Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

FIIT-0000-00000

Martin Schnürer

Spracovanie prirodzeného jazyka neuronovými sieťami

Bakalárska práca

Vedúci práce: Ing. Márius Šajgalík

Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

FIIT-0000-00000

Martin Schnürer

Spracovanie prirodzeného jazyka neuronovými sieťami

Bakalárska práca

Študijný program: Informatika

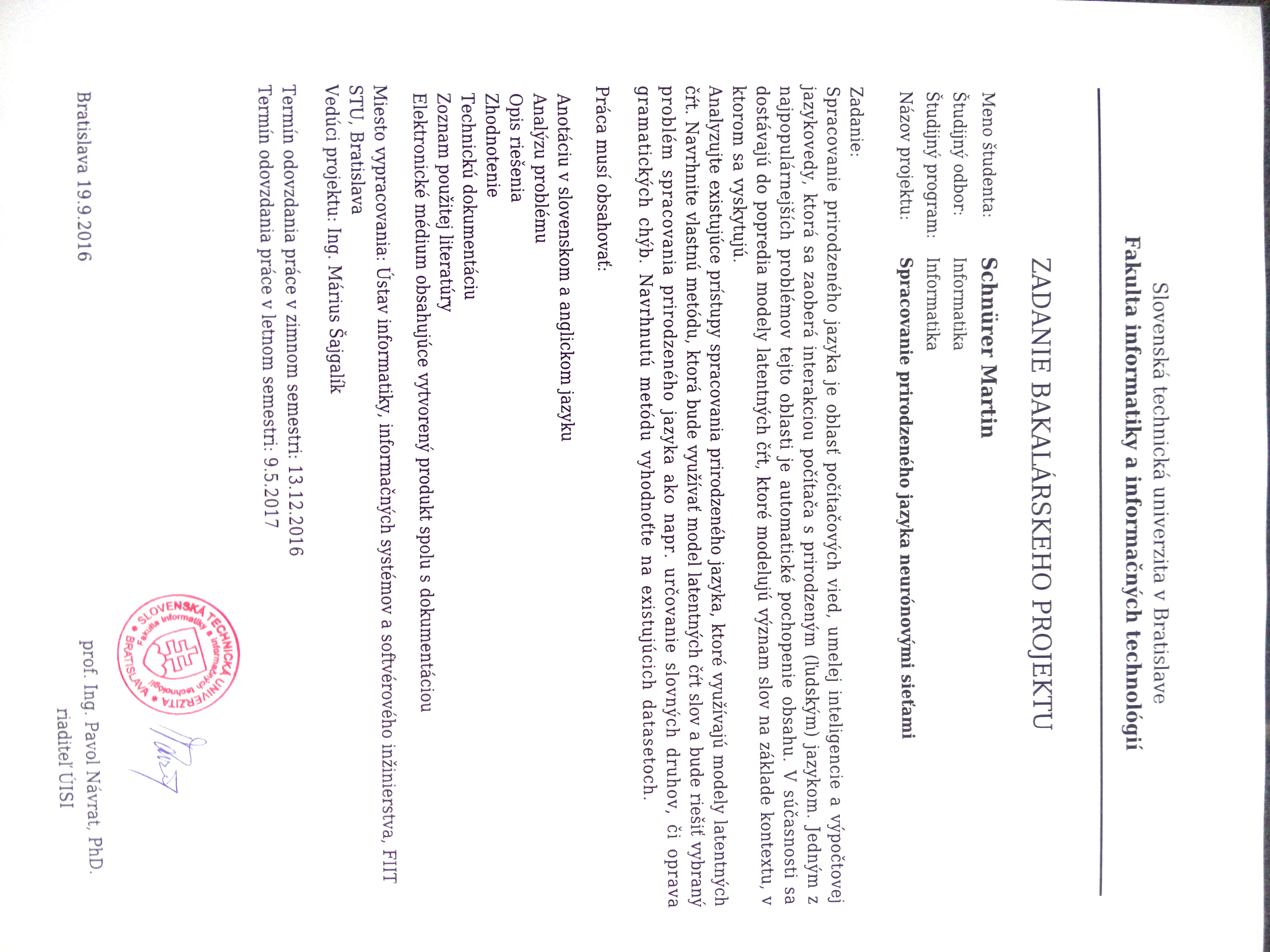
Študijný odbor: 9.2.1 Informatika

Miesto vypracovania: Ústav informatiky, informačných systémov a softvérového inžinierstva,

FIIT STU v Bratislave

Vedúci: Ing. Márius Šajgalík

december 2016



**Anotácia**

Slovenská technická univerzita v Bratislave

FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Študijný program: Informatika

Autor: Martin Schnūrer

Bakalárska práca: Spracovanie prirodzeného jazyka neurónovými sieťami

Vedúci práce: Ing. Márius Šajgalík

december 2016

Rozsiahle texty s určenými slovnými druhmi sú užitočné v rôznych lingvistických výskumoch. Pre človeka je manuálne označovanie dlhých textov zdĺhavá a náročná práca. Pri hrubom a štatistickom určovaní a analyzovaní slov v korpuse dochádza k značným nepresnostiam.

Preto sa táto práca zameria na určovanie slovných druhov neurónovými sieťami, ktorými budeme chcieť zdĺhavý proces označovania urýchliť, pričom chceme zachovať čo najväčšiu presnosť pri určovaní.

**Annotation**

Slovak University of Technology Bratislava

FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES

Degree Course: Informatika

Author: Martin Schnūrer

Bachelor thesis: Natural language processing using neural network

Supervisor: Ing. Márius Šajgalík

2016, December

Text corpora which are tagged with part-of-speech information are useful in many areas of linguistic research. Manual tagging of texts is tedious and difficult work. At raw and statistical analyzing of words in the corpus leads to considerable inaccuracies.

