OPTIMÁLIS NEURÁLIS HÁLÓZAT KIVÁLASZTÁSA BAYES-BECSLÉS SEGÍTSÉGÉVEL

Kondics Milán[a], Szekeres Béla János[b*]

[a] *ELTE Informatikai, Kar, Programtervező informatikus MSc, 1. évf.*
[b] *ELTE Informatikai, Kar, Numerikus Analízis Tanszék, egyetemi adjunktus*

ABSZTRAKT

Ezen munkánkban célunk, hogy neurális hálózatokra alkalmazva a Bayes-becslést az *a posteriori* becslések során a különböző modellek közül kiválasszuk a tanító adatoknak legjobban megfelelőt. Mindehhez egy sokdimenziós integrál kiszámítása szükséges, amely a hagyományos Monte-Carlo módszerekkel is nehéz feladat; erre a célra a beágyazott mintavételezés (nested sampling) algoritmust alkalmazzuk, és a számítások járulékos eredményeként kapjuk meg a betanított hálózatot a hiperparaméterek terében is bolyongást végezve. Továbbá rámutatunk arra, hogyan lehet ötvözni a gradiens visszaterjesztéses és a véletlen bolyongásos tanítást hibrid hálózatokat nyerve.

Kulcsszavak: *neurális hálózat, nested-sampling, Bayes-tétel*

1. Bevezetés

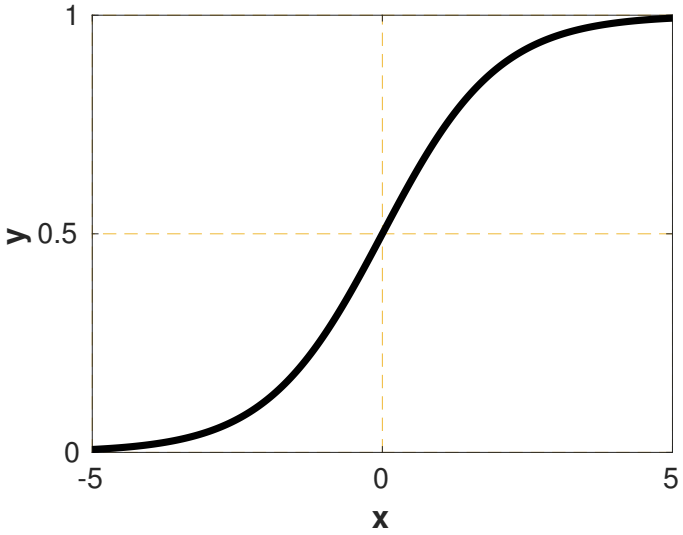
A gépi tanulás és a mesterséges intelligencia a számítástudomány sokak által kutatott területe modern világunkban [1]. A tudományos fantasztikus irodalom mellett számtalan valódi alkalmazása létezik, alkalmazzák például beszéd,- arcfelismerésre, vagy önvezető járműveknél [2]. A múlt század vége felé merült fel a gondolat biológiai alapokon működő számoló rendszerek létrehozására azzal a céllal, hogy megfigyelt mintákból vonjunk le tanulságot. De mit jelent tanulni? A neuronjaink tanulnak jelen ismereteink szerint; csoportokba szerveződnek, elektromos ingerek vezérlik a rendszerüket, úgy hogy a megfigyelt mintákra valamilyen értelemben optimálisan reagáljanak.

Ennek modellezéséhez az első fejezetben a szükséges matematikai eszközöket tekintjük át. A második fejezetben ismertetjük az alkalmazott algoritmust, melynek előnye, hogy neurális hálózatokra alkalmazva a Bayes-becslést [3] az *a posteriori* (utólagos) becslések során a különböző modellek közül kiválaszthatjuk a tanító adatoknak leginkább megfelelőt. Mindehhez egy sokdimenziós integrál kiszámítása szükséges, ez hagyományos Monte-Carlo módszerekkel nem megoldható, ezért beágyazott mintavételezést [4] alkalmazzuk. A módszer bár nem új, az újszerűségét alkalmazásának módja adja, pontosabban, hogy az optimalitást jellemző mérőszám meghatározására fókuszálunk, illetve a hálózat kimeneti rétegét nem véletlen bolyongás segítségével, hanem Tikhonov-regresszióval [5] tanítjuk. Célunk, hogy a szakirodalomban fellelhető módszerek többségével ellentétben a hiperparaméterek terében is bolyongva hasonlítsunk össze különböző hibrid neurális hálózatokat e mérőszám segítségével. Egy tesztfeladaton megmutatjuk, hogyan vizsgálható egyszerűen a segítségével a túlillesztés problémája. Numerikus szimulációkkal támasztjuk alá eredményeinket.

2. Felhasznált matematikai ismeretek

A neurális hálózatot irányított gráfként képzeljük el [2]. Tegyük fel, hogy balról jobbra haladnak az élek a csúcsok között, továbbá oszlopokba rendezzük a csúcsokat, ezeket az oszlopokat rétegeknek nevezzük. Az egyes rétegek között nem feltétlen fut az összes csúcs között él. A balszélső csúcsokat

*Kapcsolattartó: Szekeres Béla János, szekeres@inf.elte.hu
https://doi.org/10.37775/EIS.2021.1.5



1. ábra: A szigmoid függvény grafikonja

nevezzük bemeneti csúcsoknak, a jobbszélsőket pedig kimenetieknek. A gráf csúcsait neuronoknak nevezzük. Az élek súlyokkal rendelkeznek. A gráf összefüggő, nincs benne kör – a hálózatot ekkor előrecsatoltnak nevezzük – többszörös él, továbbá mindegyik kimeneti neuron elérhető valamely bemenetiből.

