δv_{kj}
5. if all patterns used, then goto 6, else go to 1
6. if stopping_criterion is met, then end
   else permute inputs and go to 1



No well-defined stopping criteria exist for BP, neither can it be shown in general to converge. Suggestions:

- when change in E_{epoch} is sufficiently small (<1%)
- when generalization performance is adequate

Sequential and batch modes of training

Sequential mode

- on line (example-by-example), stochastic
- able to track small changes in training data
- easier to implement, requires less local storage
- difficult to establish theoretical conditions for convergence

Batch mode

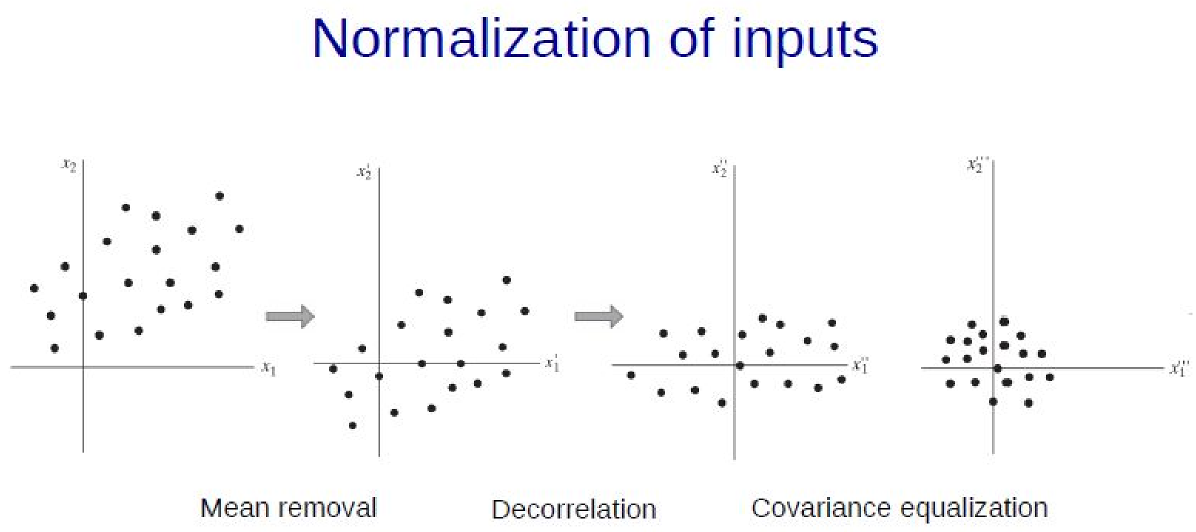
- adaptation performed at the end of each epoch, deterministic
- provides an accurate estimate of gradient vector, statistical inference
- parallelization possible

$$E_{av} = 1/(2N)\ \Sigma^{N}_{p=0} (\mathbf{d}^{(p)} - \mathbf{y}^{(p)})^2$$

$$\Delta w_{ik} \propto -\partial E_{av}(t)/\partial w_{ik}$$
$$\Delta v_{kj} \propto -\partial E_{av}(t)/\partial v_{kj}$$

- **typy úloh pre použitie doprednej NS**
  - Klasifikacia
  - Regresia

5. Viacvrstvová dopredná NS ako univerzálny aproximátor funkcií (teorém), trénovacia a testovacia množina, generalizácia, preučenie, skoré zastavenie učenia, selekcia modelu, validácia modelu. Hlboké učenie NS.



MLP as a universal approximator

Theorem: Let's have $A_{train} = \{x^{(1)},..., x^{(p)},..., x^{(N)}\}$, $x^{(p)} \in \mathbb{R}^n$. For $\epsilon > 0$ and arbitrary continuous function F: $\mathbb{R}^n \rightarrow (0,1)$ defined on discrete set A_{train} there exists such a function G:

$$G(x^{(p)}) = f\left(\sum_{k=1}^{q+1} w_k f\left(\sum_{j=1}^{n+1} v_{kj} x_j^{(p)}\right)\right)$$

where parameters w_k, $v_{kj} \in \mathbb{R}$ and $f(z) = \mathbb{R} \rightarrow (0,1)$ is a continuous and monotone-increasing function satisfying $f(-\infty) = 0$ and $f(\infty) = 1$, such that:

$$\Sigma_p \mid F(x^{(p)}) - G(x^{(p)}) \mid < \epsilon.$$

We say that G approximates F on A_{train} with accuracy ϵ.

G can be interpreted as a 2-layer feedforward NN with 1 output neuron.