Therefore, this work will focus on part of speech tagging with neural networks as we want to speed up the lengthy identification process, and we want to maintain the greatest possible accuracy.

**POĎAKOVANIE**

Ďakujem môjmu vedúcemu bakalárskej práce Ing. Máriusovi Šajgalíkovi za pomoc, trpezlivosť a cenné rady počas práce na bakalárskom projekte. Tiež chcem poďakovať rodine a priateľom za podporu a motiváciu v doterajšom priebehu štúdia.

# Obsah

1. Úvod
   1. Model neurónu
   2. Architektúry dopredných neurónových sietí
      1. Typy neurónových sietí
      2. Spätné šírenie chyby (backpropagation)
      3. Optimalizačné metódy
      4. Preučenie neurónovej siete
   3. Architektúry rekurentných neurónových sietí
      1. LSTM Bunka
      2. GRU (Gated recurrent unit)
2. Spracovanie prirodzeného jazyka
   1. Analýza kľúčových slov
   2. Syntaktická a sémantická analýza
3. Spracovanie prirodzeného jazyka neuronovymi sieťami
   1. Reprezentácia slova pre NN
   2. Word2Vec
4. Záver
5. Literatúra

**1. Úvod**

Umelá neurónová sieť je dátovo-procesný model inšpirovaný biologickým modelom mozgu z živočíšnej ríše.

Neurónová sieť, ďalej ako skratka NS, je zložená z minimálne 2 vrstiev. Každá vrstva sa skladá z neurónov. Prvá vrstva sa nazýva vstupná vrstva. Posledná vrstva sa nazýva výstupná vrstva. Ak existuje vrstva medzi vstupnou a výstupnou vrstvou, nazývame ju skrytá vrstva.

Najbežnejší model NS obsahuje plné prepojenie medzi každou nasledujúcou vrstvou.

Neurón vo vrstve *n* je spojený s neurónom vo vrstve *n+1* váhou *w****.*** Váha *w* je špecifikovaná ako reálne číslo. Neuróny sú poprepájané medzi sebou do zložitej sieťovej štruktúry. Jednotlivé spoje majú buď excitačný alebo inhibičný charakter.

Základné typy neurónových sietí :

* Dopredná
* Rekurentná

Výhody neurónovej siete:

* Relatívne ľahké použitie
* Použitie pri abstraktných a komplexných problémoch
* Rozoznávanie vzorov
* Pomerne jednoduchá implementácia
* Škálovateľnosť na veľke datasety

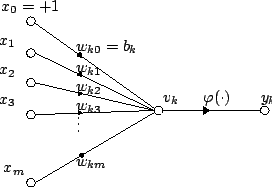
Nevýhody:

* Black box – nevnímame prečo sa deje tak, ako sa to deje
* Ťažká interpretovateľnosť modelu
* Problém inicializácie NS
* Problém pretrénovania
* Dlhé trénovanie, problém stanoviť učiacu konštantu (learning rate)

**1.1 Model neurónu**

Neurón (v informatike) je matematická funkcia koncipovaná podľa modelu biologických neurónov. Umelé neuróny sú základnými jednotkami v umelej neurónovej sieti. Neurón príjma jeden alebo viac vstupov (dendrity) a sčíta ich pre vytváranie výstupu (axon).

Obvykle sú súčty každého uzla normalizované a súčet prechádza funkciami známe ako aktivačné alebo prenosové funkcie. Prenosové funkcie majú zvyčajne tvar sigmoid, ale môžu mať aj podobu ďalších nelineárnych funkcií, po častiach lineárnej funkcie alebo skokovej funkcie. Funkcia thresholding je inšpirovaná k vybudovaniu logického hradla označovaného ako prah.



zdroj : http://neuron.tuke.sk/zvada/statnice/III/01/pictures/perceptron.png

kde:

*x0 … xm* sú vstupné signály

*wk0 … wkm* sú váhy

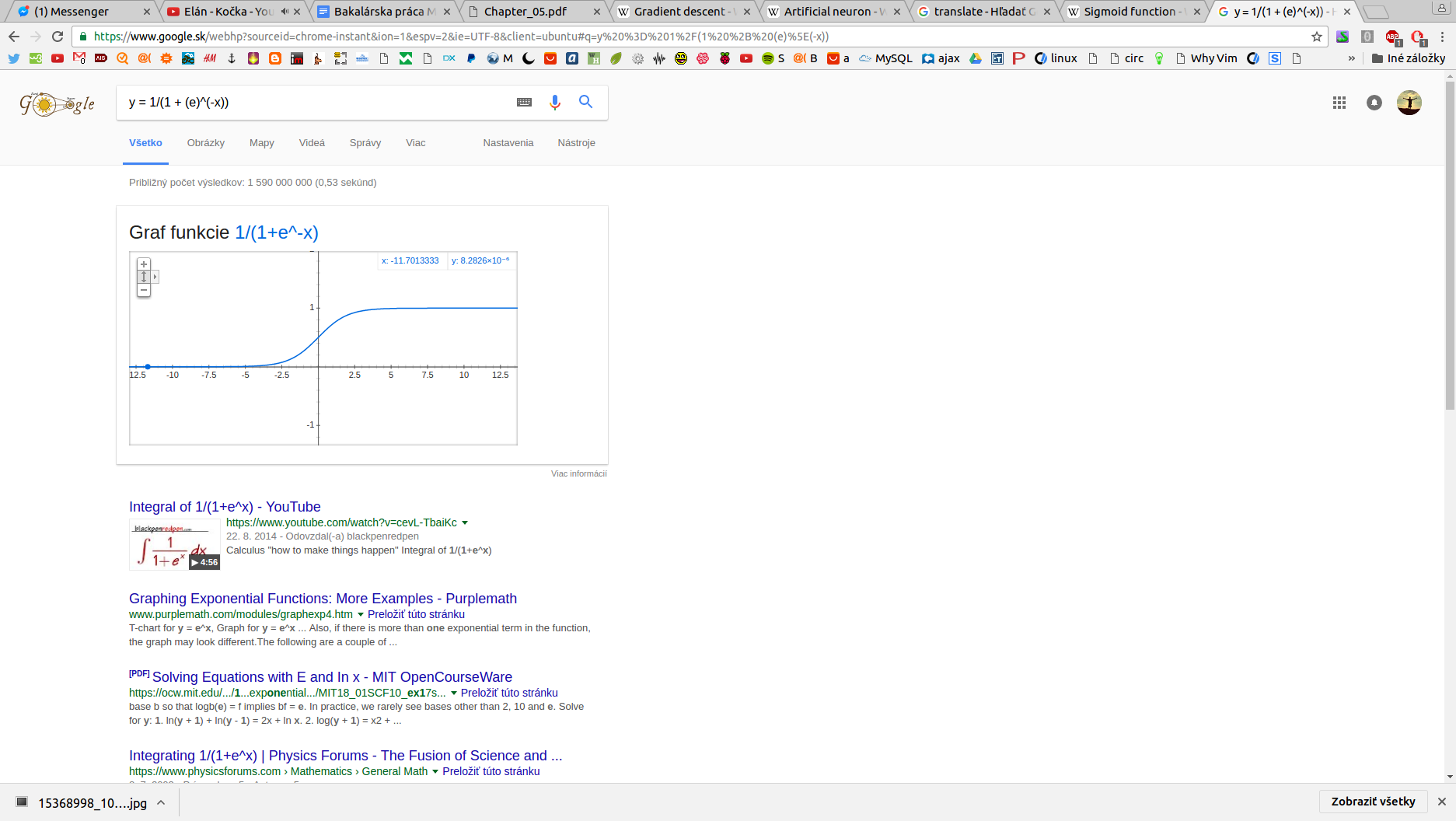
*vk* je suma vstupov násobených príslušnými váhami