2.1. Többrétegű előrecsatolt hálózatok

Legyenek adottak $\alpha, \beta > 0$ valós számok. A hálózatban legyen N darab kimenő neuron, L darab bemenő, továbbá M darab $(\mathbf{x}, \mathbf{t})$ tanítóminta-párunk, ahol $\mathbf{x} \in \mathbb{R}^L$ bemeneti,- illetve $\mathbf{t} \in \mathbb{R}^N$ célvektorok. A hálózatunknak legyen W darab éle. Az éleken az összes súlyok vektorát jelöljük $\mathbf{w} \in \mathbb{R}^W$-vel. Legyen adott továbbá az $f_j : \mathbb{R} \to \mathbb{R}$ a j indexű nem bemeneti neuron aktivációs függvénye például $f_j(a) = \frac{1}{1+\exp(-a)}$, $f_j(a) = \tanh(a)$ vagy $f_j(a) = a$.

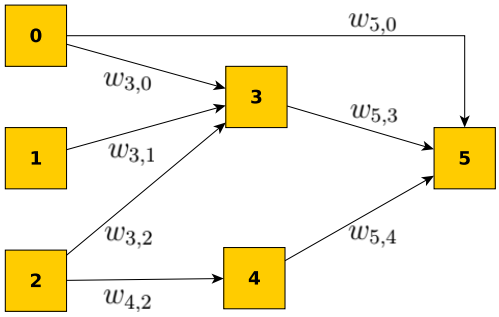
Az 1. ábrán láthatjuk az $f_j(x) = \frac{1}{1+\exp(-x)}$ szigmoid függvény grafikonját, amely az egyik leggyakrabban alkalmazott aktivációs függvény. A hálózat az agy működéséhez hasonlóan felfogható úgy, mint elektromos ingerek terjedése. A hálózat gráfjának élein értelmezett súlyok úgy képzelhetőek el, mint az adott élnek megfelelő szinapszis erőssége. Minél nagyobb a súly, annál jobban terjed rajta az inger, valamint minél nagyobb inger ér egy neuront, annál biztosabb az aktivációja, azaz lesz az értéke példánkban 1-hez közeli. Ha túl kicsi inger ér egy neuront, akkor az aktivációs függvény 0-ra állítja a neuron értékét, nem engedve a rajta áthaladó inger továbbterjedését.

A hálózat működése:

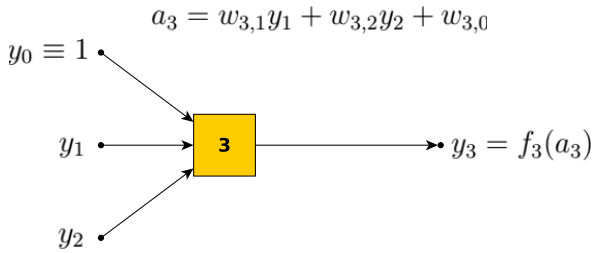
Ha a j indexű nem bemeneti neuronra az l indexű szülőjétől y_l nagyságú inger érkezik, és az őket összekötő szinapszis erőssége $w_{j,l}$, továbbá a neuron aktiválásához $w_{j,0}$ nagyságú küszöbinger szükséges, akkor az összes a neuronra ható inger összegezhető, mint

$$a_j = \sum_{l \in \{\text{j őseinek halmaza}\}} w_{j,l} y_l + w_{j,0}.$$

Az őt elhagyó hatás pedig $y_j = f_j(a_j)$ nagyságú. Amennyiben az l index bemeneti csúcsot jelöl, akkor y_l a neki megfelelő komponense az aktuális tanítómintapár $\mathbf{x}$ bemeneti vektorának. A küszöbinger felfogható úgy, mint egy, a hálózathoz adott extra 0 indexű bemeneti csúcs, amelyet az összes (nem bemeneti) neuronnal összekötünk. Ebből a j. csúcsba menő él súlya pedig $w_{j,0}$, továbbá adott $y_0 = 1$ bemeneti értékkel. 2. és 3. ábrán látható egy ilyen hálózat felépítése.



2. ábra: Neurális hálózat két bemeneti neuronnal (1 és 2 indexű) és egy kimenetivel (5 indexű)



3. ábra: A 3 indexű neuron y_3 kimenetének számítása. A 0 indexű csúcs felel meg az eltolási paraméternek, értéke $y_0 \equiv 1$

Az egyes jelölések magyarázata következik, amelyeket használni fogunk még:

