Rekurzívna SOM a automaty

diplomová práca

Tomáš Sirný

univerzita komenského v bratislave
fakulta matematiky, fyziky a informatiky
katedra aplikovanej informatiky

Informatika

Vedúci: Doc. Ing. Igor Farkaš, PhD.

Bratislava, 2008

Čestne prehlasujem, že som túto diplomovú prácu vypracoval samostatne s použitím citovaných zdrojov.

.....

Obsah

Abstrakt

Rekurzívna samoorganizujúca sa mapa (RecSOM) je pomerne nový model neurónovej siete učiacej sa bez učiteľa. Vďaka zaznamenávaniu kontextu aktuálneho vstupu je určená na spracovanie sekvenčných dát. Práca je zameraná na možnosti RecSOM v kontexte učenia niektorých formálnych automatov. Prostriedkom je stochastický jazyk definovaný Reberovou gramatikou a bezkontextový jazyk 0^n1^n vyžadujúci vybudovanie zásobníku - štruktúry zásobníkového automatu. Na príklade oboch postupnosti analyzujeme učiace schopnosti nášho modelu. Model RecSOM sme použili v spojitosti s dvojvrstvovým perceptrónom, ktorého vstupmi boli výstupné aktivity mapy. Model sme trénovali v úlohe predikcie nasledujúcich symbolov v symbolových postupnostiach. Pozorovali sme, že len v prípade Reberovej gramatiky bol model schopný úspešne generalizovať (t.j. správne predikovať symboly i v nových slovách).

Kapitola 1

Úvod

Skúmanie možností a schopností rôznych modelov neurónových sietí patrí k bežným problémom subsymbolovej časti vied zaoberajúcich sa umelou inteligenciou. V tejto práci sa venujeme rekurzívnej verzií samoorganizujúcej sa mapy (RecSOM, [VD02]). Snaží sa zhodnotiť ako sieť reaguje na štruktúrované dáta popísateľné pomocou automatov známych z teórie formálnych jazykov. Štruktúrované dáta predstavujú výzvu pre neurónové siete keďže väčšina starších modelov bola navrhnutá pre nezávislé vstupy. Dáta s určitou štruktúrou pritom predstavujú podstatnú časť informácii v bežnom živote. To podnietilo ďalší výskum a návrhy nových modelov a úprav tých starších. Rekurzívna samoorganizujúca sa mapa je pomerne čerstvým modelom - bola predstavená v roku 2002. V rámci reprezentačnej sily predstavuje najsilnejšiu úpravu pôvodnej SOM.

Motiváciou k skúmaniu tohto modelu boli predchádzajúce práce popisujúce jednak výsledky experimentov s inými modelmi rekurzívnych máp (napr. MSOM, [SH05]) ako aj teoreticky zameraná práca Nicolasa Neubauera [Neu05] snažiaca sa dokázať ekvivalenciu RecSOM so zásobníkovým automatom.

Kapitola 2

Prístupy k spracovaniu sekvencii

Spracovanie dát je jednou zo základných úloh pre výpočtové modely. Dôležitými prostriedkami pre takéto potreby sa stali neurónové siete. Väčšina neurónových sietí však prijíma dáta vo forme vektorov reálnych čísel s konečnou a pevnou dimenziou. Ak sa však pracuje s dátami z reálneho života, často je potrebné značné predspracovanie týchto vstupov. Reprezentujú sa napríklad vo forme konečno-rozmerných vektorov reálnych čísel závislých od daného problému, kategorické premenné sa kódujú pomocou one-hot-encoding. Časove serie sa včlenia do konečno-rozmerného vektorového priestoru pomocou časových okien, chemické dáta sa charakterizujú topologickými indexmi a fyzikálno-chemickými atribútmi. Niektoré typy štruktúr a príklady dát ktoré sa nimi dajú reprezentovať možno zhrnúť:

množiny - objekty na scéne

sekvencie - časové série, priestorové dáta, prirodzený jazyk, text, sekvencia DNA

stromové štruktúry - termy, logické formule, parsovacie a fylogenetické

stromy

grafové štruktúry - chemické formule, scény na obrázku, objekty zložené z rôznych primitívov

Zakódovanie podstatných čŕt týchto foriem dát je časovo náročné a často špecifické pre daný problém. Zároveň sa môže, pri zakódovaní ľubovolnerozmerných dát do vektorov pevnej dimenzie, stratiť určitá informácia. Pri skúmaní týchto problémov sa vyčlenilo viacero rôznych prístupov. Za základné možno považovať rozdelenie na prístupy využívajúce podobnosť a na rekurzívne modely.

2.1 Prístupy založené na podobnosti

Tieto prístupy využívajú doprednú neurónovú sieť. Pre vstupný vektor $\bar{x}$ vypočíta sieť skalárny súčin $\bar{\mathbf{w}}.\bar{x}$, $\bar{\mathbf{w}}$ je váhový vektor, support vector machine (typ lineárneho klasifikátora) zamení Euklidovský vektorový súčin za kernel $k(x^i, x)$, x^i je support vektor.

Funkcionálne siete považujú funkcie za dáta, väčšinou brané v tvare usporiadanej dvojice $(x^i, f(x^i))_{i=1}^{n_f}$. Príkladom takýchto typu dát sú časové série alebo spektrometrické dáta. Pretože na priestor kvadratických integrovateľných funkcii sa dajú aplikovať postupy pre štandardné vektory, objavili sa aj postupy využívajúce lineárne techniky ako principal component analysis alebo lineárne diskriminanty. Pri nelineárnych dopredných neurónových sieťach sú funkcionálne dáta prítomné iba v prvej vrstve a vnútorné ohodnotenie funkcii závisí od konkrétneho modelu.

Modely ISOMAP, ISODAT učeniiace sa bez učiteľa, sa využívajú na vizualizáciu dát a klasterizáciu. Vstupné dáta sa projektujú do nízkorozmerného tvaru tak, aby zostali zachované párové vzdialenosti. Na rovnaké účely sa používajú aj samoorganizujúce sa mapy. Kohonnen navrhol model batch-SOM

použiteľný pre akýkoľvek typ dát nastavením prototypov na generalizovaný medián - bod v trénovacej množine ktorý minimalizuje generalizovanú kvantizačnú chybu.

Kernelové metódy, ktoré sa zameriavajú na spoločné podštruktúry, teda porovnávanie primitív daných štruktúr, sa dajú použiť na rôzne typy dát. Pri sekvenciách sa ráta počet výskytov spoločného podreťazca do určitej dĺžky. Tiež záleží či sa počíta aj čiastočná zhoda, váhovanie zhôd a či musia byť podreťazce susedné. Počítanie kernelu je však náročne a preto sa hľadajú efektívnejšie schémy, napríklad dynamické programovanie alebo sufixové stromy. Podobne sa pri orientovaných acyklických grafoch rátajú (čiastočné) zhody na podstromoch. Zároveň sa môžu použiť aj metódy kernelu pre reťazce, ak ich aplikujeme na prefixovú reprezentáciu stromu.

Ďalším prístupom je využitie sémantickej informácie a následná reprezentácia dát vektorom čŕt, s využitím pravdepodobnostného modelu. Vstupné dáta sa potom reprezentujú konečno-rozmerným gradientom pravdepodobnosti modelu pri daných dátach. Dôležitým prostriedkom v tomto prístupe je Fischerov kernel, ktorý popisuje Riemmanovskú metriku v priestore modelov, a často sa kombinuje so skrytým Markovovským modelom. Prístup využívajúci difúzny kernel rozširuje známe lokálne podobnosti objektov na globálny kernel napodobňujúci difúzny proces. Tento bol použitý na spracovanie textových dokumentov a bioinformatické dáta.

2.2 Rekurzívne modely

Podstatou práce rekurzívnych modelov je rozklad vstupov na základné časti a ich následné spracovanie. Už spracované podštruktúry potom vytvárajú kontext pre ďalší výpočet, podobne ako sa daná štruktúra skladá z jednoduchších častí. Výskum rekurzívnych modelov sa opäť dá rozdeliť na dve oblasti - čiastočne rekurentné systémy a plne rekurentné systémy.

2.2.1 Čiastočne rekurentné systémy

Tieto modely sa využívajú hlavne na spracovanie časových sérii. Základný vzťah možno vyjadriť rovnicou $c_t = f(x_t, c_{t-1})$, kde c_t je stav siete v čase t, f je funkcia ktorú počíta sieť. Toto sa dá priamo rozšíriť na zložitejšie štruktúry - napríklad pre binárny strom s podstromami r a l platí $c_t = f(x, c_r, c_l)$. Hlavným spôsobom učenia týchto modelov je učenie s učiteľom.