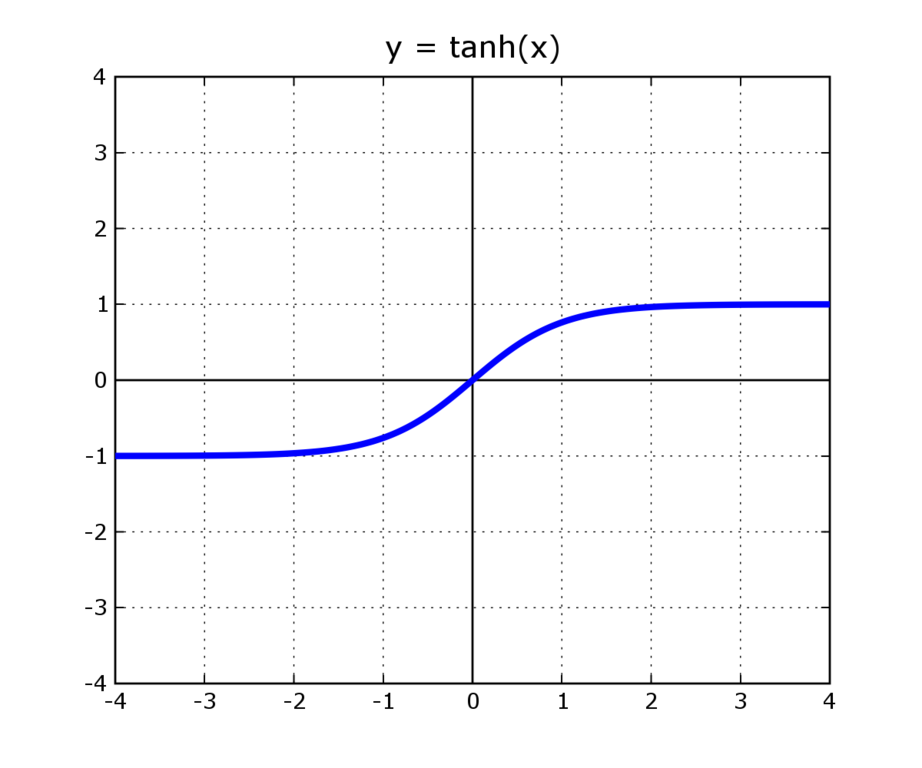
*φ (.)* je prenosová funkcia

y*k* je výstup prenosovej funkcie

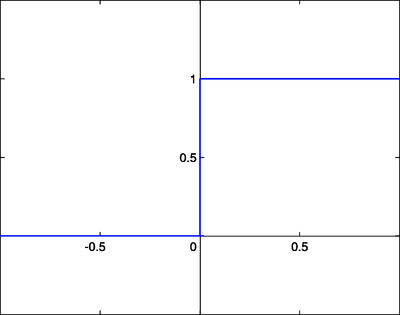
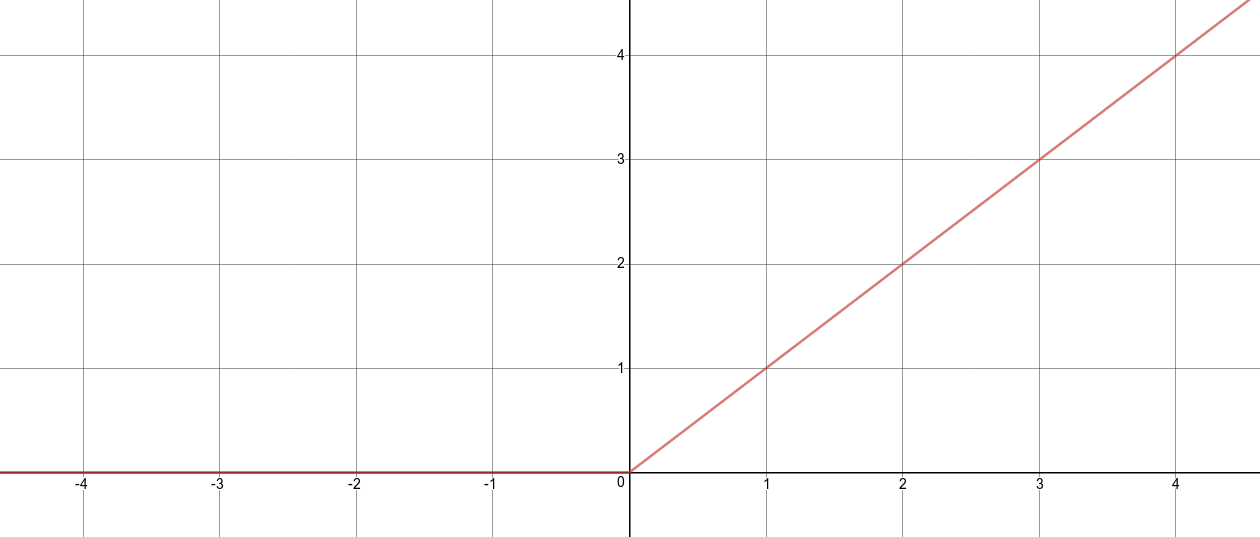
potom výstupom 