- m: az m-edik mintát vizsgáljuk épp.
- $y_i^{(m)}$: az i-edik neuron kimenete az m-edik tanítómintapár bemenetét végigáramoltatva a hálózaton, egyben a j-edik neuron bemenete, ha i-ből vezet él j-be.
- $w_{j,i}$: az i-edik neuronból a j-edik neuronba mutató élen lévő súly. Minden csúcs bemenetéhez hozzáadunk mesterségesen egy $y_0^{(m)} = 1$ értéket, a j. csúcs esetén $w_{j,0}$ súllyal.
- $\mathbf{w} \in \mathbb{R}^W$ a $w_{j,i}$ súlyokból álló vektor.
- Az m-edik mintapár négyzetes hibája:

$$\mathcal{E}(m) = \frac{1}{2}\sum_{j=1}^{N}\left(t_j^{(m)} - y_j^{(m)}\right)^2,$$

ahol $t_j^{(m)}$ jelöli az m-edik tanítómintapárban az elvárt kimenet j-edik komponensét, az ennek megfelelő tényleges kimenet pedig $y_j^{(m)}$. Az átlagos négyzetes hiba pedig az összes mintapáron ezek átlaga, azaz

$$\overline{\mathcal{E}} = \frac{1}{M}\sum_{m=1}^{M}\mathcal{E}(m). \tag{1}$$

Az átlagos négyzetes középhiba pedig ennek négyzetgyöke. A hálózat tanításának célja, hogy az (1) formulában szereplő hibafüggvényt minimalizáljuk. Erre a célra a Metropolis-algoritmust [6] fogjuk közvetve alkalmazni a gyakorlatban elterjedt gradiens visszaterjesztés helyett. Ennek okát a beágyazott mintavételezési algoritmust tárgyaló részben láthatjuk, mely szerint úgy szeretnénk a hálózat $\mathbf{w}$ súlyait véletlenszerűen változtatni, hogy közben csökkentsük a hibát és a súlyok az *a priori* (jósló) eloszlásból származzanak. A Metropolis–algoritmust a következő részben ismertetjük.

2.2. A Metropolis–Hastings algoritmus

Egy példán keresztül vizsgáljuk az algoritmust, amelyet a [6] munkában ismertettek először. A feladat az, hogy generáljunk véletlen mintákat egy képlettel adott eloszlásból, azaz $x \sim \pi(x)$. Legyen például $\pi(x)$ a következő eloszlás:

$$\pi(x) = \frac{exp(-x^2)\,(2 + \sin(5x) + \sin(2x))}{\int_{-\infty}^{\infty} exp(-x^2)\,(2 + \sin(5x) + \sin(2x))\,\mathrm{d}x}. \tag{2}$$

A probléma az, hogy már a nevezőben szereplő integrált se könnyű kiszámítani. A Metropolis–Hastings algoritmus lényege, hogy készítünk egy Markov-láncot (jelöljük a tagjait úgy, hogy

x_0, x_1, x_2,...), amelynek a stacionárius eloszlása éppen a kívánt π eloszlás. Azaz ha $n \to \infty$, akkor $x_n \sim \pi(x)$ teljesülni fog. Tegyük fel, hogy a Markov-lánc aktuális eleme x_n; mi x_{n+1}-et szeretnénk előállítani. Szükségünk van tehát egy x^* jelöltre, ehhez pedig egy $Q(x^*|x_n)$ *jelölt eloszlásra*, ami feltételesen függ az aktuális x_n állapottól. Egy lehetséges klasszikus választás a jelölt eloszlásra egy x_n-re centrált normális eloszlás, azaz

$$x^*|x_n \sim \mathcal{N}(x_n, \sigma^2),$$

a σ értéket „ügyesen" kell megválasztani. Tehát vegyünk egy x^* mintát ebből a jelölt eloszlásból, ezt valamilyen α valószínűséggel elfogadjuk, mégpedig:

$$\alpha = \min\left(1, \frac{\pi(x^*)}{\pi(x_n)} \frac{Q(x_n|x^*)}{Q(x^*|x_n)}\right).$$

Ez azt fogja eredményezni, hogy a (2) formulában szereplő normalizáló konstanssal nem is kell foglalkozni, mert a π eloszlás nevezőjében lévő integrál az előbbi törtben kiegyszerűsödik. A Q eloszlást kell még jól megválasztani, ha Q szimmetrikus, mint a normális eloszlás esetén, akkor $\frac{Q(x_n|x^*)}{Q(x^*|x_n)} = 1$. Szimmetrikus Q esetén a módszert Metropolis-algoritmusnak nevezzük.

Ezután generálunk egy 0 és 1 közötti u véletlen számot egyenletes eloszlásból. Amennyiben $u \leq \alpha$, akkor x^*-ot elfogadjuk x_{n+1}-ként, egyébként pedig $x_{n+1} := x_n$. A Metropolis-algoritmus pszeudokódját az 1. algoritmus mutatja be.

2.3. Bayesi neurális hálózatok

Ebben a részben a Bayesi neurális hálózatok elméletét ismertetjük a [3] munka alapján. Tekintsünk a továbbiakban előrecsatolt hálózatokat. A cél, hogy a $D = \left\{\mathbf{x}^{(\mathbf{m})}, \mathbf{t}^{(\mathbf{m})} : \mathbf{x}^{(\mathbf{m})} \in \mathbb{R}^L, \mathbf{t}^{(\mathbf{m})} \in \mathbb{R}^N \text{ és } 1 \leq m \leq M\right\}$ rendelkezésre álló adatokon a hálózaton $\mathbf{w} = (w_1, ..., w_W)$ optimálásával – W a gráf éleinek száma – minimalizáljuk az összes hibát, azaz az alábbi függvényt.

$$E_{\beta,D}(\mathbf{w}) = \frac{\beta}{2} \sum_{m=1}^{M} \sum_{n=1}^{N} \left(t_n^{(m)} - y_n(\mathbf{x}^{(m)}, \mathbf{w})\right)^2, \tag{3}$$

ahol $t_n^{(m)}$ jelöli a $\mathbf{t}^{(m)}$ m-edik célvektor n-edik komponensét.