Siete učiace sa s učiteľom sa využívajú napríklad na učenie sa jazykov, dokazovanie teorémov, spracovanie textu, predpovedanie štruktúry proteínov, spracovanie obrazu. Na trénovanie siete sa adaptoval algoritmus backpropagation, preto tieto modely majú podobné vlastnosti, ale aj problémy ako obyčajné rekurentné siete. Priestorové dáta a acyklické grafy sa môžu spracovávať ako sekvencie alebo stromy určením poradia spracovania vrcholov. Tím sa však stráca určitá informácia o ich vzťahoch, preto sa objavilo niekoľko opráv snažiacich sa lepšie prispôsobiť týmto dátam - rekurzívne modely pre acyklické grafy, bikauzálne siete pre predpovedanie sekundárnej proteínovej štruktúry.

Rekurzívne modely bez učiteľa sa používajú na vizualizáciu a klastering pre dočasné signály a priestorové dáta. Dynamika týchto sieti sa dá popísať rovnako ako v predchádzajúcom prípade. Voľba funkcie, ktorú počíta sieť, a kontextu je tu však zložitejšia. Kontext c_t sa môže interpretovať ako víťazný neurón alebo profil vzdialeností pre celú mapu. Sem patrí aj model rekurzívnej samoorganizujúcej sa mapy, ktorým sa budeme zaoberať aj v tejto práci.

Neurónové siete na spracovanie štruktúrovaných dát boli navrhnuté Pollackom (1990), ďalej pokračovali Spreduti, Starita a Goeller (1995, 1997). Ukázali, že môžu byť použité na klasifikáciu dátových štruktúr. Využili algoritmus označovaný ako BPTS (backpropagation through structure), rozširujúci už existujúci postup BPTT (backpropagation through time, [RHW86]) pre sekvencie. Zároveň predviedli že tento prístup je výrazne lepší pri nará-

baní so širšími závislosťami ako BPTT, kvôli prirodzenému rozvinutiu cez štruktúru. Efektívnosť BPTT môže byť vážne znížená, rovnako ako pri každom učení sa neurónovej siete, prítomnosťou lokálneho minima v asociovanej chybovej funkcii. Frasconi, Gori a Sperduti nadväzujú prácou ktorá sa zaoberá učením sa príslušnosti orientovaného usporiadaného acyklického grafu na základe lokálneho minima chybového povrchu.

Hinton [Hin90] uviedol koncept distribuovaných redukovaných deskriptorov, ktorý umožnil neurónovým sieťam reprezentovať kompozičné štruktúry. To využili Plate [Pla95] s holografickými redukovanými reprezentáciami a Pollack [Pol90] vo svojom modeli RAAM (recursive auto-associative memmory), ktorý bol následne rozšírený na LRAAM (labelling RAAM). Cadoret [Cad94] použil LRAAM na spracovanie prirodzeného jazyka, pričom dosiahol veľmi dobré výsledky pri klasifikácii distribuovaných reprezentácii syntaktických stromov využiteľných pri rečových aktoch.

2.2.2 Plne rekurentné systémy

Dôležitým poľom pôsobnosti pre tieto modely je nájdenie zhody grafov(GMP). Úlohou je nájsť takú štruktúru zachovávajúcu korešpondenciu medzi vrcholmi 2 grafov tak aby sa maximalizovala podobnostná funkcia. Tento problém je NP-úplný, preto sa hľadajú ďalšie prístupy na jej riešenie. Podobne sú záujmom skúmania aj kombinatorické optimalizačné problémy, s čím začali Hopfield a Tank. Vo svojej práci mapujú funkciu optimalizačného problému na energiu siete. Jeden z prístupov ku GMP je transformácia na problém maximálnej kliky asociovaného grafu, ďalším je snaha o obnovenie štruktúry zachovávajúcej permutačnú maticu.

2.3 Samo-organizujúce sa mapy

V nasledujúcom texte si popíšeme typ neurónovej siete využívajúci princíp samoorganizácie. Mapami sa nazývajú pre svoju vlastnosť topograficky usporiadať vstupy do oblastí prislúchajúcich triedam vstupov. Najprv spomenieme základný model a následne spomenieme hlavné úpravy umožňujúce spracovať štruktúrované vstupy. Model RecSOM, na ktorý sa zameriava táto práca predstavuje najschopnejší model vzhľadom na veľkosť spracovanej informácie.

2.3.1 Pôvodný model SOM

Samo-organizujúce sa mapu navrhol Kohonen [Koh90]. Tvorí ju jedna vrstva neurónov usporiadaná do mriežky. Zvyčajne sa používa dvojrozmerná mriežka, ale je možné vytvoriť aj reťaz - každý neurón okrem okrajových je spojený len s predchádzajúcim a nasledovným. Usporiadanie určuje výsledný priestor do ktorého sa zobrazia vstupy. Vstup (zvyčajne číselný vektor) je privedený do každého neurónu. Vypočíta sa aktivácia neurónov jednoduchým vynásobením vstupného a váhového vektora. Ďalej sa určí víťaz - neurón s váhovým vektorom najmenej sa líšiacim od vstupného. Každý neurón má funkciou okolia určených susedov - neuróny blízke v rámci usporiadania - ktoré budú ovplyvnené jeho aktiváciou. Následne sa upravia váhy víťaza j a jeho okolia Hebbovým pravidlom:

$$\mathbf{w_i}(t+1) = \mathbf{w_i}(t) + \alpha(t) \cdot h(j,i) \cdot (\mathbf{x}(t) - \mathbf{w}(t)) \tag{2.1}$$

v čase t+1, $\alpha(t) \in (0,1)$ predstavuje parameter rýchlosť učenia časom klesajúci k nule, vtedy sa končí učenie. h je funkcia okolia, založená na vzdialenosti neurónov. Obvykle používanou je miera vzdialenosti Manhattan - suma absolútnych hodnôt rozdielov súradníc, alebo gaussovské okolie. Do okolia patria neuróny s vzdialenosťou menšou ako zadaná hranica závislá od času - po-

stupne sa eliminuje vplyv susedov. Samo-organizácia sa prejavuje tým, že neuróny topograficky si blízke (v rámci štruktúry v ktorej sú usporiadané), reagujú po natrénovaní na vstupy s podobnými vlastnosťami. Nevyužíva sa učiaci signál, preto hovoríme o "samo"-organizácii. SOM teda sama extrahuje dôležité črty zo vstupov a usporadúva ich podľa týchto vlastností. Predstavuje teda zobrazenie zachovávajúce topológiu. Model má aj biologickú plauzibilitu. V mozgu môžeme nájsť topografické mapy reprezentujúce napríklad povrch tela, sluchové mapy etc. Experimentálne sa zistilo, že tieto mapy sa formujú do konečnej podoby až v dôsledku zmyslovej skúsenosti, samoorganizovane.

2.3.2 Rekurzívne úpravy

Model samorganizujúcej sa mapy ponúka účinný prístup k spracovaniu nezávislých dát. Zároveň však poskytuje platformu pre úpravy, umožňujúce spracovávať aj štruktúrované dáta. Nižšie uvedené modely vychádzajúce zo SOM, využívajú rovnaký princíp. Sieť si v určitej forme zachováva kontextovú informáciu, teda charakteristiku stavu siete v predchádzajúcom kroku. Takto sa vo výbere aktuálneho víťaza prejavia reprezentácie predchádzajúcich vstupov, čím sa zabezpečí reakcia na sekvenciu ako celok a víťazný neurón reprezentuje nielen momentálny vstup. Tento postup je inšpirovaný rekurentnými modelmi pre časové dáta ako napríklad RTRL (Real Time Recurrent Learning). V nich sú prítomné kontextové neuróny, do ktorých sa kopíruje aktivácia výstupnej vrstvy a v ďalšom kroku ovplyvňujú skryté neuróny. V modeloch SOM funkciu tejto vrstvy preberá zapamätaná kontextová informácia. Konkrétna forma sa v jednotlivých modeloch líši svojou zložitosťou, čo determinuje aj reprezentačnú schopnosť jednotlivých typov. Modely si uvedieme v poradí podľa veľkosti kontextovej informácie.

Temporal Kohonen Map

TKM je prvým z modelov, navrhnutý Chappelom a Taylorom [CT93]. Využíva neuróny ako deravé integrátory pri implementácii rekurencie. Majme sekvenciu $(x^1, \ldots, x^t), x^i \in R^n$. Vzdialenosť neurónu i, s váhovým vektorom $\mathbf{w^i}$ je daná vzťahom

$$d_i(t) = \sum_{j=1}^{t} \alpha \cdot (1-\alpha)^{(t-j)} \cdot \|\mathbf{x}_j - \mathbf{w_i}\|^2 \tag{2.2}$$

pričom α je konštanta určujúca rýchlosť učenia. Vzdialenosť teda vyjadruje ako daný neurón aproximuje aktuálny vstup, ako aj exponenciálne váhované všetky predchádzajúce. TKM sa trénuje klasickým Hebbovým učiacim pravidlom, konkrétne

$$\Delta\mathbf{w_i} = \gamma \cdot h_\sigma(d_N(i, I)) \cdot (\mathbf{x}_i - \mathbf{w_i}) \tag{2.3}$$

V prípade tohto modelu predstavuje kontext aktivácia víťazného neurónu v predchádzajúcim kroku.

