Trénovanie rekurentných neurónových sietí technikou rozšíreného Kalmanovho filtra

Peter Trebatický *
trebaticky@fiit.stuba.sk

Abstrakt Rekurentné neurónové siete sú na rozdiel od klasických dopredných neurónových sietí schopné spracovávať vstupy s časopriestorovou štruktúrou, ako sú napríklad časové postupnosti symbolov. Nakoľko klasické gradientové techniky na učenie rekurentných neurónových sietí pre dlhšie vstupné reťazce veľmi zle a pomaly konvergujú, hľadajú sa iné prístupy. V tomto príspevku stručne popisujem techniku trénovania rozšíreným Kalmanovým filtrom a jeho modifikáciami Unscented Kalman Filter, nprKF a ich zdieľanými verziami UKFj a nprKFj. Na experimente s predikciou nasledujúceho symbolu v postupnosti generovanej Reberovým automatom porovnávam výkonnosť týchto filtrov a tiež s „klasickým" skráteným spätným šírením chýb v čase (BPTT(h)).

Kľúčové slová: rekurentné neurónové siete, Kalmanov filter, rozšírený Kalmanov filter, EKF, Unscented Kalman Filter, UKF, nprKF, zdieľaný UKF a nprKF, UKFj, nprKFj

1 Úvod

Rekurentné neurónové siete sú na rozdiel od klasických dopredných neurónových sietí schopné spracovávať vstupy s časopriestorovou štruktúrou, ako sú napríklad časové postupnosti symbolov. Je ich možné použiť napríklad na spracovanie postupností slov jazykov generovaných gramatikami alebo postupností s chaotickým charakterom a pod., čo je veľmi užitočné napríklad pri riadení robotických systémov alebo analýze a predikcii výchyliek u strojných zariadení. Učenie rekurentných neurónových sietí je veľmi obtiažny numerický problém, ktorý, ak je riešený klasickými gradientovými optimalizačnými technikami pre dlhšie vstupné reťazce, tak sa veľmi zle a pomaly približuje k uspokojivým výsledkom. Preto sa hľadajú efektívnejšie spôsoby učenia rekurentných sietí.

V tomto príspevku čitateľa uvádzam do problematiky Kalmanovej filtrácie (odsek 2) a jeho rozšírenia pre nelineárne systémy, ktoré je už možné použiť pre trénovanie rekurentných neurónových sietí (odsek 3). V odsekoch 4 a 5 uvádzam vylepšenia tejto techniky, ktoré dosahujú lepšie výsledky, čo dokumentujem na experimente s trénovaním rekurentnej neurónovej siete na predikciu nasledujúceho symbolu (odsek 6). Tento príspevok predstavuje stručné zhrnutie mojej diplomovej práce.

2 Kalmanov filter

Kalmanov filter je považovaný za jeden z významných objavov dvadsiateho storočia - článok E. Kalmana vyšiel v roku 1960. Jeho prvými aplikáciami bolo riadenie zložitých dynamických systémov, ako rôzne výrobné procesy, lietadlá, lode, vesmírne lode (napr. bol použitý v navigačnej jednotke Apolla). Od začiatku bol a stále je hojne využívaný nielen

* Fakulta informatiky a informačných technológií STU Bratislava, Ilkovičova 3, 842 16 Bratislava

v automatizácii, ale aj v grafických a ekonomických aplikáciách, atď. no až relatívne nedávno sa začal objavovať aj v aplikáciách na trénovanie neurónových sietí, čo súvisí s postupom vývoja počítačových systémov.

Stavový vektor systému $\mathbf{x}_k$ (alebo len stav) je definovaný ako minimálna množina údajov, ktorá jednoznačne popisuje správanie dynamického systému, kde k je diskrétny čas. Jednoducho povedané, stav systému je vektor obsahujúci všetky relevantné premenné systému.

1. *Stavová rovnica*
$$\mathbf{x}_{k+1} = \mathbf{F}_{k+1,k}\mathbf{x}_k + \mathbf{q}_k \tag{1}$$
kde $\mathbf{F}_{k+1,k}$ je *prechodová matica* prevádzajúca stav $\mathbf{x}_k$ z času k do času $k+1$. O procesnom šume $\mathbf{q}_k$ sa predpokladá, že je aditívny, biely a gausovský s nulovou strednou hodnotou a kovariančnou maticou $\mathbf{Q}_k$.
2. *Rovnica pozorovaní*
$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{r}_k \tag{2}$$
kde $\mathbf{y}_k$ je pozorovanie v čase k a $\mathbf{H}_k$ je matica pozorovaní. O šume pozorovaní $\mathbf{r}_k$ sa takisto predpokladá, že je aditívny, biely a gausovský s nulovou strednou hodnotou a s kovariančnou maticou $\mathbf{R}_k$. Šumy $\mathbf{r}_k$ a $\mathbf{q}_k$ sú navzájom nezávislé.