1. Algoritmus: A Metropolis-algoritmus x kezdőmintából indítva, F lépésszámmal

$x_1 := x$;
for *n=1:F* **do**
  generáljunk mintát a jelölt eloszlásból, azaz $x^*|x_n \sim Q(x^*|x_n)$;
  $\alpha = \min\left(1, \frac{\pi(x^*)}{\pi(x_n)}\right)$;
  generáljunk egy u véletlen számot 0 és 1 között egyenletes eloszlásból;
end
if $u \leq \alpha$ **then**
  $x_{n+1} := x^*$
else
  $x_{n+1} := x_n$
end
return x_{F+1}

Hozzáadunk $E_{\beta,D}(\mathbf{w})$-hez még egy $E_\alpha(\mathbf{w})$ regularizáló tagot is, amely

$$E_\alpha(\mathbf{w}) = \frac{\alpha}{2}\sum_{j=1}^{W} w_j^2. \tag{4}$$

Az α/β hányados nagysága azt mutatja meg, hogy mennyire büntetjük a nagy súlyokat a hiba minimalizálásához képest. Az ötlet az, hogy $\mathbf{w}$-re tekintsünk most úgy, mintha valamilyen ismert *a priori* speciális eloszlásból származó valószínűségi változó volna, majd a Bayes-becslést használva előállíthatjuk az *a posteriori* eloszlását $\mathbf{w}$-nek. Ami azt jelenti, hogy a tanító adatoknak jobban megfelelő $\mathbf{w}$-t kapunk.

Legyen tehát a súlyok *a priori* eloszlása $1/\alpha$ szórásnégyzetű normális eloszlás, azaz a súlyok együttes sűrűségfüggvénye úgy írható, mint

$$p(\mathbf{w}|\alpha) = \prod_{i=1}^{W} p(w_i|\alpha) = \frac{\exp\left(-E_\alpha(\mathbf{w})\right)}{(2\pi)^{W/2}\alpha^{-W/2}}. \tag{5}$$

A gráfot leíró modellt jelölje röviden $\mathcal{G}$, tegyük fel továbbá, hogy a tanítás során a D adatok normális eloszlásúak $y_n^{(m)}(\mathbf{x}^{(m)}, \mathbf{w})$ ($n = 1, ..., N$ és $m = 1, ..., M$) várható értékkel és $1/\beta$ szórásnégyzettel, azaz írhatjuk hogy

$$p(D|\mathbf{w}, \beta, \mathcal{G}) = \prod_{m=1}^{M}\prod_{n=1}^{N} p\left(t_n^{(m)}|\mathbf{x}^{(m)}, \mathbf{w}, \beta\right) = \frac{1}{(2\pi)^{MN/2}\beta^{-MN/2}}\exp(-E_{\beta,D}(\mathbf{t}, \mathbf{w})). \tag{6}$$

A hálózat klasszikus tanítása során azt a $\mathbf{w}^*$ súlyt keressük, melyre $p(D|\mathbf{w}^*, \beta, \mathcal{G})p(\mathbf{w}|\alpha, \mathcal{G})$ vagy $p(D|\mathbf{w}^*, \beta, \mathcal{G})$ maximális volt, azaz amely mellett a legnagyobb valószínűséggel figyelhetjük meg a D adatokat. A Bayes-tételt felhasználva viszont azt a $\mathbf{w}$ súlyt fogjuk megkeresni, amely a legvalószínűbb a D adatok ismeretében.

Megjegyzés: Folytassuk az alábbi sorozatot:

$$3,\ 7\,, 11,\ a_4 = ?.$$

A legtöbben rávágják, hogy 15 a következő elem. De ide bármilyen $a_4 = a$ számot írhatnánk igazából, hogyha a sorozat következő tagját egy interpoláló polinommal állítjuk elő (pontosabban fogalmazva, létezik olyan f harmadfokú polinom, amelyre $f(1) = 3$, $f(2) = 7$, $f(3) = 11$ és $f(4) = a$). Mégis, az első választ érezzük a helyénvalónak. Belátható a Bayes-tétel és valószínűségszámítási megfontolások segítségével, hogy valóban a 15 a legvalószínűbb válasz. Ez Occam borotvájának elve. Ha több lehetséges magyarázat is létezik egy jó válaszra, akkor a valószínűbbet fogadjuk el. Hogy melyik a legvalószínűbb, azt a Bayes-tétel segítségével határozhatjuk meg.

A Bayes-tétel:

Jelölje $p(\theta)$ a θ keresett paraméter *a priori* eloszlását, mielőtt még látnánk a megfigyelési adatokat. Továbbá, $p(D|\theta)$ annak a valószínűsége, hogy a D adatokat figyeljük meg ismerve θ-t. A Bayes-tétel felhasználásával a következőképpen határozhatjuk meg θ *a posteriori* eloszlását az adatok ismeretében:

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)}.$$

Neurális hálózatokra alkalmazva a súlyok *a posteriori* eloszlása így

$$p(\mathbf{w}|D, \alpha, \beta, \mathcal{G}) = \frac{p(D|\mathbf{w}, \beta, \mathcal{G})p(\mathbf{w}|\alpha, \mathcal{G})}{p(D|\alpha, \beta, \mathcal{G})} = \frac{p(D|\mathbf{w}, \beta, \mathcal{G})p(\mathbf{w}|\alpha, \mathcal{G})}{\int_{\mathbb{R}^W} p(D|\mathbf{w}, \beta, \mathcal{G})p(\mathbf{w}|\alpha, \mathcal{G})\,\mathrm{d}\mathbf{w}}. \tag{7}$$

Azt a $\mathbf{w}^*$-t keressük, amelyre ez maximális, azaz az adott feltételek és megfigyelési adatok mellett a legvalószínűbb. Ez rögzített α, β hiperparaméterek és $\mathcal{G}$ gráfszerkezet mellett akkor maximális, ha a tört számlálója, $p(D|\mathbf{w}^*, \beta, \mathcal{G})P(\mathbf{w}^*|\alpha, \mathcal{G})$ maximális, a nevező ugyanis független $\mathbf{w}$-től. Ez azt jelenti, hogy a hálózat tanítása során éppen a hibát minimalizáló optimális $\mathbf{w}^*$ súly a legvalószínűbb.