Medzi najčastejšie aktivačné funkcie patria:

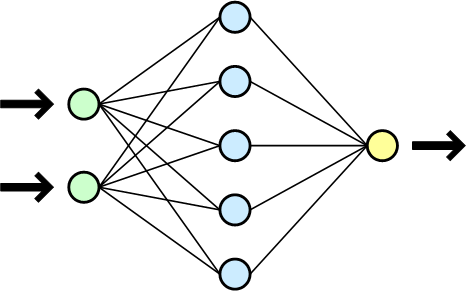
* ***Sigmoid***    
    
  *Zdroj:* obrázok vygenerovaný v prehľadaváči Chromium službou google
* ***Tanh***



*Zdroj:* [*https://i.stack.imgur.com/jW5LR.png*](https://i.stack.imgur.com/jW5LR.png)

* ***Binary step***   
    
  Zdroj: <https://upload.wikimedia.org/wikipedia/commons/thumb/a/ac/HardLimitFunction.png/400px-HardLimitFunction.png>
* **Rectifier function**  
  f (x) = *max(0, x)  
  *
* ***Softmax*** **

**1.2 Architektúry dopredných neurónových sietí**



*obrázok znázorňuje jednoduchý pohľad na základnú štruktúru neurónovej siete s jednou skrytou vrstvou*

*(zdroj:* [*https://en.wikibooks.org/wiki/Artificial\_Neural\_Networks/Neural\_Network\_Basics*](https://en.wikibooks.org/wiki/Artificial_Neural_Networks/Neural_Network_Basics)*)*

Dopredné architektúry neurónových sietí sa využívajú tam, kde je predložená celá vstupná informácia naraz a očakáva sa z tejto vstupnej informácie výstup (tzv. otázka-odpoveď) .

Šírenie informácie prebieha len jedným smerom. V tejto architektúre sú najčastejšie jednotlivé susedné vrstvy plne prepojené. Nevýhodou dopredných architektúr je absencia vlastnosti zapamätať si údaje, resp. kontext údajov z minulosti.

*1.2.1 Typy neurónových sietí*

**Neurónová sieť so spätným šírením**

Najčastejší model neurónovej siete doprednej architektúry, ktorý šíri signál dopredu, využíva učenie s učiteľom a algoritmus spätného šírenia chyby. Je jednoduchý na pochopenie a je aplikovateľný na široké spektrum problémov. Nevýhodou je dlhé trénovanie

**Neurónová sieť s radiálnym základom (angl. Radial Basis function)**

Je neurónová sieť s doprednou architektúrou podobná klasickej sieti so spätným šírením. Avšak RBF architektúra obsahuje 1 skrytú vrstvu s množinou aktivačných funkcií. Aktivačné funkcie sú Gaussové funkcie - výsledkom toho je vynikajúce rozpoznávanie vzorov. Výhodou je rýchlejšie trénovanie. Zužovaním a rozširovaním aktivačnej Gaussovej funkcie môžeme dosiahnuť natrénovanie vzorov pre variabilný počet elementov v datasete.

**Konvolučná neurónová sieť**

Konvolučné neurónové siete sú využívané hlavne pri rozoznávaní vzorov (anlg. pattern recognition). Od klasických dopredných sietí sa líšia tým, že pred samotným učením dochádza k významnému predspracovaniu údajov a vytvoreniu nových príznakov. Väčšina architektúr konvolučných neurónových sietí je tvorená 4 zložkami. Prvou zložkou je konvolučný filter, kde vstupný vektor prejde filtrami konvolúcie. Vzniknutý produkt prechádza aktivačnou vrstvou ReLu (Rectifier Linear Unit) popísaný vyššie ( f(x) = max(x , 0 ) ). Ďalšou vrstvou je tzv. Pooling vrstva, v ktorej nastáva zmenšenie dimenzií (šírka a výška vstupu). Poslednou vrstvou je plne prepojená neurónová sieť, ktorej výstupom je klasifikácia vstupu.

Výhodou predspracovania údajov je výrazne urýchlenie trénovania siete. Jedným z problémov je aj samotné predspracovanie údajov, kedže sa jedná o procesorovo náročné operácie. Výhodou konvolučných neurónových sietí je silná generalizačná schopnosť, robustnosť, nízka pamäťová náročnosť. Sú často používané pri rozpoznávaní obrazov.