SOMSD

Samo-organizujúca sa mapa pre štruktúrované dáta (SOMSD) bola pôvodne navrhnutá na spracovanie stromových štruktúr, ale dá sa použiť aj pre sekvenčné dáta. Ak faktor vetvenia stromu k nastavíme na 1, dostávame sa k sekvencii. Každý neurón má vlastný kontextový vektor $\mathbf{c}_i$. Pri vlastnom výpočte sa využíva index víťazného neurónu ako reprezentácia stavu siete v predchádzajúcom kroku. Ak je mapa usporiadaná do 2-rozmernej mriežky, kontextom sú súradnice víťazného neurónu. Pre vzdialenosť neurónu od vstupu platí

$$d_i(t) = \alpha \cdot \|\mathbf{x}_i - \mathbf{w_i}\|^2 + \beta \cdot d_G(I_{t-1}, \mathbf{c}_i) \tag{2.4}$$

I_{t-1} je index víťaza v minulom kroku. d_G označuje vzdialenosť neurónov v mriežke, zvyčajne Euklidovská.

Merge SOM

MSOM alebo Merge SOM bola navrhnutá ako vylepšenie predchádzajúceho modelu, SOMSD. Kontext vo forme indexu víťazného neurónu je informácia viazaná na schému indexovania neurónov v sieti, čo sa snaží MSOM odstrániť. Kontext v sebe spája lineárnou kombináciou váhový a kontextový vektor minulého víťaza. Dvojica vektorov $(\mathbf{w}_i, \mathbf{c}_i) \in R^2 n$ neurónu i sa Hebbovým pravidlom upraví do smeru aktuálneho vstupu a kontextu. Vzdialenosť neurónu je j v kroku t:

$$d_i(t) = (1-\alpha) \cdot \|\mathbf{x}_t - \mathbf{w}_i\|^2 + \alpha \cdot \|C_t - \mathbf{c}_i\|^2 \tag{2.5}$$

s kontextovým deskriptorom C_t, počítaný podľa vzťahu

$$C_t = (1-\beta) \cdot \mathbf{w}_{I(t-1)} + \beta \cdot c_{I(t-1)} \tag{2.6}$$

ak $I(t-1)$ je index predchádzajúceho víťaza. Víťazom sa stáva neurón s minimálnou vzdialenosťou.

Spracovanie stromov

Spomenieme schopnosť rekurzívnych modelov spracovať stromové štruktúry. Stromami totiž možno reprezentovať matematické a chemické vzorce, herné stratégie alebo stavbu viet. V [HMSS04] bol navrhnutý všeobecný postup aplikovateľný na modely RecSOM, TKM, SOMSD. Výsledný prístup bol nazvaný GSOMSD (general SOM for structured data). Nech L je množina návestí, často vektorový priestor reálnych čísel. Potom sekvecie nad L zapisujeme $[a_1, a_2, \ldots, a_t]$, prázdnu sekvenciu ako [ ]. Sekvencie môžeme považovať za stromy so stupňom vetvenia 1. Prázdny strom označujeme ako ξ. Nech strom t so stupňom vetvenia k pozostáva z koreňa s návestím $a \in L$ a k podstromov $t_1, \ldots, t_k$. Potom $t = a(t_1, \ldots, t_k)$ a uzol a je rodičom koreňov stromov $t_1, \ldots, t_k$. Sekvenciu $[s_1, \ldots, s_t]$ si môžeme predstaviť ako strom $s_t(s_{t-1}(\ldots(s_1(\xi)\ldots)))$ s koreňom s_t.

Spracovanie stromu prebehne v smere od listov ku koreňu. Postupne sa vytvárajú reprezentácie jednotlivých podstromov, ktoré sa používajú pri výpočte reprezentácií uzlov bližšie ku koreňu. Neurón váhovaný trojicou $(\mathbf{w}, c_1, c_2)$ reprezentuje strom $a(t_1, t_2)$, ak w je reprezentácia a, c_1, c_2 reprezentujú podstromy t_1, t_2. Kontextovú informáciu predstavujú c_1, c_2. Návestia a, w sa porovnávajú mierou d_W. Aktivácie neurónov v mape po spracovaní celého podstomu t_i sa transformujú pomocou zobrazenia rep. Tým sa získa reprezentácia daného stromu. Tá sa porovnáva s kontextom c_i pomocou d_R.

Kapitola 3

Teoretické podklady

Vzťah medzi neurónovými sieťami a automatmi bol skúmaný už viackrát. Teoretický výskum ukázal že dynamický systém v podobe rekurentnej neurónovej siete (Recurrent Neural Network - RNN) môže simulovať univerzálny Turingov stroj [Pol87]. Z empirických výskumov spomeňme aspoň niektoré. Schopnosti RNN so skrytou vrstvou učenej algoritmom BPTT (backpropagation through time, [RHW86]) boli popísané v [RWE99]. Experiment bol zameraný na jazyk $a^n b^n$ (iné označenie pre nami skúmaný jazyk $0^n 1^n$). Sieť, trénovaná na slovách s n najviac 11 bola schopná generalizovať a naučiť sa rozpoznávať slová až do n = 16. Ďalším príkladom pre tento jazyk môže byť rekurentná BCM sieť [TSB00] a skúmanie jej schopnosti v [Sto01]. RBCM sieť dokázala zovšeobecniť z $n \leq 8$ pri trénovaní na $n \leq 12$ pri testovaní.

Aj rekurzívnym samoorganizujúcim sa modelom bola venovaná pozornosť s ohľadom na formálne automaty. Model Merge SOM bol testovaný [SH05] na slovách Reberovej gramatiky. Sieť so 617 neurónmi si vytvorila dostatočné reprezentácie na rozpoznanie aj dlhších slov. Ekvivalenciou medzi zásobníkovým automatom a modelom RecSOM z teoretického hľadiska sa zaoberal Neubauer [Neu05]. Ukázal ako umožniť RecSOM simulovať prechodovú funkciu zásobníkového automatu. Ide však o riadený výpočet, pri ktorom sú váhy

Model	FMM	FSA	DPDA
TKM	√	×	
SOMSD		√	×
MSOM		√	×
RecSOM			

Obr. 3.1: Dokazané ekvivalencie pre rekurzívne modely podľa [Neu05]

umelo nastavované. Zásobník je zakódovaný do reálneho čísla a uložení do váh víťazného neurónu pomocou vytvorených operátorov. Ide teda o výpočet, ktorý nevznikne samotným trénovaním siete. Tieto výsledky boli motiváciou pre naše experimeny. Snažili sme preskúmať možnosti ktoré modelu dáva jeho prirodzené fungovanie.

3.1 RecSOM

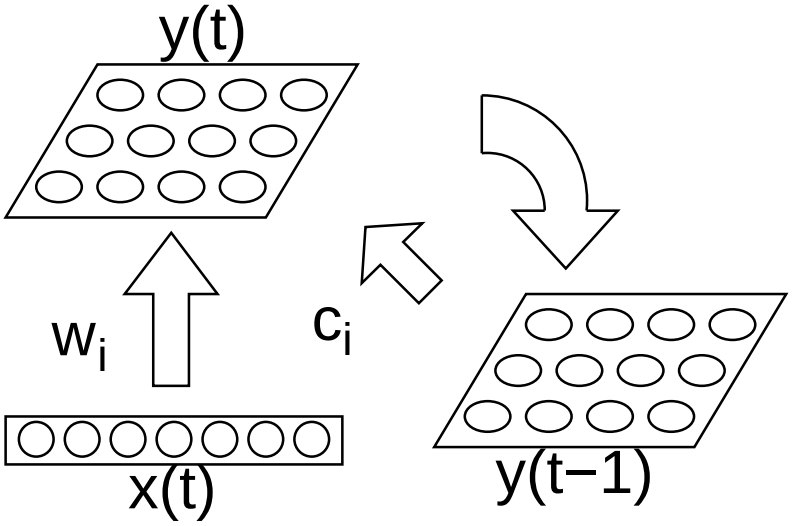
Rekurzívnu SOM predstavil Voegtlin [VD02] ako model na spracovanie časových dát. Ako dodatočnú informáciu používa aktivácie všetkých neurónov, teda kópiu stavu mapy po spracovaní predchádzajúceho vstupu. To dáva modelu väčšiu reprezentačnú schopnosť v porovnaní s predchádzajúcimi modelmi, pretože má k dispozícii viac informácii. Na druhej strane je však aj väčšia náročnosť výpočtu.