- it is an existence theorem
- curse of dimensionality – sparsity problem, how to get a dense sample for large n and complex F

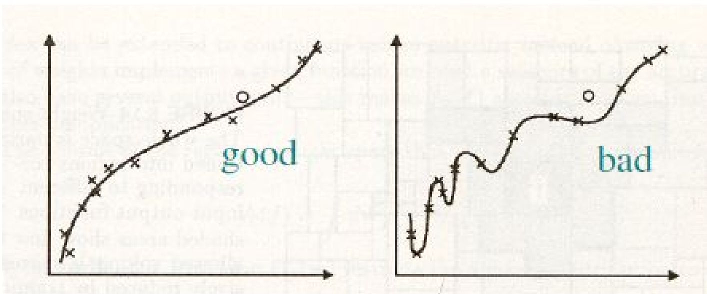
Hecht-Nielsen (1987), Hornik, Stinchcombe & White (1989)

Generalization

Data set:

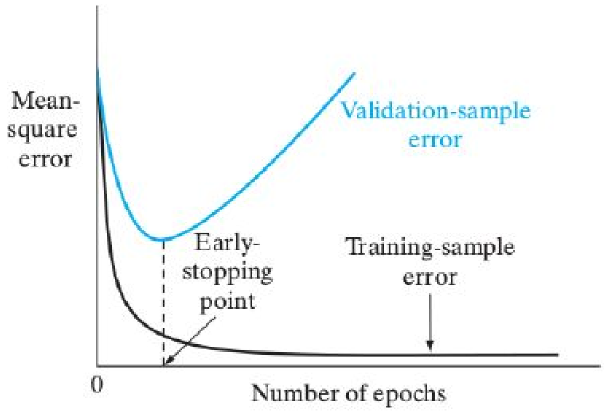
$$A = A_{estim} \cup A_{val} \cup A_{test}$$

- Validation set is used for model selection.
- Generalization (= testing set performance) is important in using ANNs.



Generalization is influenced by:

- size of A_{estim} and its representativeness
- architecture of NN
- complexity of the problem



Cross validacia - typicky na 10 - 20 % trénovacej množiny

Early stopping

- “BP algorithm is considered to have converged when
  - … the Euclidean norm of the gradient vector reaches a sufficiently small gradient threshold.” (Kramer and Sangiovanni-Vincentelli, 1989)
  - … the absolute rate of change in the average squared error per epoch is sufficiently small.”

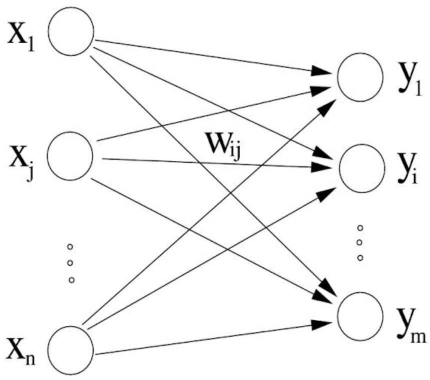
6. Lineárne modely NS: vzťah pre riešenie systému lin. rovníc v jednovrstvovej sieti, pojem pseudoinverzie matice (Moore-Penrose), autoasociatívna pamäť: lineárny obal, princíp funkcie modelu, detektor novosti.

Linear NN models

Input vector: $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$

Output vector: $\boldsymbol{y} = [y_1, y_2, \ldots, y_m]^T$

Weight matrix: $\mathbf{W}$ ~ type $[m \times n]$



Linear transformation $\varphi : \Re^n \to \Re^m$, $\boldsymbol{y} = \mathbf{W}\boldsymbol{x}$

☹ ignores saturation property of neurons

☺ allows to find analytic solutions using linear algebra.

(Kohonen, 1970; Anderson, 1972; Cooper, 1973)

- Adding layers in a linear NN does not appear reasonable (since no complexity is added).
- But: It allows nonlinear learning dynamics in linear deep networks (Saxe, 2015).

Auto-associator case

Let's consider $N < n$ and the autoassociative case: $\boldsymbol{y}^{(p)} = \boldsymbol{x}^{(p)}$, $m = n$

Model is supposed to remember N prototypes $[\boldsymbol{x}^{(1)}\ \boldsymbol{x}^{(2)} \ldots \boldsymbol{x}^{(N)}]$.

Goal: train on prototypes and then submit a corrupted version of a prototype. Model should be able to reconstruct it.

Since $\mathbf{Y} = \mathbf{X}$, $\mathbf{W} = \mathbf{X}\mathbf{X}^{+}$. How to interpret $\mathbf{W}$?