*1.2.2 Spätné šírenie chyby (angl. Backpropagation)*

Algoritmus spätného šírenia pozostáva z troch krokov. V prvom kroku sa predloží sieti trénovací prvok pozostávajúci z dvoch zložiek. Kedže sa jedná o učenie s učiteľom, prvou zložkou je vstupný vektor hodnôt a druhou zložkou je vektor očakávaných hodnôt. Keď sa dostane vstupný signál až do výslednej vrstvy, porovná sa výstupný a očakávaný vektor výsledkov a vypočíta sa chyba. V poslednom kroku sa táto chyba šíri späť do predošlých vrstiev a upravujú sa váhy medzi vrstvami. Zmena váhy je úmerná množstvu chyby vyprodukovanej konkrétnym neurónom, z ktorého sa signál predtým šíril. Zmena váhy je regulovaná *učiacou konštantou* (angl. *learning constant*).

**Vzorkové a dávkové učenie**

V obidvoch prípadoch používame algoritmus spätného šírenia chyby. Hlavným rozdielom je, kedy sa aplikuje zmena váh v neurónovej sieti. Pri vzorkovom učení sa aplikuje akonáhle sa vo vrstvách vypočíta chyba - chyba sa pretransformuje do zmeny váh. Pri dávkovom učení sa pri každej vstupnej vzorke spätne vypočítava chyba, ale aplikuje sa až po určitom počte vzoriek. Výslednými chybami je súčet vytvorených chýb predchádzajúcich vzoriek.

Vzorkové učenie je používané, ak sa učíme z nových dát vyprodukovaných prostredím. Je dosiahnutá rýchlejšia zmena váh a tým pádom učenie prebieha rýchlejšie. Dávkové učenie je pomalšie avšak lepšie eliminuje zvyškovú chybu po učení.

*1.2.3 Optimalizačné metódy*

**Postupný zostup (angl. gradient descent)**

Intuitívnou cestou k pochopeniu optimalizácii gradient descent je predstaviť si prameň rieky vyvierajúci na vysokom vrchu. Úlohou rieky, ktorá vyviera na tomto vrchu je dostať sa na čo najnižšie možné položené miesto v priestore. Povrch pohoria je často nerovnomerný a môže nastať situácia, keď rieka uviazne na mieste, ktoré nie je najnižšie. V terminológii strojového učenia ma táto metafora pomenovanie uviaznutie v lokálnom minime a nie je to optimálne miesto, kde sa chceme dostať. Taktiež cesta, ktorou sa vydáme závisí aj od učiacej konštanty. Výška rieky reprezentuje chybu, ktorú chceme minimalizovať. Táto chyba je vlastne rozdielom výstupu vstupného signálu a očakávanou hodnotou výstupu. Našou úlohou je dosiahnuť čo najmenšiu chybu - avšak nesmieme zabúdať na pretrénovanie siete.

**Momentum**

Aby sme sa čo najviac eliminovali výstupnú chybu neurónovej siete, snažíme sa pomaly upravovať váhy tak, aby chyba klesala k čo najmenšej možnej hodnote. Momentum je čiastka, o ktorú upravíme hodnotu novej váhy. Táto čiastka je hodnota zmeny váhy v minulom učení prenásobená konštantou v intervale <0, 1> . Momentum poskytuje rýchlejšie učenie siete a zároveň je ďalším spôsobom ako sa vyhnúť lokálnemu minimu.



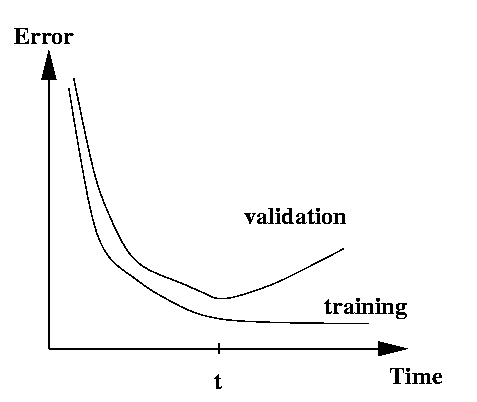
kde *△W(t + 1)* je celková čiastka o ktorú sa stará váha upravý *-μ* je úprava váhy podľa chyby násobená *učiacou konštantou* a *β △W(t)* je momentum

**Adam optimizer (**skr.Adaptive Moment Estimation**)**

Je optimalizačna metóda, ktorá urýchľuje proces učenia neurónovej siete. Metóda je založená na výpočte pohyblivých priemerov a následnými dynamickými zmenami učiacich konštánt (*learning constant*)a momentov (*momentum*).

*1.2.4 Preučenie neurónovej siete*

Preučenie alebo pretrénovanie (angl. overfitting) neurónovej siete je stav, keď sú váhy nadmerne prispôsobené trénovaciemu datasetu, ale pri klasifikácií dát, ktoré nepatria do trénovacej množiny zlyhávajú. Riešením je vhodný výber datasetu a rozdelenie na trénovaciu a validačnú množinu. Na trénovacej množine sa neurónová sieť učí a porovnáva chybu oproti validačnému datasetu. Pomer testovacieho a validačného datasetu je väčšinou 75 : 25. Učenie zastavujeme, keď chyba oproti validačnému datasetu začne narastať.