Neurón i okrem normálneho vektora w_i^x určujúceho váhu vstupu aj vektor w_i^y váhujúci kontextovú informáciu, vektor minulých aktivácii. Pre jeho aktiváciu v čase t platí:

$$y_i(t) = \exp(-\alpha\|x(t) - w_i^x\|^2 - \beta\|y(t-1) - w_i^y\|^2) \tag{3.1}$$

s konštantami α, β. Víťazom sa stáva neurón s najväčšou aktiváciou, nech má index k. Potom sa váhy upravujú pravidlami:

$$\Delta w_i^x = \gamma h_{ik}(x(t) - w_i^x) \tag{3.2}$$



Obr. 3.2: Schéma RecSOM. Spolu s aktuálnym vstupom $\bar{x}(t)$ sa berie do úvahy aj kópia siete z minulého kroku net(t-1).

$$\Delta w_i^y = \gamma h_{ik}(y(t-1) - w_i^y) \tag{3.3}$$

pričom γ je miera učenia a k_{ik} je funkcia okolia, definovaná Gaussovou funkciou Euklidovskej vzdialenosti $d(i,k)$ medzi neurónmi i a k vzťahom:

$$h_{ik} = \exp(-d(i,k)^2/\sigma^2) \tag{3.4}$$

a σ vyjadruje šírku Guasiánu. Algoritmus učenia RecSOM je ako vidieť iba postup pre SOM uplatnený na oba váhové vektory.

Voegtlin použil na lepšie hodnotenie vlastností mapy pojem receptívne pole pre každý neurón. Receptívne pole neurónu predstavuje najdlhší spoločný sufix postupností pri ktorých bol daný neurón víťazom. Poskytuje nám teda lepší prehľad o tom, ako reaguje sieť. Zároveň sa dá na jeho základe určiť veľkosť pamäte mapy, veličinou kvantizačná hĺbka RF (receptive field) vyjadrenou vzťahom

$$QD = \sum_{i=1}^{N} p_i l_i \tag{3.5}$$

kde p_i je pravdepodobnosť receptívneho poľa neurónu i a l_i jeho dĺžka.

V [TFvM06] bola navrhnutá miera zachovania topografického usporiadania TP (topography preservation measure). Najprv sa vypočíta pre každý neurón i dĺžka najdlhšieho sufixu spoločného pre receptívne polia R_i daného neurónu a jeho okolia, označíme ho $l(R_i)$. TP potom predstavuje priemernú hodnotu týchto sufixov v mape:

$$TP = \frac{1}{N} \sum_{i=1}^{N} l(R_i) \tag{3.6}$$

3.2 Viacvrstvový perceptrón

Viacvrstvový perceptrón (MLP - Multi-level perceptron) je zovšeobecnením jednoduchého perceptrónu. Má väščie klasifikačné schopnosti, dokáže rozlíšiť triedy lineárne neseparovateľné (napr. XOR problém). Dá sa použiť aj ako univerzálny aproximátor pre spojité funkcie. Pozostáva z viacerých neurónov v aspoň jednej skrytej a jednej výstupnej vrstve. Neuróny majú nelineárnu aktivačnú funkciu (logistická, softmax, tanh). Vrstvy sú plne prepojené. Učenie MLP prebieha s učiacim signálom a so spätným šírením chyby. Proces trénovania prebieha v dvoch krokoch: výpočet výstupu siete v doprednom smere a šírenie chyby v spätnom smere. Nech h_i je výstup skrytých neurónov a y_i výstupných neurónov. Potom pre dopredný výpočet platí:

$$h_k = f(\sum_{j=1}^{n+1} v_{kj} x_j) \tag{3.7}$$

$$y_i = f(\sum_{k=1}^{q+1} w_{ik} h_k) \tag{3.8}$$

s aktivačnou funkciou logistickou $f(net) = 1/(1+e^{-net})$. Softmaxová funkcia je vhodná ak je cieľom zaradiť vstup do niektorej z kategórii. Po výpočte aktivácie prebehne výpočet chyby pre váhy medzi skrytou a výstupnou vrstvou

$$w_{ik}(t+1) = w_{ik}(t) + \alpha \delta_i h_k \tag{3.9}$$

$$\delta_i = (d_i - y_i) f_i' \tag{3.10}$$

a pre váhy medzi vstupmi a skrytou vrstvou:

$$v_{kj}(t+1) = v_{kj}(t) + \alpha \delta_k x_j \tag{3.11}$$

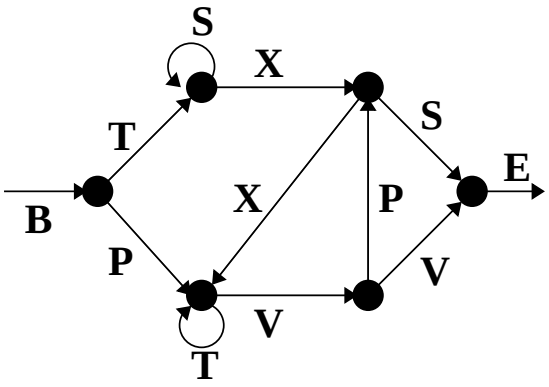
$$\delta_k = (\sum_i w_{ik} \delta_i) f_k' \tag{3.12}$$

3.3 Automaty

Jazyky podľa Chomského hierarchie, spolu s príslušnými gramatikami a automatmi, sa ukázali ako dôležitý formalizmus pri teoretickom chápaní počítačov. Pomohli nielen kategorizovať ich výpočtovú silu, ale aj vytýčiť jej hranice. Automaty a Turingove stroje predstavujú akceptačný prístup k práci s triedami jazykov, ako opak generatívneho prístupu gramatík. Pri skúmaní vlastností samoorganizácie sa zatiaľ pracovalo s regulárnymi a bezkontextovými jazykmi, jednoduchšími triedami v hierarchii. Základom automatov je konečná množina stavov medzi ktorými môžu prechádzať. Automat teda číta vstupnú sekvenciu znakov a podľa predpísaných pravidiel mení stav v ktorom sa nachádza.

3.3.1 Konečný automat

Konečný automat (FSA - Final State Automaton) predstavuje najjednoduchší automat. Jeho výpočtová sila je rovná regulárnej gramatike, dokáže teda akceptovať regulárne jazyky. K dispozícií má iba konečný počet stavov a prechodovú funkciu určujúcu pravidlá pre zmenu stavu. Automat je najprv v začiatočnom stave. Ak sa po prečítaní celého slova ocitne v jednom z akceptačných stavov, akceptuje dané slovo. Ak posledný stav nie je akceptačným, slovo nie je akceptované. Príkladmi regulárnych jazykov sú $\{a, aa, aaa, \ldots\}, \{ab, abab, ababab, \ldots\}$ ale aj $\{a^m b^n, m, n > 0\}$.



Obr. 3.3: Automat pre Reberovu gramatiku

3.3.2 Reberova gramatika

Jazykom, pre ktorý možno nájsť akceptujúci konečný automat je aj Reberova gramatika. Vytvoril ju psychológ Arthur Reber [Reb67] ako nástroj pri svojom skúmaní implicitného učenia - subjekt si pri ňom nie je vedomý jednak procesu, učenia ani jeho výsledku. Tento jazyk sa často využíva pri výskume učenia u zdravých ľudí alebo duševne chorých, s poškodením mozgu ale aj umelej inteligencie. Pri ľuďoch prebiehajú pokusy tak, že sú predložené slová z jazyka, bez zmienky o existencií pravidiel ich tvorby. Subjekt má následne rozhodnúť ktoré z ďalších poskytnutých slov patria do jazyka. Reberove pokusy ukázali, že človek si neuvedomí konkrétne pravidlá, jednoducho má pocit že slová patria k sebe.

Abeceda jazyka pozostáva zo 7 písmen, slovo začína písmenom B, končí E. Jednoducho sa dá popísať konečným stochastickým automatom so 6 stavmi. V každom stave sú 2 možné nasledujúce stavy, každý s 50% pravdepodobnosťou. Z obrázku 3.3.2 je zároveň vidieť že, každé písmeno môže vzniknúť prechodom z dvoch rôznych stavov. Výber dvojice nasledujúcich písmen teda závisí aj od predchádzajúceho stavu.

3.3.3 Zásobníkový automat

Zásobníkový automat (PDA - Push-down Automaton) má oproti FSA navyše zásobník - potenciálne nekonečnú dátovú štruktúru typu LIFO (last in - first out). Pri prečítaní znaku automat zároveň vyberie znak zo zásobníka a podľa toho sa rozhodne či zmení stav a koľko a aké znaky zapíše na zásobník. To mu umožňuje zaznamenávať si informácie nielen zmenou stavu ale aj pridávaním znakov na zásobník. PDA akceptuje buď akceptačným stavom alebo prázdnou pamäťou. Akceptácia stavom prebieha rovnako ako pri FSA, ak sa po prečítaní posledného znaku nachádza v akceptačnom stave. Na obsahu zásobníka nezáleží. Naopak akceptácia prázdnou pamäťou nastane ak automat vyberie posledný znak zo zásobníka, nezávisle na stave. Oba spôsoby sú ekvivalentné výpočtovou silou a pre každý PDA akceptujúci jedným spôsobom existuje ekvivalentný PDA akceptujúci druhým spósobom.