Problém Kalmanovho filtrovania spočíva vo vyriešení stavovej rovnice a rovnice pozorovaní súčasne, optimálnym spôsobom, pre neznámy stav. Treba použiť všetky pozorované údaje pozostávajúce z vektorov $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_k$, na nájdenie odhadu minimálnej strednej kvadratickej chyby pre každé $k \geq 1$.

Kalmanov filter pracuje v dvoch neustále sa opakujúcich krokoch

1. *Časová aktualizácia*, tiež sa nazýva krok predikcie. Sem spadajú rovnice
$$\hat{\mathbf{x}}_k^- = \mathbf{F}_{k,k-1}\hat{\mathbf{x}}_{k-1} \tag{3}$$
$$\mathbf{P}_k^- = \mathbf{F}_{k,k-1}\mathbf{P}_{k-1}\mathbf{F}_{k,k-1}^T + \mathbf{Q}_{k-1} \tag{4}$$
Čiže vypočítame apriórne predikcie stavu a kovariancie chyby.
2. *Aktualizácia pozorovaním*, tiež sa nazýva krokom korekcie
$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^T[\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k]^{-1} \tag{5}$$
$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^-) \tag{6}$$
$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^- \tag{7}$$
T.j. korigujeme odhad získaný v predchádzajúcom kroku na základe pozorovania $\mathbf{y}_k$.

3 Rozšírený Kalmanov filter (EKF)

Ak je model *nelineárny*, čo je aj prípad neurónových sietí, musíme Kalmanov filter rozšíriť pomocou linearizačnej procedúry. Výsledný filter sa potom nazýva *rozšírený Kalmanov filter* (angl. *extended Kalman filter*) (EKF).

Neurónová sieť je nelineárny dynamický systém, ktorý môžeme popísať rovnicami [2, 5]
$$\mathbf{x}_{k-1} = \mathbf{x}_k + \mathbf{r}_k \tag{8}$$
$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{u}_k, \mathbf{v}_{k-1}) + \mathbf{q}_k \tag{9}$$

Stavová rovnica popisuje stav neurónovej siete ako stacionárny proces narušený procesným šumom $\mathbf{r}_k$, pričom stav siete $\mathbf{x}$ je daný vektorom jej váh. Rovnica pozorovania vyjadruje želaný výstup siete ako nelineárnu funkciu vstupného vektora $\mathbf{u}_k$, vektora váh $\mathbf{x}_k$ a pre rekurentné siete ešte aktivácii rekurentných neurónov z predchádzajúceho kroku $\mathbf{v}_{k-1}$. Táto rovnica je rozšírená o náhodný šum pozorovania $\mathbf{q}_k$. Kovariancia šumu $\mathbf{r}_k$ je $\mathbf{R}_k = E[\mathbf{r}_k \mathbf{r}_k^T]$ a kovariancia šumu $\mathbf{q}_k$ je $\mathbf{Q}_k = E[\mathbf{q}_k \mathbf{q}_k^T]$.

Základná myšlienka rozšíreného Kalmanovho filtra spočíva v linearizovaní rovnice pozorovania v každom časovom kroku okolo najnovšieho odhadu stavu $\hat{\mathbf{x}}_k$. Pre tento účel používame prvý stupeň Taylorovej aproximácie nelineárnej funkcie [2].

Trénovanie neurónovej siete môžeme vyjadriť pomocou teórie Kalmanovho filtra ako problém hľadania odhadu stavu $\mathbf{x}$, ktorý minimalizuje strednú kvadratickú chybu, použijúc všetky doterajšie pozorovania. Riešenie tohto problému môžeme vyjadriť nasledovne:

$$\mathbf{K}_k = \mathbf{P}_k \mathbf{H}_k^T [\mathbf{H}_k \mathbf{P}_k \mathbf{H}_k^T + \mathbf{R}_k]^{-1} \tag{10}$$

$$\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k + \mathbf{K}_k [\mathbf{y}_k - h(\hat{\mathbf{x}}_k, \mathbf{u}_k), \mathbf{v}_{k-1}] \tag{11}$$

$$\mathbf{P}_{k+1} = \mathbf{P}_k - \mathbf{K}_k \mathbf{H}_k \mathbf{P}_k + \mathbf{Q}_k \tag{12}$$

kde $\hat{\mathbf{x}}$ predstavuje vektor všetkých váh, $h(\cdot)$ funkciu vracajúcu vektor skutočných výstupov, $\mathbf{y}$ vektor požadovaných výstupov, $\mathbf{K}$ je tzv. matica Kalmanovho zosilnenia, $\mathbf{P}$ kovariančná matica chyby stavu a $\mathbf{H}$ je matica pozorovania (Jacobián). Každý prvok matice $\mathbf{H}$ vypočítame ako parciálnu deriváciu i-teho výstupu k j-tej váhe. Jej výpočet sa realizuje buď algoritmom spätného šírenia chýb (angl. *Back Propagation*) v prípade doprednej neurónovej siete, alebo v prípade rekurentnej siete algoritmami *Back Propagation Through Time*, *Truncated Back Propagation Through Time*, prípadne *Real Time Recurrent Learning*.

