VILNIAUS UNIVERSITETAS

MATEMATIKOS IR INFORMATIKOS FAKULTETAS

Kursinis darbas

Baltymų struktūros nustatymas naudojant neuroninį tinklą

Protein structure prediction with a neural network

Dovydas Kičiatovas

VILNIUS 2017

MATEMATIKOS IR INFORMATIKOS FAKULTETAS
MATEMATINĖS INFORMATIKOS KATEDRA

Darbo vadovas (pedagoginis vardas, vardas, pavardė) _____

Darbas apgintas (įrašoma data)

Registravimo NR. _____

Santrauka

Turinys

1 Įžanga

Dirbtinių neuroninių tinklų (ANN - artificial neuron network) universalumas leidžia juos pritaikyti įvairiausioms sritims. Dėl jų savybės aproksimuoti įvairaus tipo duomenis, kitaip tariant - "išmokti" ir pastebėti tam tikrus duomenų dėsningumus, mokslo pasaulyje sparčiai populiarėjančios neuronų tinklų sistemos naudojamos vaizdų atpažinimui, robotikai, didelės apimties duomenų klasifikavimui ir t.t. Sparčiai kylant įvairių biologinių eksperimentų metu nuskaitomų baltymų sekų kiekiui, atsiranda poreikis šių baltymų duomenų įvertinimą ir apdorojimą automatizuoti. Iš baltymo aminorūgščių sekos neuroniniai tinklai gali padėti nustatyti baltymo funkcijas, pastebėti tam tikrus sekos motyvus arba baltymus kaip nors klasifikuoti. Šiuo metu naudojami eksperimentiniai metodai, pavyzdžiui, struktūros nustatymo atveju, kristalografija, yra resursų atžvilgiu brangūs ir eikvojantys daug laiko. Neuroninių tinklų naudojimas gali šį darbą palengvinti kokiu nors priimtinu tikslumu spėjant baltymo struktūrą, siekiant priskirti tiriamiems baltymams prioritetus, jei kokiems nors poreikiams ieškoma konkretų šabloną atitinkanti struktūra.

Šiame kursiniame darbe apžvelgiama baltymų struktūrų nustatymo galimybė naudojantis neuroniniais tinklais. Taip pat pristatomas neuroninio tinklo modelis - atliekami matematiniai skaičiavimai ir jų principai, neuronų ir jų sluoksnių savybės, konfigūracija ir t.t., modelio pritaikymas baltymų struktūrų spėjimui iš baltymo sekos. Su asmeniniu nešiojamuoju kompiuteriu atliktų skaičiavimų rezultatai rodo, kad pakankamai geram spėjimo tikslumui (80-85 proc.) gauti, klasifikuojant baltymus į kelias pasirinktas klases pagal SCOP duomenų bazę, nereikia itin našių kompiuterių ar jų sistemų. Be to, naudojamas modelis lengvai leidžia tinklą pritaikyti norimam skaičiui pasirinktų baltymų klasių bei testuoti įvairias tinklo parametrų kombinacijas.

2 Neuroninio tinklo modelis

2.1 Tinklo dalys ir mokymosi principai

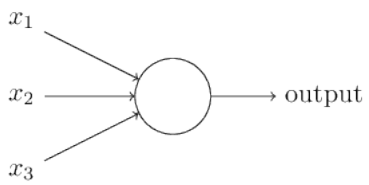
Visų pirma, apibrėžkime neuroninio tinklo modelį ir aptarkime jo sudedamąsias dalis. Tinklas gali būti apibrėžiamas keliais sluoksniais, iš kurių kiekvienas yra sudarytas iš dirbtinių neuronų. Sluoksniai jungiami sinapsėmis - tam tikrų svorių rinkiniu. Paskutinis sluoksnis atspindi tinklo grąžinamą išvestį. Kiekviena iš šių dalių atskirai sekančiuose skyreliuose aptariamos plačiau. Modelis paruoštas pagal Michael A. Nielsen[1].

2.1.1 Dirbtinis neuronas

Dirbtiniu neuronu vadinsime matematinę funkciją, atspindinčią realaus biologinio neurono veikimą. Biologinis neuronas gali turėti keletą jungčių (dendritų), atliekančių informacijos įvesties į neuroną funkciją, centrinę dalį, kurioje atliekamas skaičiavimas (somą) bei išvesties jungtį (aksoną), kuri atspindi neurono išvestį.