A hiperparaméterekről:

Felmerül a kérdés, hogy az α, β hiperparamétereket hogyan válasszuk meg, hiszen a (7) formulában ezeket *a priori* ismertnek tételeztük fel. Két lehetőségünk van:

(a) Felváltva megkeressük az optimális $\mathbf{w}$ súlyokat - akár visszaterjesztéses módszerrel, akár véletlen kereséses eljárással - rögzített α, β értékek mellett és fordítva, azaz az imént talált $\mathbf{w}$ súlyokat rögzítve megkeressük a legjobb α, β paramétereket. Ezt iterálva eljuthatunk egy optimális $\mathbf{w}_{\text{opt}}$, α_{opt}, β_{opt} hármashoz.

(b) Alkalmazhatjuk a Bayes-tételt α és β mentén marginalizálva, azaz

$$p(\mathbf{w}|D, \mathcal{G}) = \frac{1}{p(D|\mathcal{G})} \int_0^\infty \int_0^\infty p(D|\mathbf{w}, \beta, \mathcal{G}) p(\mathbf{w}|\alpha, \mathcal{G}) p(\alpha) p(\beta) \, \mathrm{d}\alpha \; \mathrm{d}\beta \,, \tag{8}$$

ahol

$$p(D|\mathcal{G}) = \int_{\mathbb{R}^W} \int_0^\infty \int_0^\infty p(D|\mathbf{w}, \beta, \mathcal{G}) p(\mathbf{w}|\alpha, \mathcal{G}) p(\alpha) p(\beta) \, \mathrm{d}\alpha \; \mathrm{d}\beta \; \mathrm{d}\mathbf{w} \,. \tag{9}$$

A gráf szerkezetének megválasztása:

Tegyük fel, hogy rendelkezésünkre állnak a $\mathcal{G}_1$ és $\mathcal{G}_2$ gráf szerkezetekhez tartozó (9) egyenlőségbeli $p(D|\mathcal{G}_1)$ és $p(D|\mathcal{G}_2)$ mennyiségek, vagyis hogy mekkora valószínűséggel figyeljük meg a D adatokat az *a priori* feltételezett két különböző gráf mellett. Nekünk arra volna szükségünk, hogy látva a D adatokat, meghatározzuk az adatoknak legjobban megfelelő gráfot.

Alkalmazva a (9) formulát és a Bayes-tételt összehasonlíthatjuk a két gráf valószínűségét az adatok megismerése után. Amennyiben ugyanolyan valószínűséggel választunk két különböző gráfszerkezetet, meghatározhatjuk az alábbi hányadost.

$$\begin{aligned} \frac{p(\mathcal{G}_1|D)}{p(\mathcal{G}_2|D)} &= \frac{p(\mathcal{G}_1)}{p(\mathcal{G}_2)} \frac{p(D|\mathcal{G}_1)/p(D)}{p(D|\mathcal{G}_2)/p(D)} = \\ &= \frac{\int_{\mathbb{R}^W} \int_0^\infty \int_0^\infty p(D|\mathbf{w}, \beta, \mathcal{G}_1) p(\mathbf{w}|\alpha, \mathcal{G}_1) p(\alpha) p(\beta) \, \mathrm{d}\alpha \; \mathrm{d}\beta \; \mathrm{d}\mathbf{w}}{\int_{\mathbb{R}^W} \int_0^\infty \int_0^\infty p(D|\mathbf{w}, \beta, \mathcal{G}_2) p(\mathbf{w}|\alpha, \mathcal{G}_2) p(\alpha) p(\beta) \, \mathrm{d}\alpha \; \mathrm{d}\beta \; \mathrm{d}\mathbf{w}} , \end{aligned} \tag{10}$$

amely ha nagyobb egynél, akkor a $\mathcal{G}_1$ gráf írja le jobban az adatokat. Ezen (9)-beli integrálok még Monte-Carlo integrálással is nehezen számolhatóak, ezért beágyazott mintavételezést alkalmazunk a numerikus approximációhoz és mintegy járulékos eredményként kapjuk meg az optimális $\mathbf{w}$ súlyokat és α, β hiperparamétereket.

2.4. A beágyazott mintavételezés

A feladat a (9) integrál kiszámítása, erre a célra a beágyazott mintavételezést fogjuk alkalmazni, melyet az [4] munka alapján ismertetünk. A feladat egy többdimenziós $\int_{\mathbb{R}^k} L(\theta)\pi(\theta)\mathrm{d}\theta$ alakú integrál kiszámítása, L-t a továbbiakban likelihood függvénynek nevezzük, π pedig valamilyen k dimenziós eloszlás. Legyen N adott, ekkor az eljárás pszeudokódját a 2. algoritmus szemlélteti.