4 Filtre UKF a UKFj

V EKF technike je kľúčovým elementom vývoj kovariančnej matice $\mathbf{P}$, ktorá je v každom kroku aproximovaná použitím prvého stupňa Taylorovho rozvoja nelineárnej funkcie, ktorá dáva do súvisu výstupy siete s váhami. Linearizácia pomocou Taylorovho rozvoja však vo veľa prípadoch poskytuje nedostatočne presnú reprezentáciu [4].

S úspechom bolo použité aj nelineárne zovšeobecnenie Kalmanovho filtra [7] v rôznych aplikáciách vrátane trénovania rekurentných neurónových sietí pod názvom Unscented Kalman Filter (UKF) [3]. Koncepčne je základný princíp UKF podobný s EKF. Implementácia je však značne rozdielna. Oproti aproximácii pomocou Taylorovho rozvoja nie sú potrebné žiadne derivácie, ale len vyhodnotenia funkcií, čo umožňuje jednoduchšiu implementáciu filtra. Podstatné však je, že táto technika dosahuje lepšie výsledky ako EKF.

Hlavný rozdiel medzi EKF a UKF spočíva v spôsobe, ako sú gausovské náhodné premenné (GNP) reprezentované na šírenie dynamikou systému. V EKF je stav aproximovaný pomocou GNP, ktoré sú šírené analyticky cez linearizáciu prvého rádu nelineárneho systému, čo môže zanášať veľké chyby v skutočnej strednej hodnote a kovariancii transformovaných GNP.

UKF využíva tzv. transformáciu bez zápachu (angl. *unscented transformation*) [3], čo je relatívne nová metóda na výpočet štatistík náhodnej premennej, ktorá prešla nelineárnou transformáciou. Vyberieme množinu tzv. *sigma bodov* tak, aby tieto body úplne zachytávali skutočnú strednú hodnotu a kovarianciu GNP. Po prešírení *skutočným* nelineárnym

systémom presne zachytávajú posteriórnu strednú hodnotu a kovarianciu minimálne do druhého stupňa Taylorovho rozvoja pre *ľubovoľnú* nelinearitu. Naproti tomu EKF dosahuje presnosť len do prvého stupňa.

Táto metóda pripomína Monte Carlo metódy, kedy sa náhodne vyberie veľa vzoriek, ktoré následne tiež prejdú nelineárnou transformáciou. Tu je ale značný rozdiel v tom, že sigma body nie sú vyberané náhodne, ale deterministicky a navyše je počet sigma bodov nízky – $2n_x + 1$, kde n_x je dimenzia náhodnej premennej (veľkosť vektora) [3].

Nižšie uvádzam rovnice pre zdieľaný UKF (angl. *Joint UKF – UKFj*), pri ktorom zahrnieme aktivácie rekurentných neurónov do stavového vektora, čím použijeme UKF na odhad váh aj stavu zároveň [7]. Treba upozorniť, že tento názov sa používa takisto pre filter, v ktorom sa do jedného stavového vektora spoja stav systému so vstupmi [2]. Toto sa používa v situácii, kedy nemáme k dispozícii nezašumený vstup do systému. V kontexte trénovania *rekurentných* neurónových sietí som tento filter v literatúre nenašiel. Vytvoril som ho na základe podobnosti s filtrom nprKFj popísaním v [7].

Rozdiel UKFj oproti UKF je jedine v časovej aktualizácii, nakoľko sa funkcia f pridaním ďalších stavov stala nelineárna (pri UKF: $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{q}_k$).

$$\begin{aligned}\mathbf{x}_{k+1} &= f(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{q}_k \\ \mathbf{y}_k &= g(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{r}_k\end{aligned}$$

Pri trénovaní rekurentných sietí technikou UKF je potrebné pre každý sigma bod (t.j. rôzne váhy) prešíriť sieť niekoľko krokov z minulosti. UKFj teda na jednej strane zväčšuje výpočtovú náročnosť tým, že rozširuje veľkosť stavového vektora $\mathbf{x}$, na strane druhej zjednodušuje výpočet výstupu siete pre rôzne hodnoty váh (sigma bodov).

Pre prehľadnosť uvádzam sumár všetkých rovníc aj s odporúčanými hodnotami parametrov [2, 3]. Po inicializácii sa rovnako ako pri EKF striedajú kroky predikcie a korekcie.