3.3.4 Jazyk 0^n1^n

Ako vstupnú množinu sme zvolili známy bezkontextový jazyk 0^n1^n, obsahujúci slová $\{01, 0011, 000111, \ldots\}$. Často sa totiž používa ako príklad jazyka na ktorý už konečné automaty nestačia. Konečný automat nemá prostriedky na to, aby si mohol zapamätať počet prečítaných núl. K dispozícii ma totiž iba konečný počet stavov. FSA teda dokáže skontrolovť slová patriace do jazyka iba do určitej hodnoty n. Možnosť zapamätať si neobmedzené množstvo informácii poskytuje až zásobník. Do neho sa pri čítaní núl pridáva jeden znak slúžiaci ako počítadlo. Pri čítaní jednotiek sa naopak daný znak vyberá z vrchu zásobníka. Automat akceptuje slovo vtedy, ak prečítaním poslednej jednotky vyberie zo zásobníka posledný znak. Zároveň vychádza z dát na ktorých už prebiehali experimenty s modelom RecSOM. Voegtlin použil sekvenciu symbolov "0" a "1" produkovanú dvoj-stavovým automatom. Pravdepodobnosť prechodu z "0" na "1" bola 0,3 a pravdepodobnosť

opačného prechodu z "1" na "0" bola 0,4.

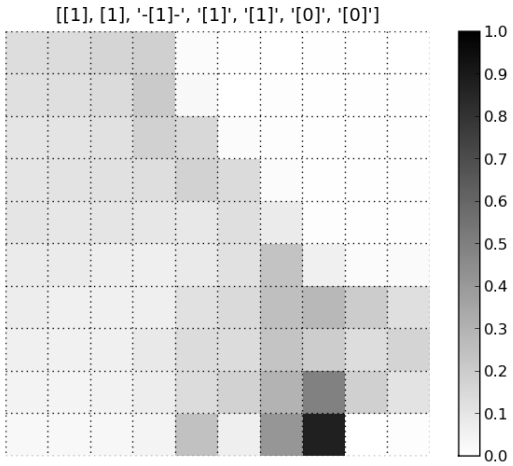
Kapitola 4

Experimenty

4.1 Metódy a ciele

Našim hlavným cieľom bolo zistiť, do akej miery je schopný model RecSOM spracovať sekvencie z konkrétneho jazyka. Pre automat, schopnosť spracovať jazyk znamená rozhodnúť pre každé slovo príslušnosť do jazyka prostredníctvom stavu v ktorom sa ocitne po prečítaní slova. Našim cieľom je dosiahnuť podobnú reakciu od nášho modelu. V tomto prípade je však odpoveď zložitejšia. Keďže testujeme sieť založenú na samoorganizácii, výstupné dáta sú vo forme mapy aktivácii siete. Táto forma však nemôže sama o sebe poskytnúť jednoznačné odpovede na naše otázky. Ponúka nám iba reprezentáciu aktuálneho vstupu ako mapu. Preto sme na lepšie rozhodovanie použili viacvrstvový perceptrón ako kategorizátor reprezentácii výstupov z RecSOM, keďže poskytuje jednoduchšie spracovateľnú formu výsledku.

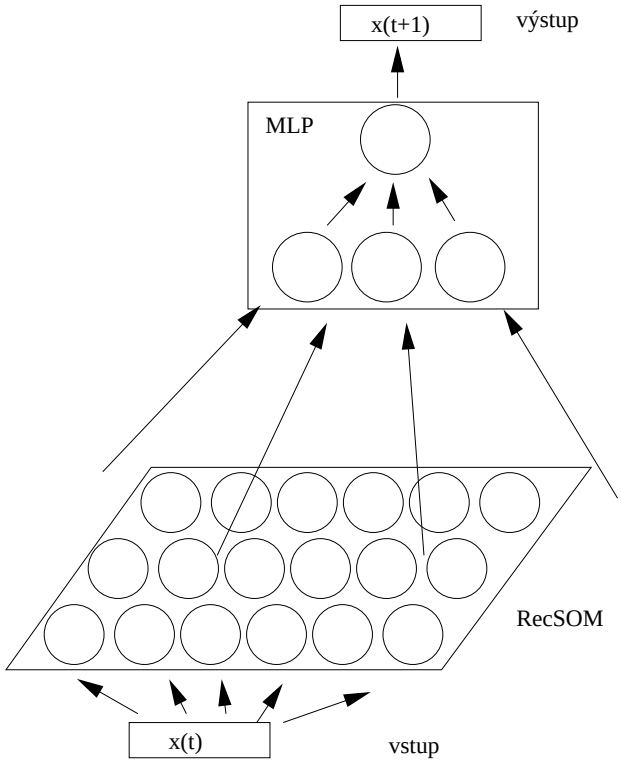
S tým súvisí aj voľba úlohy, pomocou ktorej chceme sieť testovať. Rozhodli sme sa pre bežnú predikčnú úlohu - sieť má predpovedať nasledujúci symbol. Predikciu môžeme považovať za porovnateľnú akceptácii, respektíve predikčná úloha zahŕňa aj akceptačnú.. Napríklad ak sieť na aktuálny vstup "1" odpovie "0", predpovedá koniec slova, čo je porovnateľné s akceptačným



Obr. 4.1: Mapa aktivácii RecSOM

stavom automatu. Vrámci pokusov s jednotlivými jazykmi sme ešte rozlišovali viaceré skupiny vstupov, podla konkrétnej skúmanej schopnosti siete.

Pokus vždy pozostával z viacerých fáz. Pre lepšiu orientáciu označíme jednotlivé množiny dát kódmi. Najprv sme natrénovali sieť RecSOM na vstupných dátach - slovách z jazyka bezprostredne nasledujúce za sebou - množine *Train*. Natrénovanej sieti sme predložili rovnaké dáta ako pri trénovaní, čím sme získali reprezentácie *Repr1* pre tieto dáta. Následne sme množinu *Train* spolu s *Repr1* použili na natrénovanie MLP. Vstupmi boli aktivácie siete v *Repr1*, učiacimi signálmi v symboly v množine *Train1*, avšak posunutú o jeden znak dopredu. MLP sa teda pokúša rozpoznať či v sa reprezentácii aktuálneho vstupu prejavuje očakávanie nasledujúceho znaku. Testovanie MLP prebiehalo na ďalšej sade *Test* a *Repr2*.



Obr. 4.2: Architektúra použitej siete. Mapu aktivácii pri aktuálnom vstupe $x(t)$ spracuje viac-vrstvový perceptrón, výsledkom je predikcia nasledujúceho vstupu $x(t+1)$

4.2 Reberova gramatika

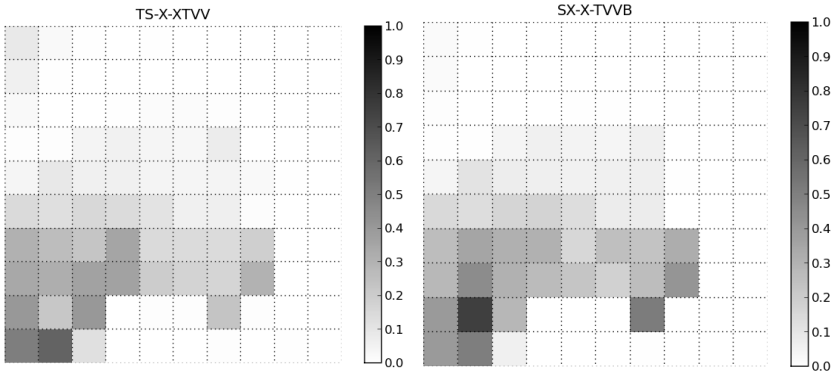
Trénovacími aj testovacími dátami boli slová vygenerované Reberovou gramatikou. Pri testovaní sme predkladali sieti slová s väčšou dĺžkou a zároveň nepoužité pri trénovaní. 6 možných symbolov sme reprezentovali pomocou 'one-hot' kódovania - vektormi (1,0,0,0,0,0) s "1" na rôznom mieste pre každý symbol. Výstupná vrstva MLP potom pozostávala zo šiestich neurónov so softmaxovou aktivačnou funkciou používanou pri kategorických výstupoch

$$f(net_i) = \exp(net_i) / \sum_{j=1}^{n} \exp(net_j) \tag{4.1}$$

kde net_j je produkt násobenia váhového vektora a vektora aktivácii skrytých neurónov pre j-ty výstupný neurón. Jej použitie je podmienené tým že jednotlivé výstupy sú v intervale $(0, 1)$ a ich súčet je rovný 1, čo použité kódovanie symbolov spĺňa.