2. Algoritmus: A beágyazott mintavételezés algoritmusa tol toleranciaküszöbbel

Húzunk N darab mintát a π *a priori* eloszlásból és kiszámítjuk L értékeit, azaz adottak $L(\theta_1), \ldots, L(\theta_N)$ kezdeti likelihoodok.
$Z = 0$; $X_0 = 1$; $j = 0$;
while $max\{L_1, \ldots, L_N\}\, X_j < tol \cdot Z_j$ **do**
  $j = j + 1$; $X_j = \exp(-j/N)$; $w_j = X_{j-1} - X_j$;
  Keressük meg a legkisebb likelihood értéket, ez tegyük fel, hogy θ_i esetén vétetik fel;
  $Z = Z + L(\theta_i) w_j$;
  Keressünk az *a priori* eloszlásból olyan θ értéket, amely független az N darab θ_j-től és $L(\theta) > L(\theta_i)$ teljesül rá. Cseréljük ki $L(\theta_i)$-t erre az $L(\theta)$-ra, illetve θ_i-t a talált θ-ra.
end
$Z = Z + X_j \left(L(\theta_1) + \ldots + L(\theta_N)\right) N^{-1}$;
return Z

2.5. Lineáris regresszió a kimeneti rétegen

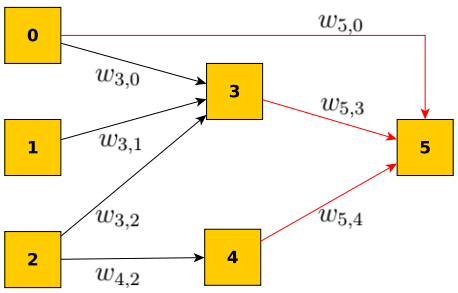
Ha olyan egyenletrendszert szeretnénk megoldani, amely több egyenletből áll, mint ismeretlent tartalmaz, azaz az $A\mathbf{x} = \mathbf{b}$ lineáris egyenletrendszer $A \in \mathbb{R}^{n \times m}$ mátrixára $n > m$ teljesül, akkor lehetőségünk van az általánosított inverz segítségével az euklideszi normában legjobban approximáló megoldást előállítanunk az $\tilde{\mathbf{x}} = (A^T A)^{-1} A^T \mathbf{b}$ alakban. Az $\tilde{\mathbf{x}}$ minimalizálja a $h(\mathbf{x}) = \|A\mathbf{x} - \mathbf{f}\|_0^2$ függvényt, ahol $\|\cdot\|_0$-val az euklideszi normát jelöljük. Előfordulhat, hogy a $g(\mathbf{x}) = \|A\mathbf{x} - \mathbf{f}\|_0^2 + \gamma\|\mathbf{x}\|_0^2$ függvényt szeretnénk minimalizálni, ahol $\gamma \geq 0$. Ezt Tikhonov-regularizáció segítségével tehetjük meg, a minimumhelyet az $\tilde{\mathbf{x}} = (A^T A + \gamma I)^{-1} A^T \mathbf{b}$ formulával nyerhetjük [5], ahol $I \in \mathbb{R}^{n \times n}$.

A 4. ábrán látható példában a kimeneti csúcs az 5 indexű, a piros színűek a kimeneti élek, melyek súlyai $w_{5,0}$, $w_{5,3}$ és $w_{5,4}$. Olyan hálózatokat vizsgáltunk, ahol a kimeneti csúcsok aktivációs függvénye az identitás, így a kimeneti élek súlyait Tikhonov-regularizáció segítségével határozzuk meg, mely so-rán az $\tilde{x}$-et megadó formulába $\gamma = \frac{\alpha}{\beta}$ kerül. Gradiens visszaterjesztést is alkalmazhatunk ezeken az éleken, így építhetünk egy egész előrecsatolt hálót is, a háló többi élét pedig továbbra is véletlen bolyongás segítségével tanítjuk, hibrid hálót nyerve.

3. A fejlesztett algoritmus

Ebben a részben ismertetjük a teljes algoritmust. A gyakorlatban a (10) formulában szereplő integrálokat nem szokás kiszámolni, helyettük az α, β hiperparaméterek *a posteriori* eloszlásából következtetnek az optimális α_{opt}, β_{opt} értékekre, majd gradiens visszaterjesztés segítségével minimalizálják (3) és (4) hibaformulák összegét.

Az *a priori* eloszlások megválasztása nem egyszerű feladat, az egyszerűség kedvéért noninformatív eloszlást választottunk a [3] munka javaslata alapján: legyenek $\alpha \sim \mathcal{U}(0, I_\alpha)$, $\beta \sim \mathcal{U}(0, I_\beta)$ egyenletes



4. ábra: A kimeneti réteg szemléltetése

eloszlásúak valamilyen I_α és I_β pozitív számokkal. Kiszámítandó tehát a (9)-beli integrál, amelyet jelöljünk Z-vel. Ez az integrál $W+2$ dimenziós, ahol W a háló éleinek számát jelöli.
A beágyazott mintavételezés algoritmusában az $L(\mathbf{w},\alpha,\beta)$ likelihood függvény és a $\pi(\mathbf{w},\alpha,\beta)$ *a priori* eloszlás az alábbiak $\alpha\in[0,I_\alpha]$, $\beta\in[0,I_\beta]$ és $\mathbf{w}\in\mathbb{R}^W$ esetén.