*Obrázok znázorňuje chybu počas trénovania pri datasete rozdelenom na trénovací a validačný. Trénovanie je dobré zastaviť v čase t, keď validačná chyba začne narastať.*

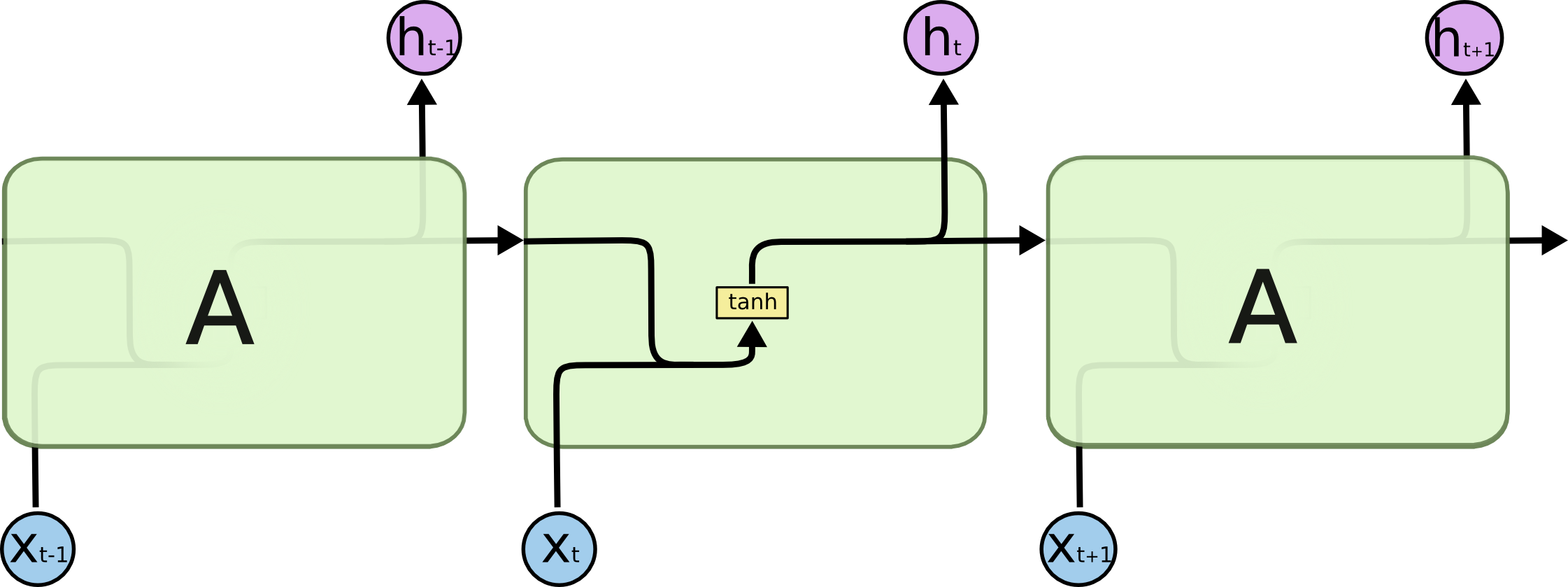
*zdroj: http://www.willamette.edu/~gorr/classes/cs449/figs/earlystop.gif*

**1.2 Architektúry rekurentných neurónových sietí**

Rekurentné siete sa líšia od dopredných neurónových sietí svojou architektúrou. V klasických dopredných neurónových sieťach sa šíri signál vždy dopredu, posúva signál z vrstvy *n* do vrstvy *n+1*. Rekurentné siete majú vlastnosť šíriť signál aj do predošlých vrstiev (aj do svojej vlastnej vrstvy) a zachovávať tak informáciu pre ďalšie vstupy.

Takáto sieť slúži hlavne na zapamätanie si určitej sekvencie dát z minulosti.

Klasická rekurentná sieť si niekedy nedokáže zapamätať dlhodobý kontext, ktorý je na vstupe

*Šírenie výstupného signálu buniek v rekurentnej sieti*

*zdroj: http://colah.github.io/posts/2015-08-Understanding-LSTMs/*

*Príklad.*

*Majme dve vety.* Predpokladajme, že chceme predikovať ďalšie slovo po prvej vete na základe kontextu oboch viet. Čím je ale medzera medzi vetami väčšia, tým ma RNS väčší problém spojiť kontexty týchto dvoch viet.

Na tento účel lepšie poslúži LSTM Cell – bunka s krátkodobo-dlhodobou pamäťou.

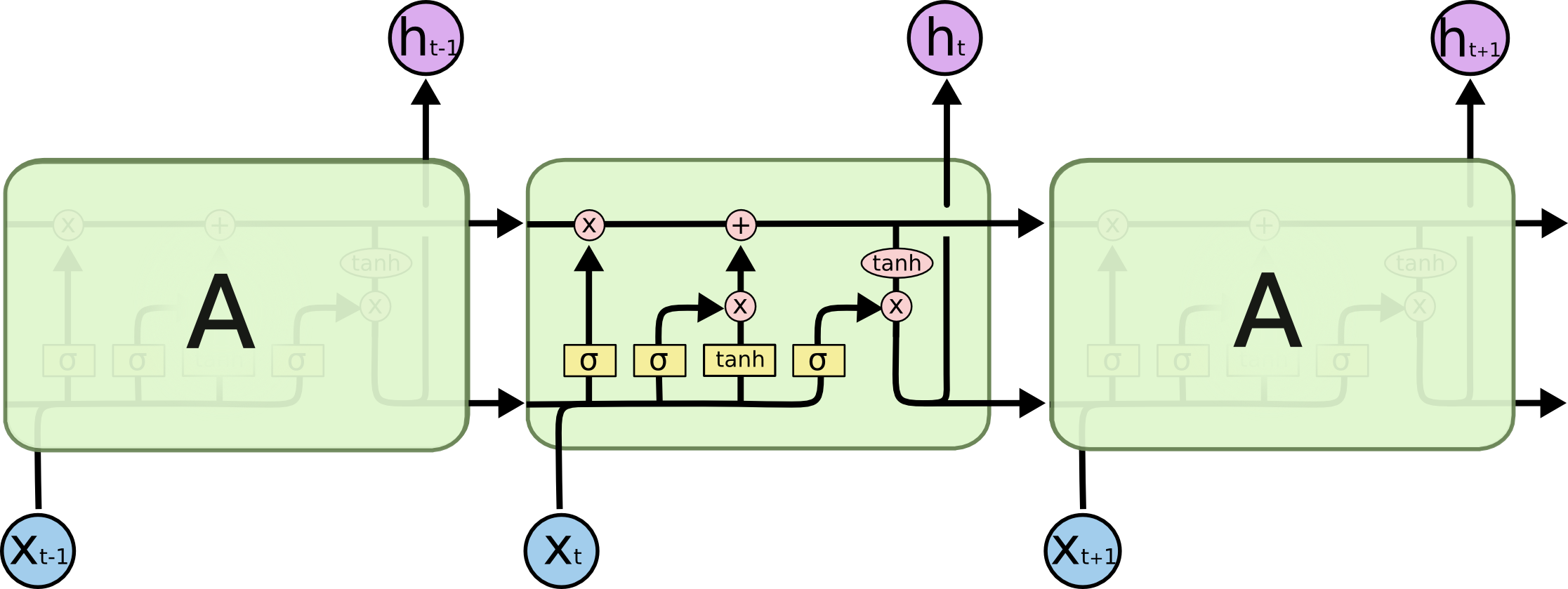
*1.2.1 LSTM Bunka (skr. angl. Long short term memory)*