– Inicializácia
  1. Počiatočný odhad stavu a kovariancií a parametre pre UKF
     $\hat{\mathbf{x}}_{[n_x\times 1]}$, $\hat{\mathbf{P}}_{[n_x\times n_x]} = \mathbf{I}\,P_0$, $\mathbf{Q}_{[n_x\times n_x]} = \mathbf{I}\,Q_0$, $\mathbf{R}_{[n_o\times n_o]} = \mathbf{I}\,R_0$, $\kappa = 3 - n_x$, $\beta = 2$, $1 \leq \alpha \leq 10^{-4}$, $\lambda = \alpha^2(n_x + \kappa) - n_x$, $\gamma = \sqrt{n_x + \lambda}$
  2. Váhy pre UKF
     $W_0^{(m)} = \frac{\lambda}{n_x+\lambda}$
     $W_0^{(c)} = \frac{\lambda}{n_x+\lambda} + 1 - \alpha^2 + \beta$
     $W_i^{(m)} = W_i^{(c)} = \frac{1}{2(n_x+\lambda)},\ i = 1, \ldots, 2n_x$
– Časová aktualizácia – predikcia
  - Pomocné výpočty
    1. $\mathcal{X}_{[n_x\times 2n_x+1]} = \left[\hat{\mathbf{x}}\ \hat{\mathbf{x}} + \gamma\sqrt{\hat{\mathbf{P}}}\ \hat{\mathbf{x}} - \gamma\sqrt{\hat{\mathbf{P}}}\right]$ (sigma body)
    2. $\mathcal{X}^*_{[n_x\times 2n_x+1]} = f(\mathcal{X}, \mathbf{u})$
  - Aktualizácia odhadu stavu a kovariancie
    1. $\bar{\mathbf{x}}_{[n_x\times 1]} = \sum_{i=0}^{2n_x} W_i^{(m)} \mathcal{X}_i^*$
    2. $\bar{\mathbf{P}}_{[n_x\times n_x]} = \sum_{i=0}^{2n_x} W_i^{(c)} (\mathcal{X}_i^* - \bar{\mathbf{x}})(\mathcal{X}_i^* - \bar{\mathbf{x}})^T + \mathbf{Q}$
– Aktualizácia pozorovaním – korekcia
  - Pomocné výpočty
    1. $\bar{\mathcal{X}}_{[n_x\times 2n_x+1]} = \left[\bar{\mathbf{x}}\ \bar{\mathbf{x}} + \gamma\sqrt{\bar{\mathbf{P}}}\ \bar{\mathbf{x}} - \gamma\sqrt{\bar{\mathbf{P}}}\right]$ (sigma body)
    2. $\mathcal{Y}_{[n_o\times 2n_x+1]} = g(\bar{\mathcal{X}}, \mathbf{u})$

3. $\bar{\mathbf{y}}_{[n_o \times 1]} = \sum_{i=0}^{2n_x} W_i^{(m)} \mathcal{Y}_i$
4. $\mathbf{P}_{yy[n_o \times n_o]} = \sum_{i=0}^{2n_x} W_i^{(c)} (\mathcal{Y}_i - \bar{\mathbf{y}})(\mathcal{Y}_i - \bar{\mathbf{y}})^T + \mathbf{R}$
5. $\mathbf{P}_{xy[n_x \times n_o]} = \sum_{i=0}^{2n_x} W_i^{(c)} (\bar{\mathcal{X}}_i - \bar{\mathbf{x}})(\mathcal{Y}_i - \bar{\mathbf{y}})^T$
6. $\mathbf{K}_{[n_x \times n_o]} = \mathbf{P}_{xy} \mathbf{P}_{yy}^T$

- Aktualizácia odhadu stavu a kovariancie
  1. $\hat{\mathbf{x}} = \bar{\mathbf{x}} + \mathbf{K}(\mathbf{y}_d - \bar{\mathbf{y}})$
  2. $\hat{\mathbf{P}} = \bar{\mathbf{P}} - \mathbf{K}\mathbf{P}_{yy}\mathbf{K}^T$

Výraz $\begin{bmatrix}\mathbf{A}\ \mathbf{B}\end{bmatrix}$ znamená spojenie matíc $\mathbf{A}$ a $\mathbf{B}$ do jednej, výraz $B = \sqrt{A}$ znamená, že platí rovnosť $A = B\,B^T$. Nie je to teda odmocnica v pravom zmysle ($A = B\,B$), ale tzv. Choleského faktor. Pri UKF nezáleží na voľbe tohto faktora (existuje ich viac), preto je možné použiť efektívne a numericky stabilné metódy, ako napr. Choleského dekompozíciu [6].

Stav predstavuje v prípade neurónových sietí vektor všetkých trénovateľných váh. Odhad stavového vektora a kovariancie pred pozorovaním je označený $\bar{\mathbf{x}}$ a $\bar{\mathbf{P}}$, ich odhad po pozorovaní ako $\hat{\mathbf{x}}$ a $\hat{\mathbf{P}}$. Pozorovaním sa v prípade neurónových sietí rozumie požadovaný výstup v danom čase ($\mathbf{y}_d$). Konštanta α určuje rozptýlenie sigma bodov okolo $\bar{\mathbf{x}}$. κ je druhotný škálovací koeficient (nakoľko ovplyvňuje primárny koeficient λ). Uvedená hodnota je pre gausovské rozdelenie vhodnou heuristikou [3]. Parameter β zahŕňa znalosť rozdelenia – pre gausovské je optimálne $\beta = 2$ [3].