$$L(\mathbf{w},\alpha,\beta)=\left(\frac{\beta}{2\pi}\right)^{MN/2}\exp\left(-\frac{\beta}{2}\sum_{m=1}^{M}\sum_{n=1}^{N}\left(t_n^{(m)}-y_n(\mathbf{x}^{(m)},\mathbf{w},\alpha,\beta)\right)^2\right)\text{ és}$$
$$\pi(\mathbf{w},\alpha,\beta)=\left(\frac{\alpha}{2\pi}\right)^{W/2}\frac{\exp\left(-\frac{\alpha}{2}\sum_{j=1}^{W}w_j^2\right)}{I_\alpha I_\beta} \tag{11}$$

A (11) formulában a hálózat $y_n(\mathbf{x}^{(m)},\mathbf{w},\alpha,\beta)$ kimenetének α és β argumentuma arra utal, hogy azt egyértelműen meghatározza a Tikhonov-regularizáció. A π *a priori* eloszlásunkból úgy generálunk egy $\theta\in\mathbb{R}^{W+2}$ véletlen vektort, hogy veszünk $\alpha\sim\mathcal{U}(0,I_\alpha)$, $\beta\sim\mathcal{U}(0,I_\beta)$ véletlen számokat (ezek lesznek θ utolsó két komponense), majd generálunk W darab standard normális eloszlású véletlen számot, és ezek mindegyikét megszorozzuk $\alpha^{-1/2}$-nel, így normális eloszlásúak lesznek $1/\alpha$ szórásnégyzettel. Adott θ helyen L úgy számolható, hogy a hálót kiértékeljük θ első W darab komponensével adott súlyszettel, a kimeneti élek súlyát pedig α/β paraméterű Tikhonov-regularizáció segítségével határozzuk meg, melyek θ utolsó két komponensei. A teljes algoritmust a 3. pszeudokódban foglaltuk össze.

3. Algoritmus: A fejlesztett algoritmus

Legyenek $\mathbb{R}^{W+2}\ni\theta_j\sim\pi$, $j=1,\dots,N$ véletlen vektorok. $Z:=0$ és $X_0=1$, $f\in\mathbb{R}$ toleranciaküszöb adott az integrálhoz, továbbá Q egy $W+2$ dimenziós jelölt eloszlás és $F\in\mathbb{N}$ fix lépésszám a Metropolis-algoritmushoz, azaz adottak $L(\theta_1),\dots,L(\theta_N)$ kezdeti likelihoodok.
$Z=0$; $X_0=1$; $j=0$;
while $max\{L_1,\dots,L_N\}\,X_j<tol\cdot Z_j$ **do**
  $j=j+1$; $X_j=\exp(-j/N)$; $w_j=X_{j-1}-X_j$;
  Keressük meg a legkisebb likelihood értéket, ez tegyük fel, hogy θ_i esetén vétetik fel.
  $Z=Z+L(\theta_i)w_j$;
  Legyen $\tilde\theta_0$ a tárolt θ_j $(j=1,\dots,N)$ elemek közül véletlenszerű;
  for $n=0:F$ **do**
    generáljunk véletlen $\tilde\theta^*$ mintát a Q jelölt eloszlásból, azaz $\tilde\theta^*|\tilde\theta_n\sim Q(\tilde\theta^*|\tilde\theta_n)$;
    számítsuk ki a γ elfogadási valószínűséget $\gamma=\min\left(1,\frac{\pi(\tilde\theta^*)}{\pi(\tilde\theta_n)}\right)$;
    generáljunk egy $u\sim\mathcal{U}(0,1)$ véletlen számot;
    if $u\le\gamma$ *és* $L(\tilde\theta^*)>L(\theta_i)$ **then**
      $\tilde\theta_{n+1}:=\tilde\theta^*$
    else
      $\tilde\theta_{n+1}:=\tilde\theta_n$
    end
  end
  Cseréljük ki $L(\theta_i)$-t erre az $L(\tilde\theta_{F+1})$-re, illetve θ_i-t a talált $\tilde\theta_{F+1}$-re.
end
$Z=Z+X_j\left(L(\theta_1)+\dots+L(\theta_N)\right)N^{-1}$;
return Z

1. táblázat: A [7] munka 9. táblázatában szereplő 5×2 keresztvalidáció eredménye

Módszer	1.	2.	3.	4.	5.	6.	7.	8.
Átlaghiba [MWh]	5,426	4,561	4,572	5,399	8,487	4,561	4,563	4,656
Módszer	9.	10.	11.	12.	13.	14.	15.	
Átlaghiba [MWh]	3,861	8,221	5,556	3,779	4,128	4,087	4,211	

4. Numerikus eredmények

Választott modellünk témája, egy alapterhelésen működő, kombinált ciklusú erőmű teljes töltésnél mért elektromos kimenetének becslése [7]. Ez egy általános approximációs probléma a gépi tanulás módszereivel, viszont megoldása fontos hatékonysági és gazdasági szempontból egyaránt. A rendszer működését négy fő paraméter befolyásolja, amelyeket bemeneti változóként használnak az adathalmazban. Ezek az adatok, egy több, mint hat éven át tartó mérési sorozat eredményei, összesen 9 568 darab bemeneti-kimeneti adatpárt tartalmaznak, ahol minden mérési adat egy óránkénti átlag, melyet az erőmű szenzorai rögzítettek. A bemenő adatok a környezet hőmérséklete, a légköri nyomás, a relatív nedvességtartalom, valamint a kiáramló gőznyomás.