Úlohou siete bolo predikovať nasledujúci symbol, ktorý bol učiacim signálom pre MLP. Vektor aktivít neurónov vo výstupnej vrstve sa vyhodnotil určením neurónu s maximálnou hodnotou. Predikovaným symbolom bol ten kódovaný vektorom s "1" na rovnakom mieste. Zo schémy automatu pre Reberovu gramatiku však vidieť, že v každom stave sú dve možnosti pre ďalší symbol. Kvôli stochastickosti automatu nie je možné povedať ktorý z možných symbolov bude určite nasledovať. Preto sme pri testovaní brali do úvahy oba možné symboly a výstupom siete bola dvojica určená dvoma komponentami vektora s najvyššou aktivitou (podľa poradia). Za chybu sme považovali ak ani jeden z možných symbolov nebol predikovaný. Výnimkou bol koniec slova. V oboch stavoch predchádzajúcich akceptujúcemu je iba jedna možnosť ďalšieho postupu. V tomto prípade musel byť symbol "B" predikovaný ako prvý.

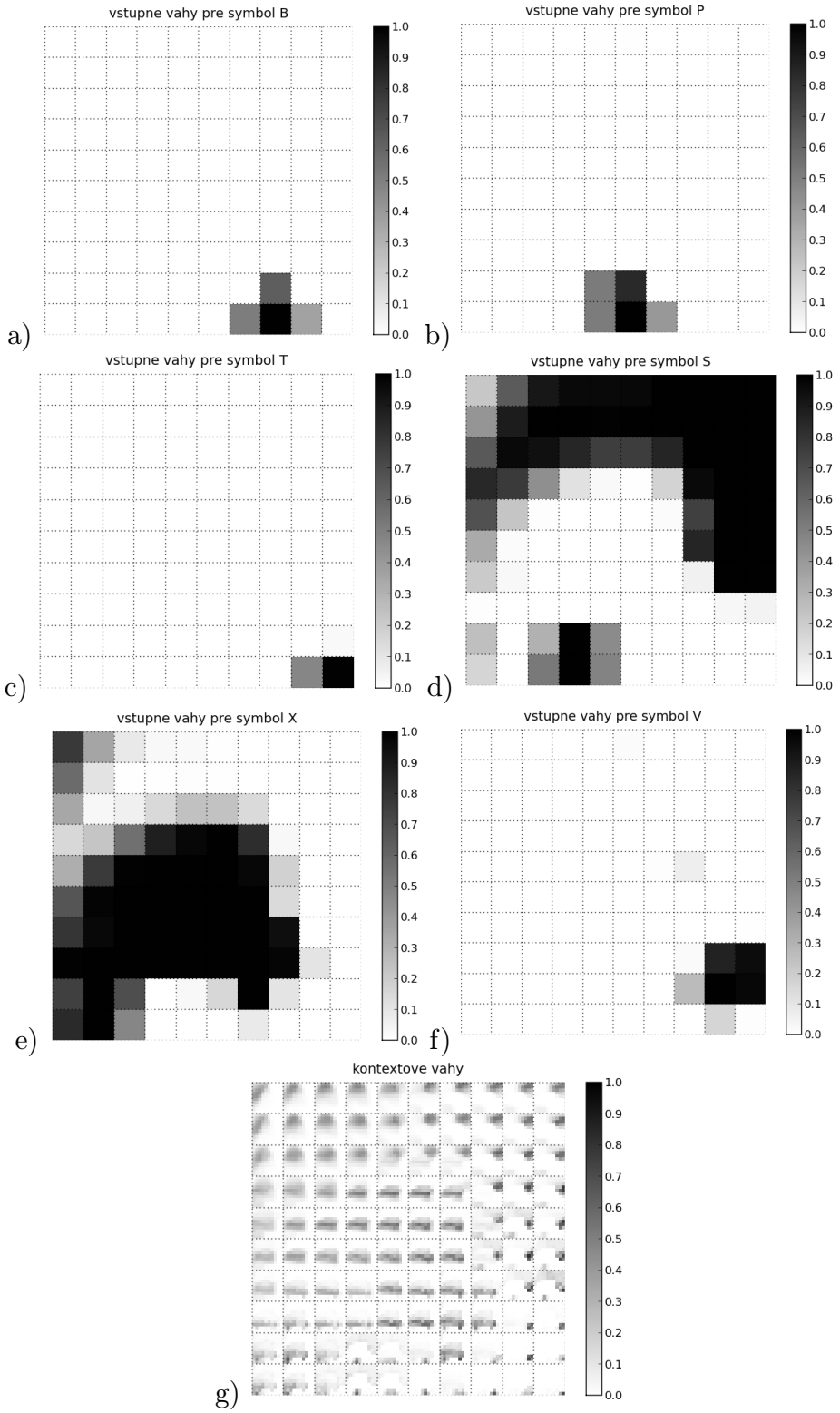
Celkovo možno povedať, že sieť úlohu zvláda dobre. Nastavenie vstupných váh na obrázku 4.2 ako aj zmeny aktivácii na obrázku 4.2 ukazujú, že



Obr. 4.3: Reprezentácie symbolu "X" v závislosti od kontextu

RecSOM si vytvorila jasne rozlíšiteľné reprezentácie pre jednotlivé symboly umožňujúce zachytiť aj pravidlá Reberovej gramatiky. Mierne prekvapujúci je veľký nepomer medzi počtom neurónov aktívnych pri jednotlivých písmenách. Pomerne veľká časť neurónov bola takmer vždy spojená s písmenom "S", hoci ide o priemerne frekventované slovo v trénovacej množine. Podobná pozornosť bola vždy venovaná ešte aspoň jednému písmenu, často "X", "T" alebo "V". Naopak pre zostávajúce písmená vyhradila sieť iba niekoľko neurónov. Možným vysvetlením je skutočnosť, že "S", "X", "T" sa vyskytovali viackrát za sebou a sieť potrebovala viac prostriedkov na ich rôznu reprezentáciu. "B" a "P" sa však nikdy neopakovali viackrát za sebou. Na znázornených mapách (obrázok 4.2) aktivácii možno pozorovať spomínané rozdiely v reprezentácii jednotlivých symbolov podľa momentálneho kontextu. MLP ako kategorizátor sa následne tieto črty naučil rozpoznávať.

Ako trénovacie dáta poslúžilo 1325 slov s celkovou dĺžkou 8337 znakov. Testovali sme na 1661 znakoch tvoriacich 116 slov nepoužitých pri trénovaní. Mape bola testovacia sada predložená v 100 epochách, počet epoch na natrénovanie MLP závisel od počtu skrytých neurónov. Ako vhodné parametre siete sa ukázali hodnoty $\alpha = 2,0, \beta = 1,0, \gamma = 0,1$ pre RecSOM, rýchlosť učenia $\alpha = 0,01$ pre MLP. Mapa pozostávala z 100 neurónov v mriežke 10



Obr. 4.4: Rozloženie a) -f) vstupných a g) kontextových váh RecSOM pri Reberovej gramatike

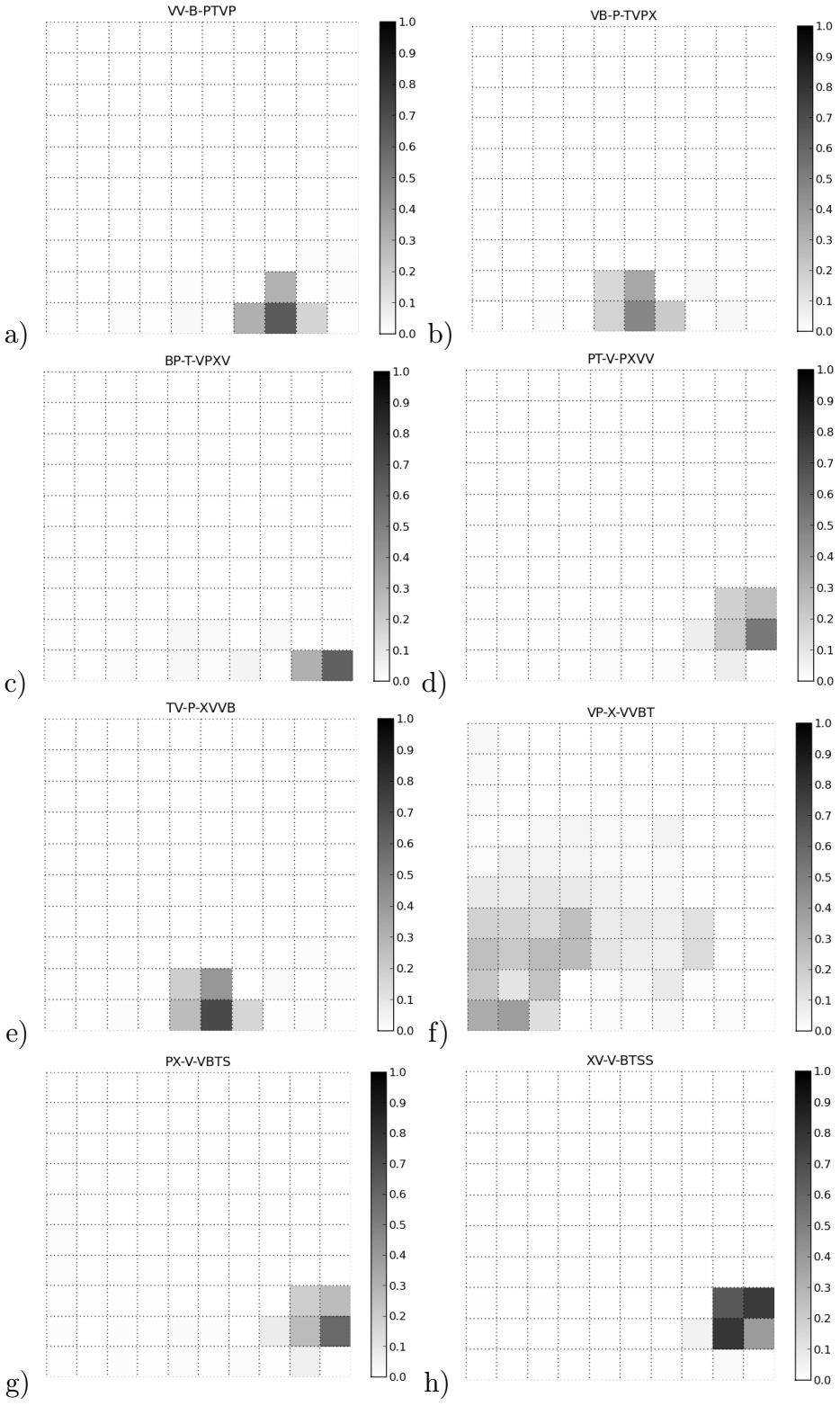
x 10. Skrytú vrstvu perceptrónu sme škálovali v rozmedzí 2 až 10 neurónov, výstupnú vrstvu tvorilo 6 neurónov.