Návratovou hodnotou funkcie $f(\mathbf{x}, \mathbf{u})$ je nový stavový vektor po tom, ako sme nastavili váhy a výstupy rekurentných neurónov podľa $\mathbf{x}$ a sieť prešírime (jeden časový krok) pri vstupnom vektore $\mathbf{u}$. Z toho vyplýva, že pre zložky stavového vektora $\mathbf{x}$, ktoré zodpovedajú váham, platí $f(\mathbf{x}) = \mathbf{x}$, a že funkcia f mení len výstup rekurentných neurónov.

Funkciu $g(\mathbf{x})$ treba interpretovať ako vektor výstupov siete po tom, čo sa nastavili váhy a výstupy rekurentných neurónov a neurónom, ktorých výstupy nie sú súčasťou stavového vektora, sa prepočíta ich výstup. Takáto interpretácia je konzistentná s obvyklou rovnicou pozorovania v dynamických systémoch, kedy je výstup systému funkciou jeho stavových premenných[7].

5 Filtre nprKF a nprKFj

Pri UKF sa na výpočet štatistík náhodnej premennej, ktorá prešla nelineárnou transformáciou, používajú sigma body a transformácia bez zápachu. Pri EKF sa používa prvý stupeň Taylorovho rozvoja. Ďalším prístupom, resp. ďalšou modifikáciou Kalmanovho filtra je filter s názvom nprKF [7], pomenovaný po jeho autoroch.

Rovnako ako pri UKF, ani pri tomto filtri nie je potrebné počítať derivácie, ale z iného dôvodu. Pri nprKF je Taylorov rozvoj nahradený Stirlingovou formulou pre aproximáciu funkcie na intervale, pričom sa funkcia f aproximuje prvými dvoma stupňami [4]. Túto formulu môžeme zjednodušene chápať ako Taylorov rozvoj, kde sme derivácie nahradili *pomernými diferenciami* (angl. *divided difference*). Autori kvôli numerickej stabilite narábajú s Choleského faktormi kovariančných matíc.

Podobne ako v predchádzajúcom odseku uvádzam vzorce pre *zdieľaný nprKF* (nprKFj), t.j. spojíme váhy s výstupmi rekurentných neurónov do stavového vektora, čím sa vyhneme ďalšiemu parametru – od koľkých krokov prešíriť sieť pre každý vektor váh.

– Inicializácia
$\hat{\mathbf{x}}_{[n_x \times 1]}$, $\hat{\mathbf{S}}_{x[n_x \times n_x]} = \mathbf{I}\, P_0^{1/2}$, $\mathbf{S}_{v[n_x \times n_x]} = \mathbf{I}\, Q_0^{1/2}$, $\mathbf{S}_{w[n_o \times n_o]} = \mathbf{I}\, R_0^{1/2}$, $h^2 = 3$