Rekurentné neurónové siete majú rozsiahle použitie v oblastiach rozpoznávania reči, modelovanie jazyka, preklad alebo značkovanie obrázkov. Narozdiel od dopredných neurónových sietí na svoju správnu funkciu potrebujú znalosť o údajoch z minulosti. Napríklad, ľudia potrebovali na pochopenie aktuálnej snímky (frame) vo filme predošlé snímky, aby správne pochopili kontext prebiehajúceho príbehu.

Problém klasických rekurentných sietí je, že dokážu prepojiť kontext len z krátkodobej minulosti a predikovanie môže byť nepresné, aj keď dlhodobé dáta z minulosti nám jasne vravia, o aký kontext sa jedna. Problém bol ďalej preskúmaný *Seppom Hochreiterom* a *Yoshua-om Bengi-om*

v roku 1994.

LSTM – sú neurónové bunky schopné naučiť sa dlhodobé aj krátkodobé závislosti. Táto bunka bola zavedená *Hochreiterom* a *Schmidhubeom* v roku 1997. LSTM pracujú pozoruhodne dobre pri širokom spektre rozličných problémov a dnes sú rozsiahlo využívané.



*Znázornenie LSTM bunky (Zdroj:* [*http://colah.github.io/posts/2015-08-Understanding-LSTMs*](http://colah.github.io/posts/2015-08-Understanding-LSTMs/) *)*

1.2.2 *GRU (Gated Recurrent Unit)*

GRU je špeciálny typ LSTM, ktorý taktiež riadi tok informácii, ale bez toho, aby bolo nutné použiť pamäťovú jednotku. GRU vystavuje celý srytý obsah, bez akejkoľvek kontroly.

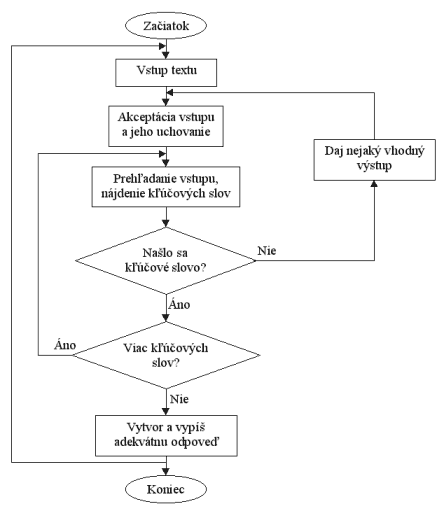
**2. Spracovanie prirodzeného jazyka**

Na spracovanie prirodzeného jazyka sa často používajú dve metódy, a to hľadanie kľúčových slov a syntakticko-sémantická analýza. Tretí prístup ktorý sa používa zriedkavejšie sa nazýva koncepčná závislosť.

## 2.1 Analýza kľúčových slov

Prvé programy na spracovanie jazyka pracovali na základe analýzy kľúčových slov. Takýto program hľadal vo vstupnom reťazci kľúčové slová a frázy. Program bol schopný identifikovať, či “spoznať”, iba vybrané slová a frázy. Keď program rozpoznal nejaké slovo, vykonal podľa toho príslušnú špecifickú akciu.

Prípadne mohol program vykonštruovať odpoveď na základe čiastočného kopírovania kontextu kľúčového slova zo vstupu. Program rozozná špecifický vstup, ktorý potom použije na tvorbu výstupného textu alebo spustenie nejakej akcie.



*Zdroj:*[*http://www2.fiit.stuba.sk/~bielik/courses/zs-slov/clanok/ref-new98/trabalka/276.gif*](http://www2.fiit.stuba.sk/~bielik/courses/zs-slov/clanok/ref-new98/trabalka/276.gif)

Program hľadá v zadanom texte nejaké kľúčové slovo zo svojho slovníka slov prípadne fráz pomocou procesu hľadania vzorov (*pattern-matching*). Každé slovo či slovné spojenie, ktoré má program rozpoznávať, musí v ňom byť vopred uložené.

Pretože chceme, aby program vedel reagovať na ľubovoľnú formu otázky, musíme uložiť všetky potenciálne variácie slova na jeho rozoznanie, prípadne aj jeho synonymá. Napríklad kľúčové slovo môže byť *otec*. Ale keďže ľudia volajú otcov rozličnými inými menami, musíme dať do slovníka aj slová *tatko*, *tato*, oco a pod.

Program zozbiera kolekciu všetkých kľúčových slov, ktoré sa mu podarilo nájsť. Ak program nenájde v texte žiadne kľúčové slovo, jednoducho vyhlási, že nerozumie textu.

Ak však podarilo nájsť nejaké kľúčové slovo či slová, program pomocou nich vykonštruuje výsledok.