Ako možno vidieť, každý neurón mapy prispieval svojou aktivitou k reprezentácii niektorého symbolu. Neobjavovali sa teda neaktívne neuróny bez špecializácie. Na druhej strane výsledky trénovania perceptrónu napovedajú, že tieto rozmery siete sú postačujúce pre danú úlohu. Použitie siete s väčším počtom (15 x 15) neurónov viedlo iba k rýchlejšiemu učeniu MLP. Väčší vplyv na trénovanie MLP mal rozsah skrytej vrstvy. V niektorých prípadoch stačili dva skryté neuróny po 30 epochách, v iných štyri. V rozmedzí 2 - 8 skrytých neurónov však rýchlosť učenia variovala a viac neurónov mohlo viesť aj k zhoršeniu učenia sa perceptrónu. Po 60 epochách bola vo veľkej väčšine prípadov sieť natrénovaná. Použitie 9 a viac neurónov už viedlo k bezchybnému natrénovaniu siete do 20 epoch. Pri mape 15 x 15 neurónov a 3 a skrytých neurónoch nastávalo natrénovanie do 10 epoch.

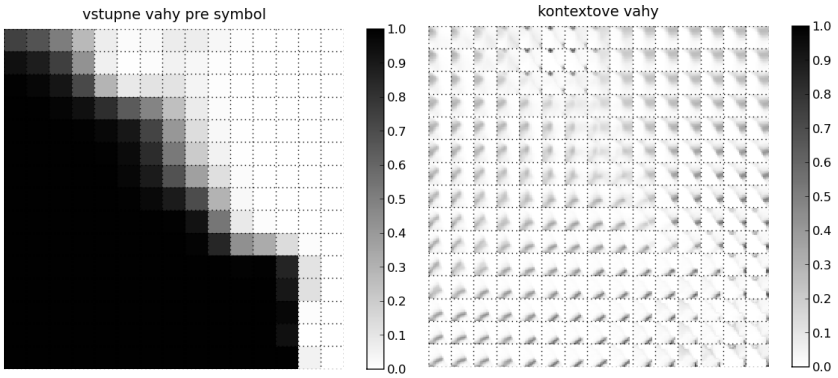
Možno teda skonštatovať, že model RecSOM je schopný extrahovať zo slov Reberovej gramatiky pravidlá ich tvorby. Skupiny aktívnych neurónov pri jednotlivých vstupoch by sme mohli považovať za reprezentácie istých stavov. Nejde však o priamu paralelu so stavmi automatu, keďže jeden stav automatu sieť reprezentuje inak pre rôzne vstupy a kontexty. Sú však natoľko diverzifikované, aby vystihli všetky pravidlá gramatiky. Úloha je v rámci možností modelu a jeho prostriedkov, pretože vyžaduje zachytenie kontextu aktuálneho vstupu do hĺbky 2.

4.3 Jazyk 0^n1^n

Základom pri tomto bezkontextovom jazyku bolo zistiť, či sieť dokáže simulovať zásobník, teda ako sa do nastavenia natrénovaných váh premietne štruktúra dát. Následne sme sa zamerali na dve schopnosti modelu RecSOM súvisiace s akceptáciou jazyka - generalizáciu a interpoláciu. Schopnosť zo-



Obr. 4.5: Zmeny aktivácii RecSOM postupne pri vstupoch a) B, b) P, c) T , d) V, e) P, f) X, g) V, h) V



Obr. 4.6: Rozloženie vstupných a kontextových váh RecSOM pri jazyku 0^n1^n

všeobecnenia (generalizáciu) sme testovali nasledovne: zvolili sme maximálnu dĺžku slova v trénovacej množine, napr. 10 (n = 5). Pri testovaní sme sieti predkladali aj dlhšie slová. Ich maximálna dĺžka bola dvojnásobná oproti trénovacej. Sieť tak musela reagovať aj na slová ktoré ešte nevidela. Vstupnými dátami pri skúmaní interpolačnej schopnosti siete boli slová s opačnou paritou. Sieť bola natrénovaná na slovách s párnym n a testovaná na slovách s nepárnym a opačne.

Keďže výstup z perceptrónu nie je binárny ale z intervalu (0,1), použili sme prahovú hodnotu 0,5. Klasická chyba učenia počítaná ako súčet rozdielov učiaceho signálu a výstupu siete v tomto prípade nemá veľkú výpovednú hodnotu vzhľadom na priebeh učenia. Problémom je predikcia pri aktuálnom vstupe "0". Sieť totiž nemá na základe čoho predpovedať či ďalším vstupom bude "0" alebo "1". Až s príchodom prvej "1" môže sieť správne voliť nasledujúci symbol. Preto sme si všímali hlavne odpovede siete po objavení sa "1". Na základe toho sme počítali chybu ako súčet zlých predikcii pri vstupe "1". Ďalšou pomôckou bol počet zle predikovaných slov - slov v ktorých bol aspoň jeden znak predpovedaný zle.

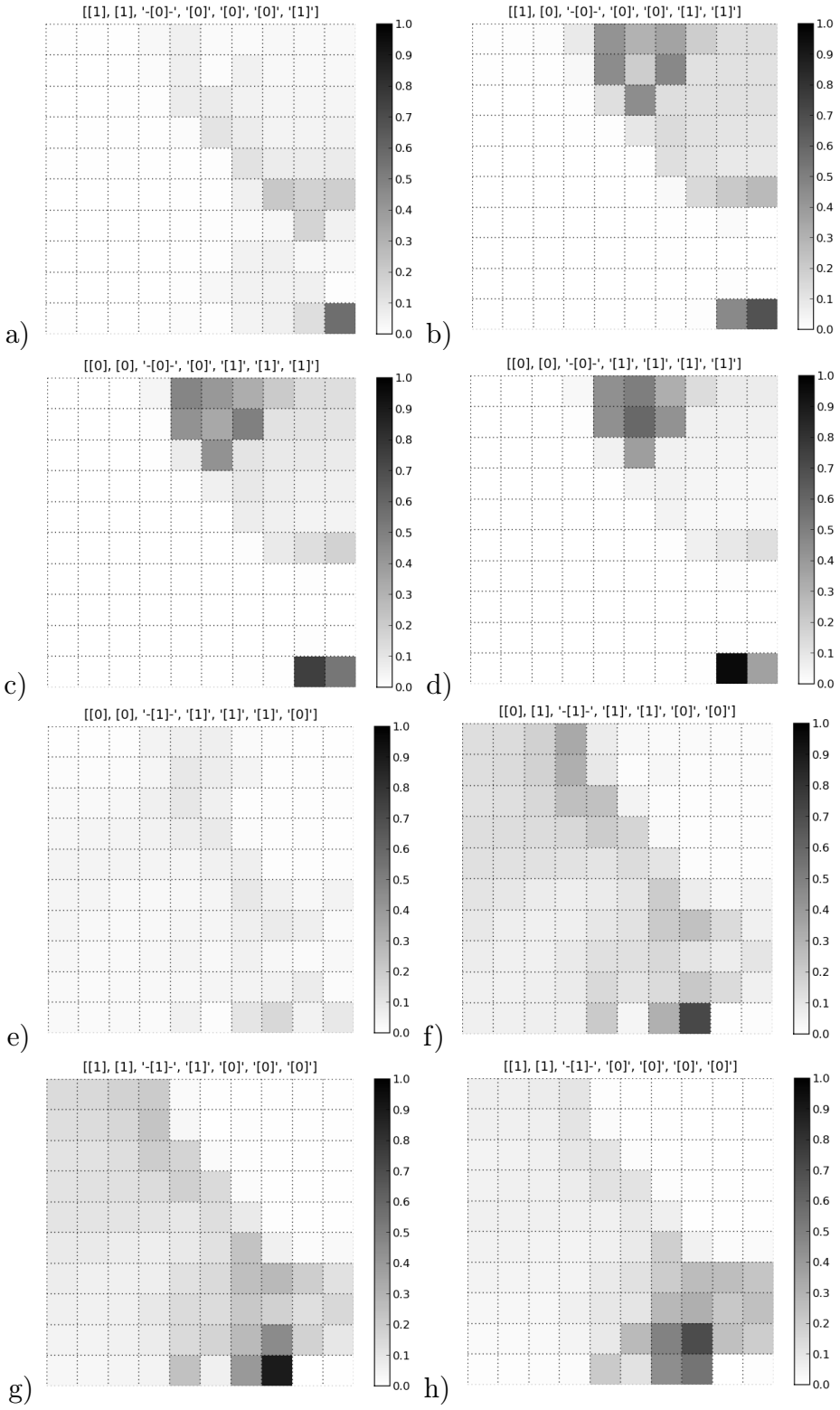
Na zobrazení máp aktivácii RecSOM 4.3 možno vidieť reprezentácie vytvorené pre symboly s rôznym kontextom. Reprezentácia pre "1" v slove "01"