– Časová aktualizácia
  - Pomocné výpočty
    1. $\mathbf{S}^{(1)}_{x\hat{x}\,[n_x\times n_x]} : (\mathbf{S}^{(1)}_{x\hat{x}})_{i,j} = \frac{1}{2h}\left[f_i(\hat{\mathbf{x}} + h\hat{\mathbf{s}}_{x,j}, \mathbf{u}) - f_i(\hat{\mathbf{x}} - h\hat{\mathbf{s}}_{x,j}, \mathbf{u})\right]$
    2. $\mathbf{S}^{(2)}_{x\hat{x}\,[n_x\times n_x]} : (\mathbf{S}^{(2)}_{x\hat{x}})_{i,j} = \frac{\sqrt{h^2-1}}{2h^2}\left[f_i(\hat{\mathbf{x}} + h\hat{\mathbf{s}}_{x,j}, \mathbf{u}) + f_i(\hat{\mathbf{x}} - h\hat{\mathbf{s}}_{x,j}, \mathbf{u}) - 2f_i(\hat{\mathbf{x}}, \mathbf{u})\right]$
  - Aktualizácia odhadu stavu a kovariancie
    1. $\bar{\mathbf{x}}_{[n_x\times 1]} = \frac{h^2-n_x}{h^2} f(\hat{\mathbf{x}}, \mathbf{u}) + \frac{1}{2h^2}\sum_{p=1}^{n_x}\left[f(\hat{\mathbf{x}} + h\hat{\mathbf{s}}_{x,p}, \mathbf{u}) + f(\hat{\mathbf{x}} - h\hat{\mathbf{s}}_{x,p}, \mathbf{u})\right]$
    2. $\bar{\mathbf{S}}_{x\,[n_x\times n_x]} = \left[\mathbf{S}^{(1)}_{x\hat{x}}\ \mathbf{S}_v\ \mathbf{S}^{(2)}_{x\hat{x}}\right]_{[n_x\times 3n_x]}$ a zredukujeme ju na trojuholníkovú maticu
– Aktualizácia pozorovaním
  - Pomocné výpočty
    1. $\bar{\mathbf{y}}_{[n_o\times 1]} = \frac{h^2-n_x}{h^2} g(\bar{\mathbf{x}}, \mathbf{u}) + \frac{1}{2h^2}\sum_{p=1}^{n_x}\left[g(\bar{\mathbf{x}} + h\bar{\mathbf{s}}_{x,p}, \mathbf{u}) + g(\bar{\mathbf{x}} - h\bar{\mathbf{s}}_{x,p}, \mathbf{u})\right]$
    2. $\mathbf{S}^{(1)}_{y\bar{x}\,[n_o\times n_x]} : (\mathbf{S}^{(1)}_{y\bar{x}})_{i,j} = \frac{1}{2h}\left[g_i(\bar{\mathbf{x}} + h\bar{\mathbf{s}}_{x,j}, \mathbf{u}) - g_i(\bar{\mathbf{x}} - h\bar{\mathbf{s}}_{x,j}, \mathbf{u})\right]$
    3. $\mathbf{S}^{(2)}_{y\bar{x}\,[n_o\times n_x]} : (\mathbf{S}^{(2)}_{y\bar{x}})_{i,j} = \frac{\sqrt{h^2-1}}{2h^2}\left[g_i(\bar{\mathbf{x}} + h\bar{\mathbf{s}}_{x,j}, \mathbf{u}) + g_i(\bar{\mathbf{x}} - h\bar{\mathbf{s}}_{x,j}, \mathbf{u}) - 2g_i(\bar{\mathbf{x}}, \mathbf{u})\right]$
    4. $\mathbf{S}_{y[n_o\times n_o]} = \left[\mathbf{S}^{(1)}_{y\bar{x}}\ \mathbf{S}_w\ \mathbf{S}^{(2)}_{y\bar{x}}\right]_{[n_o\times n_x+n_o+n_x]}$ a zredukujeme na trojuholníkovú maticu
    5. $\mathbf{K}_{[n_x\times n_o]} = \bar{\mathbf{S}}_x(\mathbf{S}^{(1)}_{y\bar{x}})^T\left(\mathbf{S}_y\mathbf{S}_y^T\right)^{-1}$
  - Aktualizácia odhadu stavu a kovariancie
    1. $\hat{\mathbf{x}} = \bar{\mathbf{x}} + \mathbf{K}(\mathbf{y}_d - \bar{\mathbf{y}})$
    2. $\hat{\mathbf{S}}_x = \left[\bar{\mathbf{S}}_x - \mathbf{K}\mathbf{S}^{(1)}_{y\bar{x}}\ \mathbf{K}\mathbf{S}_w\ \mathbf{K}\mathbf{S}^{(2)}_{y\bar{x}}\right]_{[n_x\times n_x+n_o+n_x]}$ a zredukujeme na trojuholníkovú

Výraz $\bar{\mathbf{s}}_{x,j}$ znamená j-ty stĺpec matice $\bar{\mathbf{S}}_x$. Význam funkcií f a g je rovnaký ako pri UKFj. Redukovanie obdĺžnikovej matice A na trojuholníkovú znamená nájsť hornú alebo dolnú trojuholníkovú maticu S tak, aby platila rovnosť $AA^T = SS^T$. Toto je možné dosiahnuť Householderovými transformáciami (viď. napr. [6]). Je to kvôli tomu, aby sa zachoval vzťah, kedy napr. $\bar{\mathbf{P}} = \bar{\mathbf{S}}_x\bar{\mathbf{S}}_x^T$, kde $\bar{\mathbf{S}}_x$ je Choleského faktor kovariančnej matice. Tento ale musí byť štvorcový, kvôli výpočtu napr. matice $\mathbf{S}^{(1)}_{y\bar{x}}$. Spomínaná triangularizácia je jedna z efektívnych metód, ako toto dosiahnuť.

Pri pohľade na uvedené vzorce a na vzorce filtra UKFj je zrejmá podobnosť oboch filtrov. Autori nprKF explicitne nehovoria o sigma bodoch, v skutočnosti ich vlastne tiež využívajú a dokonca aj rovnaký počet $(2n_x + 1)$. Potvrdzuje to aj analýza v [4], nakoľko odhad strednej hodnoty stavu je u oboch filtrov identický. Odlišujú sa však v odhade kovariancie. nprKF o niečo lepšie zachytáva jej skutočnú hodnotu – rozdiely v aktualizácii kovariancie sa však v porovnaní s UKF pohybujú okolo štvrtého a vyššieho stupňa Taylorovho rozvoja. Odhad kovariancie pomocou UKF je teda predsa len menej presný a navyše môže niekedy viesť na odhad, ktorý nie je kladne semidefinitný [4].