Obvykle však program ešte vykoná nejaké ďalšie spracovanie pomocou ktorého vytvorí výstupný reťazec. Napríklad môže použiť časť vstupného textu, či už slovo alebo frázu, ktorú si zapamätá a potom skombinuje s naprogramovanou časťou a vytvorí tak celú vetu.

Zrejme najznámejším príkladom programu pracujúceho popísaným spôsobom je program Josepha Weizenbauma z  MIT nazvaný ELIZA, ktorý vznikol v 60-tych rokoch

## 2.2 Syntaktická a sémantická analýza

Napriek tomu, že prístup pomocou kľúčových slov je často používanou technikou pri spracovaní jazyka, jeho užitočnosť a použitie je značne limitované, pretože nedokáže chápať najrozličnejšie variácie, ktoré sa v jazyku bežne objavujú.

Najpriamejším a najobvyklejším prístupom je vykonanie detailnej jazykovednej analýzy vstupu. Pomocou takéhoto analytického prístupu je možné pre počítač naozaj pochopiť vetu a jej význam.

Aj tento prístup má ale mnoho problémov. Jedným z nich je, že často nevieme rozoznať vzťah medzi slovom a jeho významom. Táto vlastnosť sa prejavuje v dvoch formách, a to homonymia a synonymia. Ďalším problémom je, že rôzne významy môžeme vyjadriť nespočetným množstvom kombinácií slov. Napokon, aj obrovský rozsah slovnej zásoby v danom jazyku predstavuje zložitý proces pre analýzu.

Samotná analýza prirodzeného jazyka sa dá rozdeliť na niekoľko rovín:

* fonologická analýza – rozbor zvukového vstupu (do elektronickej podoby)
* morfologická analýza – analýza tvaroslovia
* syntaktická analýza – identifikácia vetných členov
* kontextová analýza – dáva vetu do súvislosti s kontextom

**3. Spracovanie prirodzeného jazyka neurónovými sieťami**

**3.1 Reprezentácia slova pre neurónovú sieť**

Umelá neurónová sieť a počítače sú schopné pracovať iba s číslami. Takisto aj znaky v počítačoch sú reprezentované jedinečným miestom v tabuľke (ASCII), ktoré označuje nejaké číslo, v tomto prípade v ASCII tabuľke od hodnoty 0 až 255.

Neurónová sieť dokáže pracovať iba s (reálnymi) číslami, preto musíme prísť na spôsob, ako rozumne pretransformujeme slovo ako reťazec znakov do číselnej počítačovej podoby, ktorej neurónová sieť bude rozumieť.

Jednou z možností je pretransformovať slovo do vektoro-priestorového modelu, kde významovo blízke slová sú mapované blízko seba v priestore.

U tohto modelu môžeme pristupovať dvoma spôsobmi :

* Sčítacou metódou - Latentnou sémantickou analýzou
* Prediktívnou metódou – neurónový pravdepodobnostný jazykový model

Sčítacia metóda počíta na základe štatistiky, ako často je nejaké slovo so svojím susedným slovom v rozsiahlom textovom korpuse. Potom pre každé slovo zmapuje tieto sčítané štatistiky do malého vektora.

Známou metódou na mapovanie slov do vektora je Word2Vec.

**3.2 Word2Vec**

Je výpočtovo výhodný prediktívny model pre učenie slovných výrazov na základe surového textu. Učenie nastáva v dvoch zložkách – a to učenie sa na základe „neprerušovanom množstve slov“ (angl. Continuous Bag-of-Words, skr. CBOW) a na základe Skip-Gram modely.

Algoritmicky sú si tieto modely veľmi podobné, s jedným rozdielom, že prvý model (CBOW) predpovedá cieľové slovo na základe viacerých zdrojov (veta, reťazec slov) a naopak, Skip-Gram predpovedá zdroj (vety, slová) iba na základe jedného alebo viacerých slov.

**4. Záver**

V prvej časti bakalárskej práce sme sa venovali analýze oblasti neurónových sietí a samotnej oblasti spracovania jazyka. Hľadali sme vhodnú reprezentáciu slov z hľadiska vstupu pre neurónovú sieť. Preskúmali sme rôzne typy a architektúry neurónových sietí. Dospeli sme k záveru, že pre daný problém spracovania prirodzeného jazyka bude vhodné použiť rekurentnú architektúru s LSTM bunkami.

**4.1 Plán na druhú fázu**

V druhej časti sa chcem zamerať na implementáciu rekurentnej neurónovej siete pomocou knižnice *Tensorflow*. V práci musí byť neodmysliteľne aj extrahovanie potrebných informácii z korpusu textu, ktorý som obdržal. Nový dataset informácií z korpusu by mal byť pamäťovo nenaročný a programom ľahko čitateľný. Chcem preskúmať rôzne architektúry neurónových sietí a porovnať výsledky na základe predošlých testovaných architektúr. Výsledkom druhej fázy by mal byť program, resp. neurónová sieť, ktorá na základe textu, dokáže slovám z tohto textu určiť slovné druhy.

|  |  |
| --- | --- |
| *týždeň* | *cieľ* |
| 1-2 | Spracované vstupy z korpusu (hotový vstup-výstup) |
| 3-7 | Zhotovenie prvej NS, ktorá sa dokáže učiť slovné druhy |
| 8-9 | Testovanie ďalších architektúr |
| 10-11 | Analýza a zhodnotenie, finalizácia programu |

**5. Literatúra**

[1] V. Kvasnička, L. Beňušková, J. Pospíchal, I. Farkaš, P. Tiňo, and A. Kráľ – *Úvod do teórie neurónových sietí* (IRIS, Bratislava, 1997, ISBN 80-88778-30-1)

[2] P. Sinčák and G. Andrejková – *Neurónové siete. Inžinierske aplikácie*  *I a II*

[3] *Kempe,A. (1993).A stochastic Tagger and an Analysis of Tagging Errors. Internal paper. Institute for Computational Linguistics, University of Stuttgart.*