sa líši od reprezentácií v dlhších slovách. Úlohou MLP bolo teda postrehnúť črty spoločné pre "1" na konci slov.

Sieť pri experimentoch ukázala veľkú závislosť od trénovacích dát. Trénovali sme postupne na slovách maximálnej dĺžky 10, 12, 14, 16. Ak jej boli pri trénovaní predkladané slová s určitým n, pri testovaní nebola schopná spracovať slová s väčším n. Ak bolo pri trénovaní $n = 5$, v slove dĺžky 12 pri šiestej "0" bola odpoveď siete nad prahovou hodnotou - sieť očakávala "1". Podobne pri piatej "1" sieť odpovedala pod-prahovou hodnotou - neočakávala ešte jednu "1". Sieť teda nepreukázala generalizačnú schopnosť.

Interpoláciu sme testovali na párnych slovách maximálnej dĺžky 8, 12, 16 a testovali na nepárnych dĺžky najviac 6, 10, 14. Paritu trénovacích a testovacích dát sme tiež vymenili. Ani pri tejto úlohe sieť nebola schopná reagovať správne na testovacie dáta. Jediný pozitívny prípad sme zaznamenali, ak sa sieť učila slová s n 1 a 3. Pri testovaní bola následne schopná rozoznať slová s n=2.

Kapacitné možnosti modelu sa ukázali obmedzené, dobre dokázala zvládnuť slová dĺžky najviac 10, ale iba v prípade že ich videla pri učení. Pri trénovaní na dlhších slovách robila sieť chyby. Väčšinou sa naučila krátke slová (n = 1,2,3) a najdlhšie slovo. Ostatné slová zvládala s premenlivou úspešnosťou.



Obr. 4.7: Zmeny aktivácii RecSOM postupne pri vstupoch a) 0, b) 0, c) 0 , d) 0, e) 1, f) 1, g) 1, h) 1

Kapitola 5

Záver

Sekvenčné a štruktúrované dáta predstavujú v poslednej dobe veľkú výzvu pre umelé neurónové siete, pretože táto forma je v reálnom svete oveľa častejšia. Preto je často objektom výskumu umelej inteligencii. Skúmanie správania neurónových sietí pri spracovaní takýchto dát umožňuje lepšie chápať aké mechanizmy prebiehajú v ľudskom mozgu. Model rekurzívnej samoorganizujúcej sa mapy je pomerne nový model (2002) neurónovej siete, navrhnutý na spracovanie sekvencii. Nadväzuje na predchádzajúce rekurzívne úpravy pôvodného algoritmu SOM. Doterajšie experimenty sa zameriavali na jednoduchšie sekvencie (stochastický automat, laserové dáta) ale aj anglický text.

V tejto práci sme sa zaoberali skúmaním schopností a kapacitných možností siete RecSOM. Zamerali sme sa na spracovanie sekvenčných dát vo forme Reberovej gramatiky, navrhnutej pre potreby psychologických výskumov implicitného učenia, a bezkontextového jazyka 0^n1^n. Jazyk 0^n1^n vyžaduje od siete, aby sa naučila "rátať" počet symbolov ktoré jej boli predložené a na základe toho reagovala na aktuálny vstup.

Podarilo sa nám ukázať, že RecSOM je schopná na základe slov z jazyka pre Reberovu gramatiku odvodiť pravidlá ich tvorby a správne reagovať aj na slová dlhšie a ešte nevidené. Naopak jazyk 0^n1^n predstavuje pre sieť problém.

Dokáže sa naučiť rozpoznávať slová dĺžky 10 ak na nich bola trénovaná. S rastúcou dĺžkou sa zväčšuje chyba už na trénovacích dátach. Sieť neprejavila schopnosť generalizácie. Nevedela správne reagovať na slová dlhšie ako v trénovacej sade, ani na slová s opačnou paritou ako pri trénovaní.

Zoznam obrázkov

Literatúra

[Cad94] V. Cadoret. Encoding syntactical trees with labelling recursive autoassociative memory. In *In Proceedings of the ECAI 94*, 1994.

[CT93] G. Chappel and J. Taylor. The temporal kohonen map. In *Neural Networks, 6*, pages 441–445. 1993.

[Hin90] G.E. Hinton. Mapping part-whole hierarchies into connectionist networks. *Artificial Intelligence*, 46(1-2):47–75, 1990.

[HJ04] B. Hammer and B. J. Jain. Neural methods for non-standard data. *ESANN'2004 proceedings*, pages 281–292, 2004.

[HMSS04] B. Hammer, A. Micheli, A. Sperduti, and M. Strickert. A general framework for unsupervised processing of structured data, 2004.

[Koh90] T. Kohonen. Statistical pattern recognition revisited. In *Advanced Neural Computers*, pages 137–144. Elsevier Science Publ. B.V, North-Holland, 1990.

[Neu05] N. Neubauer. Recursive soms and automata. Master's thesis, University of Osnabruck, 2005.

[Pla95] T. Plate. Holographic reduced representations. *IEEE Transactions on Neural Networks*, 6(3):623–641, 1995.

[Pol87] J. B. Pollack. *On Connectionist Models of Language*. PhD thesis, University of Illinois at Urbana-Champaign, 1987.

[Pol90] J.B. Pollack. Recursive distributed representations. *Artificial Intelligence*, 46(1-2):77–105, 1990.

[Reb67] A. S. Reber. Implicit learning of artificial grammars. *Journal of Verbal Learning and Verbal Behavior*, 1967.

[RHW86] D. Rumelhart, G. Hinton, and R. Williams. Learning internal representations by error propagation. *Parallel Distributed Processing*, 1986.

[RWE99] P. Rodriguez, J. Wiles, and J. L. Elman. A reccurent neural network that learns to count. *Connection Science*, 11(1):5–40, 1999.

[SH03] M. Strickert and B. Hammer. Unsupervised recursive sequence processing, 2003.

[SH04] M. Strickert and B. Hammer. Self-organizing context learning, 2004.

[SH05] M. Strickert and B. Hammer. Merge som for temporal data. *Neurocomputing*, 64:39 – 71, 2005.

[Sto01] Svorad Stolc. Učenie bezkontextového jazyka $l_{a^n b^n}$ pomocou rekurentnej bcm neurónovej siete. Master's thesis, Univerzita Komenského, 2001.

[Str] M. Strickert. Self-organizing neural networks for sequence processing.

[TFvM06] P. Tiňo, I. Farkaš, and J. van Mourik. Dynamics and topographic organization of recursive self-organizing maps. *Neural Comput.*, 18(10):2529–2567, 2006.

[TSB00] P. Tiňo, M. Stančík, and Ľ. Beňušková. Building predictive models on complex symbolic sequences via a first-order recurrent bcm network with lateral inhibition. In *IEEE-INNS-ENNS Int. Join Conference on Neural Networks*, 2000.

[VD02] T. Voegtlin and P. F. Dominey. Recursive self-organizing maps. *Communications of the ACM*, 2002.

[VHd97] M. Varsta, J. Heikkonen, and J. del R. Millan. Context learning with the self organizing map. In *Proceedings of WSOM'97, Workshop on Self-Organizing Maps, Espoo, Finland, June 4–6*, pages 197–202. Helsinki University of Technology, Neural Networks Research Centre, Espoo, Finland, 1997.

[WZ89] R. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, pages 270–280, 1989.