6 Experiment

Hlavným cieľom mojich experimentov bolo porovnanie výkonnosti všetkých spomínaných fitrov – EKF, UKF, nprKF, UKFj a nprKFj. Vykonal som viacero experimentov aj z oblasti automatizácie, čím sa potvrdila ich široká použiteľnosť. V tomto odseku uvádzam jeden z experimentov, ktorý je zameraný na trénovanie rekurentných neurónových sietí na predikciu nasledujúceho symbolu.

Kvôli linearizácii nelineárneho systému (neurónovej siete) pri nasadení filtra EKF je potrebné vypočítať maticu derivácií výstupov siete od každej váhy. Na to som použil skrátené spätné šírenie chýb v čase (angl. *truncated backpropagation through time*) (BPTT(h)), podľa odporúčania v [7].

Predikcia nasledujúceho symbolu prebieha tak, že na vstup siete predložíme prvý symbol v poradí, pričom ako požadovaný výstup siete je symbol nasledujúci. Zvolil som Elmanovu architektúru rekurentnej neurónovej siete – t.j. sieť s jednou skrytou vrstvou, ktorá je rekurentná.

Na meranie úspešnosti predikcie nasledujúceho symbolu som použil ukazovateľ NNL (angl. *normalised negative log-likelihood*), ktorá je počítaná cez symbolickú postupnosť v časoch $t = 1$ až T [1]

$$NNL = -\frac{1}{T}\sum_{t=1}^{T} \log_{|A|} p^{(t)}(s^{(t)}) \tag{13}$$

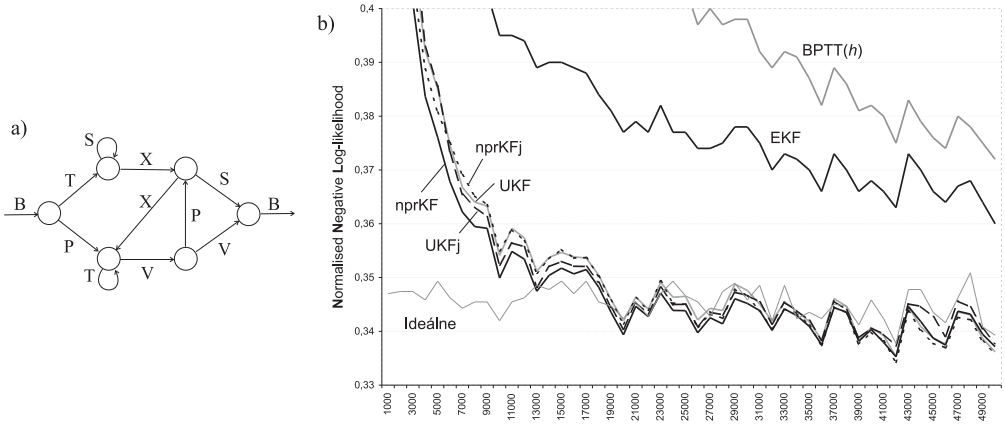
kde základ logaritmu $|A|$ predstavuje počet symbolov vo vstupnej abecede A a výraz $p^{(t)}(s^{(t)})$ predstavuje pravdepodobnosť predikovania symbolu $s^{(t)}$ v čase t. Ak sa $NNL = 0$, potom sieť predikuje nasledujúci symbol na 100%, zatiaľ čo $NNL \geq 1$ zodpovedá veľmi nepresnej predikcii.

Trénovacia postupnosť bola vygenerovaná použitím Reberovho automatu (obr. 1a), pričom s pravdepodobnosťou 50% sa vybral jeden z dvoch možných nasledujúcich symbolov (okrem prípadu, kedy nasleduje symbol B). Reberova gramatika obsahuje 6 symbolov, ktoré zakódujeme pomocou one-hot kódovania a dávame sieti na vstup. Toto kódovanie znamená, že počet vstupov siete je rovný počtu symbolov gramatiky. Pre každý symbol je teda určený jeden vstup, ktorý nastavíme na hodnotu 1 a ostatné na hodnotu 0. Podobne aj výstupy som zakódoval pomocou one-hot kódovania. Týmto dokážeme jednoducho vypočítať ukazovateľ NNL – stačí znormalizovať výstupný vektor (t.j. súčet jeho zložiek bude rovný 1) a pravdepodobnosť predikovania požadovaného symbolu je priamo hodnota zložky vektora prislúchajúcej danému symbolu.