In special case, which is too restrictive ($N = n$, linearly independent inputs), we would have a trivial solution $\mathbf{W} = \mathbf{I}$ (identity).

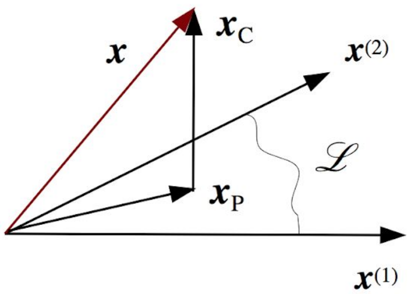
How about a general case?

Training set $\mathrm{A}_{\mathrm{train}} = \{\boldsymbol{x}^{(p)}, p = 1,2,\ldots,N\}$ forms the linear manifold $\mathscr{L}$.
NN considers every departure $\boldsymbol{x}$ from $\mathscr{L}$ as added noise that needs to be filtered out by projecting $\boldsymbol{x}$ to $\mathscr{L}$:

We need to show that output $\mathbf{W}\boldsymbol{x} = \mathbf{X}\mathbf{X}^{+}\boldsymbol{x} = \boldsymbol{x}_{\mathrm{P}}$ (filtered version of $\boldsymbol{x}$), i.e. that operator $\mathbf{W} = \mathbf{X}\mathbf{X}^{+}$ makes an orthogonal projection to $\mathscr{L}$.

Alternatively, the NN model with operator $\mathbf{W} = \mathbf{I} - \mathbf{X}\mathbf{X}^{+}$ is called novelty detector, where $\mathbf{W}\boldsymbol{x} = \boldsymbol{x}_{\mathrm{C}} \in \mathscr{L}^{\perp}$.

Now assume: you learned N patterns, and want to add $(N+1)$-st pattern.
How to change $\mathbf{W}$ efficiently?



7. Lineárne modely NS: účel Grammovho-Schmidtovho ortogonalizačného procesu, GI model. Pamäť korelačnej matice ako autoasociatívna pamäť, vzťah pre výpočet váh, presluch, porovnanie s GI.

Summary

- Linear models were studied during connectionist depression in the 1970s
- Single layer models as auto-associative memories
- Analytic solutions possible
- General inverse model – noise filtering by projection to linear manifold (of the training data)
- GI – as novelty detector
- Correlation Matrix Memory – Hebbian-based learning, subject to cross-talk
- GI better in general, for sufficiently dissimilar inputs both models are comparable

8. Samoorganizácia v NS, základné princípy, pojem učenia bez učiteľa, typy úloh použitia, Ojovo pravidlo učenia pre jeden lineárny neurón, vysvetlenie konvergencie.

9. Metóda hlavných komponentov pomocou algoritmu GHA a APEX, architektúra modelu, vzťah pre adaptáciu váh, pojem vlastných vektorov a vlastných čísel, redukcia dimenzie, aplikácia na kompresiu obrazu.

10.SOM model: algoritmus, parametre, základné koncepty, vlastnosti, príklad použitia.

11. RBF model: aktivačné vzorce, bázové funkcie, príznakový priestor, problém interpolácie, trénovanie modelu, aproximačné vlastnosti RBF siete, princíp algoritmu RLS.

12. NS na spracovanie sekvenčných dát: reprezentácia času, typy úloh pre rekurentné NS. Modely s časovým oknom do minulosti, výhody a nedostatky, príklad použitia.

13. Rekurentné NS: princíp trénovania pomocou algoritmu BPTT a RTRL. Teoretické vlastnosti RNS.

14. Elmanova sieť: interné reprezentácie pri symbolovej dynamike, Markovovské správanie, architekturálna predispozícia.

15.Sieť s echo stavmi (ESN): architektúra, inicializácia, trénovanie modelu, vplyv parametrov na vlastnosti rezervoára, echo vlastnosť, pamäťová kapacita.

16. Hopfieldov model NS: deterministická dynamika, energia systému, relaxácia, typy atraktorov, autoasociatívna pamäť – nastavenie váh, princíp výpočtu kapacity pamäte.

17. Nelineárne dynamické systémy: stavový portrét, dynamika, typy atraktorov. Stochastický Hopfieldov model NS: parameter inverznej teploty, princíp odstránenia falošných atraktorov.

18. Hlboké učenie: základné koncepty, spôsoby pred/trénovania trénovania hlbokých sietí (DN), diskriminatívny a generatívny prístup, koncept konvolúcie, autoenkóder, GAN model, príklady úspešného použitia DN.