A szerzők a különböző módszereket 5×2 keresztvalidáció segítségével hasonlították össze, ez azt jelenti, hogy a teljes adathalmazt 5 alkalommal véletlenszerűen két részre osztották, a halmaz egyik felét tanításra használták, a másik felén pedig tesztelték a tanított modellt, majd felcserélték a két halmaz szerepét. Az 5×2 keresztvalidációban a módszer hatékonyságát a 10 teszthalmazon vett hiba átlaga adja: minél kisebb, annál jobb a módszer. A [7] munkában 15 módszert hasonlítottak össze így a szerzők, az erre vonatkozó eredményeiket láthatjuk az 1. táblázatban.

Az alábbiakban ismertetjük szimulációs eredményeinket. Egy darab rejtett réteggel rendelkező előrecsatolt hálózatokat hasonlítottunk össze eltérő neuronszámmal a rejtett rétegben, ahol az aktivációhoz a szigmoid függvényt választottuk, valamint a kimeneti rétegen α/β paraméterű Tikhonov-regularizációt alkalmaztunk. Az adatsor véletlenszerűen kiválasztott felén végeztük a hálók tanítását az előző fejezetben ismertetett algoritmussal, az adatsor másik felén pedig kiértékeltük őket. Ezen eredmények a 2. táblázatban láthatóak, amelyekben a rejtett rétegbeli neuronok számát R jelöli, a tanító halmazon számolt négyzetes középhibát és a validációs hibát pedig T és V. A Bayes-faktor a $p(\mathcal{G}_R|D)/p(\mathcal{G}_{10}|D)$ kifejezés logaritmusát jelenti (a nagyobb érték a jobb).

5. Következtetések

Munkánk során azt vizsgáltuk, hogy neurális hálózatokra alkalmazva a Bayes-becslést az *a posteriori* becslések során a különböző modellek közül hogyan válasszuk ki a tanító adatoknak leginkább megfelelőt. Mindehhez egy sokdimenziós integrál kiszámítása szükséges, amelyre a beágyazott mintavételezést alkalmaztuk. Megmutattuk, hogyan lehet ötvözni a gradiens visszaterjesztéses és a véletlen bolyongásos tanítást hibrid hálózatokat nyerve. A módszer hasznosságát numerikus kísérletekkel támasztottuk alá. Szimulációink kimenetét a [7] munka 1. táblázatban látható eredményeivel összehasonlítva azt állíthatjuk, hogy az általunk alkalmazott módszer validálási hibája még a legkevesebb neuronszám esetén is a 15 módszerrel összevetve a 3. legjobb. A Bayes-faktor segítségével azt is meg-

2. táblázat: Szimulációs eredmények $N = 100$ hálózattal

R	10	20	30	40	50	60	70	80	90	100
T [MWh]	3,895	3,823	3,843	3,839	3,875	3,853	3,849	3,860	3,953	3,926
V [MWh]	4,062	4,000	4,017	3,984	4,056	4,032	3,994	4,043	4,080	4,075
Bayes-faktor	0,0	−41,4	38,3	82,4	−340,2	−161,0	−67,0	−212,6	−74,2	4,9

állapíthatjuk, hogy 40-nél több neuront alkalmazni a rejtett rétegben túlillesztéshez vezet, amelyet alátámaszt, hogy a validációs hibák ennél nagyobb hálózat esetén már nem csökkennek. A legkisebb validációs hibához tartozik a legnagyobb Bayes-faktor, melyet a 40 rejtett rétegbeli neuronnal rendelkező hálózat esetén figyelhetünk meg, vagyis ez az optimális választás az általunk vizsgált hálózatok közül.

6. Köszönetnyilvánítás

A kutatást a 2019-1.3.1-KK-2019-00011 számú projekt támogatta, ami a Nemzeti Kutatási Fejlesztési és Innovációs Alapból biztosított támogatással, a „KOMPETENCIA KÖZPONTOK LÉTREHOZÁSA- KUTATÁSI INFRASTRUKTÚRA FEJLESZTÉS” pályázati program finanszírozásában valósult meg. Továbbá ezen munka elkészítése során a Kormányzati Információs Fejlesztési Ügynökség (KIFÜ) szuperszámítógépes infrastruktúráját használtuk.

7. Irodalomjegyzék

[1] D.E. Rumelhart, G.E. Hinton, R.J. Williams, *Learning representations by back-propagating errors*, Nature 323, 1986, pp. 533-536, CrossRef

[2] Horváth G., Altrichter M., Horváth G., Pataki B., Strausz Gy., Takács G., Valyon J., *Neurális hálózatok*, Budapest, Panem Kiadó, 2006.

[3] C.M. Bishop, *Bayesian methods for neural networks*, in L. Tarassenko, E. Rolls, D. Sherrington (Eds.), Oxford Lectures on Neural Networks, Oxford University Press., 1995.

[4] J. Skilling, *Nested sampling for general Bayesian computation*, Bayesian Anal. 1(4), 2006, pp. 833-859, CrossRef

[5] D.L. Phillips, *A Technique for the Numerical Solution of Certain Integral Equations of the First Kind*, Journal of the ACM 9, 1962, pp. 84-96, CrossRef

[6] W.K. Hastings, *Monte Carlo Sampling Methods Using Markov Chains and Their Applications*, Biometrika 57, 1970, pp. 99-109, CrossRef

[7] P. Tüfekci, *Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods*, International Journal of Electrical Power & Energy Systems 60, 2014, pp. 126-140, CrossRef