Predikcia symbolov generovaných Reberovým automatom je pomerne jednoduchý problém, pretože z predchádzajúcich dvoch symbolov sme schopní určiť, v ktorom stave sa nachádzame. Neurónová sieť by mala po natrénovaní byť schopná predikovať, s akou pravdepodobnosťou nasleduje ten-ktorý symbol. Z tohto taktiež vyplýva, že NNL nemôžeme očakávať blízke nule, skôr naopak, nakoľko nevieme s istotou, aký bude nasledujúci symbol v prípade, ak sú dve možnosti.

Každou technikou som trénoval rekurentnú neurónovú sieť s troma skrytými neurónmi, čo pre tento problém postačuje. Trénovanie prebiehalo postupným predkladaním symbolov, pričom aktualizácia váh bola vykonaná v každom kroku. Ukazovateľ NNL som počítal pre každých 1000 symbolov. Porovnanie všetkých filtrov na základe tohto ukazovateľa uvádzam na obrázku 1b. Pre zaujímavosť som do grafu pridal aj krivku znázorňujúcu „ideálnu" hodnotu NNL. Tú dostaneme, ak predpovedáme nasledujúci symbol presne s pravdepodobnosťou 0.5, okrem prípadov, kedy nasleduje symbol B s pravdepodobnosťou 1.

Z obrázku pekne vidno výkonnosť jednotlivých techník (v zmysle rýchlosti konvergencie a celkového výsledku). Najslabšou sa ukázala byť technika BPTT(h). Rozšírený Kalmanov filter konverguje rýchlejšie a oproti BPTT(h) dosiahne lepší výsledok o približne 0.01. No a nakoniec je zrejmá dominancia filtrov UKF, UKFj, nprKF a nprKFj. Konvergencia všetkých štyroch je v porovnaní s BPTT(h) rapídna, v porovnaní s EKF veľmi rýchla. Najzaujímavejší je však najmä nimi dosiahnutý výsledok. Rozdiely medzi nimi sú minimálne – nedá sa určiť „víťaza", ale dosiahli hodnotu NNL *lepšiu* ako je „ideálna". Možnú príčinu tohto javu vidím v tom, že tieto filtre pomohli sieti „naučiť sa" okrem Reberovej gramatiky, čiastočne aj systém použitý na generovanie pseudonáhodného čísla aplikovaný pri vytváraní slova z Reberovho jazyka.



Obrázok 1: a) Reberov automat b) Závislosť NNL od počtu vstupných symbolov pre jednotlivé techniky trénovania rekurentnej siete.

7 Zhodnotenie

V literatúre sa uvádzajú filtre EKF, UKF a nprKF väčšinou pre všeobecný dynamický systém. Na jednej strane to svedčí o širokej použiteľnosti jednotlivých filtrov, na strane druhej sú rovnice vyjadrujúce činnosť týchto filtrov zložité a je ich viacero. Preto prínosom tejto práce je uvedenie rovníc jednotlivých filtrov upravených čisto pre účely trénovania rekurentných sietí. Jej význam ocenia najmä tí, ktorým nepostačuje výkonnosť klasických techník trénovania neurónových sietí, a hľadajú preto lepšie alternatívy.

Ďalším prínosom mojej práce je priame porovnanie všetkých filtrov v jednom experimente práve pri ich použití na trénovanie rekurentných neurónových sietí. Nakoľko som v mnou preštudovanej literatúre nenašiel zmienku o filtri UKFj v kontexte trénovania rekurentných neurónových sietí tak, ako je to pri nprKFj, dovolím si opatrne tvrdiť, že je to pôvodná myšlienka.

Literatúra

1. ČERŇANSKÝ, Michal, MAKULA, Matej, BEŇUŠKOVÁ, Ľubica: *Processing Symbolic Sequences by Recurrent Neural Networks Trained by Kalman Filter-Based Algorithms*, Bratislava: Slovak University of Technology and Comenius University, 2003.
2. HAYKIN, Simon editor: *Kalman Filtering and Neural Networks*, New York: John Wiley & Sons, Inc., 2002. ISBN: 0-471-36998-5
3. JULIER, Simon J., UHLMANN, Jeffrey K.: *A New Extension of the Kalman Filter to Nonlinear Systems*, Proceedings of AeroSence: The 11th International Symposium on Aerospace/Defence Sensing, Simulation and Controls, Mar. 2000.
4. NØRGAARD, Magnus, POULSEN, Niels K., RAVN, Ole: *Advances in Derivative-Free State Estimation for Nonlinear Systems*, Revidovaná edícia. Technical University of Denmark, 2000. Technická správa IMM-REP-1998-15.
5. PATEL, Gaurav S.: *Modeling Nonlinear Dynamics with Extended Kalman Filter Trained Recurrent Multilayer Perceptrons*, McMaster University, 2000. Diplomový projekt.
6. PRESS, William H. et al.: *Numerical Recipes in C - The Art of Scientific Computing*, 2. vyd. Cambridge University Press, 1992.
7. PROKHOROV, Danil V.: *Kalman Filter Training of Neural Networks: Methodology and Applications*, Ford Research Laboratory, 2002.