Dirbtinis neuronas, atitinkamai, gali turėti keletą įvesčių. Šios įvestys įgyja kokias nors reikšmes, pavyzdžiui, 0 arba 1. Kiekviena įvestis turi savo svorį - kokį nors realų skaičių. Svoriai sudauginami su įvesties reikšmėmis, susumuojami ir pagal kokią nors funkciją (dar vadinama aktyvacijos funkcija) yra paskaičiuojama neurono išvestis. Šiame kursiniame darbe naudojamo neuroninio tinklo dirbtiniai neuronai yra vadinami *sigmoid* neuronais - šis neuronas gali įgyti reikšmes intervale nuo 0 iki 1, o jo aktyvacijos funkcija yra *sigmoid* funkcija.

Apibrėžkime x_1, x_2, x_3 - trys *sigmoid* neurono įvestys. Tada schematiškai dirbtinis, tarp jų ir *sigmoid* neuronas, pavaizduotas 1 paveiksle.



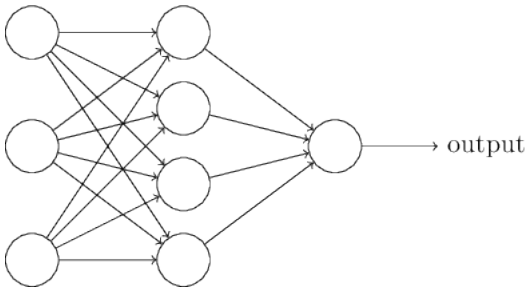
1 pav.: Dirbtinio neurono schema[1].

Kiekvienai įvesčiai priskyrę svorius w_1, w_2, w_3, bei apibrėžę z kaip atitinkamų įvesčių ir jų svorių sandaugų sumą, galime apibrėžti *sigmoid* funkciją: $\sigma(z) = \frac{1}{1+e^{-z}}$. Ši funkcija

ir bus neurono aktyvacijos funkcija. Prie z dar gali būti pridedamas poslinkis (*bias*), bet paprastumo dėlei šiame tinkle jis nenaudojamas[2].

2.1.2 Neuroninio tinklo sluoksniai ir sinapsės

Kiekvienas iš neuroninio tinklo sluoksnių yra sudaryti iš kokio nors skaičiaus dirbtinių neuronų (šiuo atveju - *sigmoid* neuronų). Visi kurio nors sluoksnio neuronai yra sinapsėmis sujungti su sekančio ir (arba) prieš tai buvusio sluoksnio visais neuronais, kaip pavaizduota 2 paveiksle. Paprastumo dėlei, įvestys į pirmąjį sluoksnį nėra vaizduojamos.



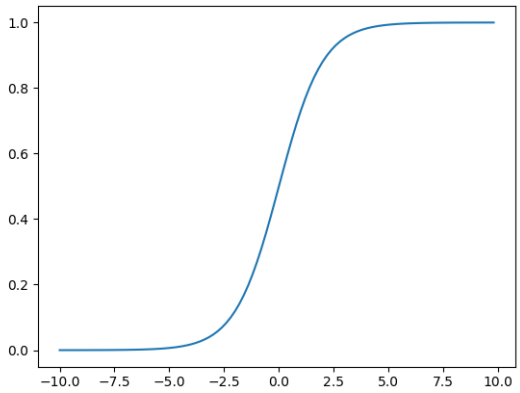
2 pav.: Paprasto 3 sluoksnių neuroninio tinklo schema[1].

Visi svoriai kartu tarp kurių nors dviejų sluoksnių vadinami sinapse (svorių rinkinių). Vienas svoris dar vadinamas sinaptiniu svoriu. Šių svorių kitimas yra esminė neuroninio tinklo savybė, reikalinga tinklo mokymui.

2.1.3 Neuroninio tinklo mokymas

Neuroninio tinklo mokymo proceso metu tinklui pateikiami pavyzdžiai ir pageidaujama išvestis - taikinys (*target*). Mokymo algoritmo esmė yra paklaidos minimizavimas tarp kiekvieno tinklui paduoto pavyzdžio ir jo taikinio. Šis procesas gali būti kartojamas nustatytą kartų skaičių (iteracijų). Kaip ir minėjau, tinklo mokymo procese yra naudojama *sigmoid* funkcija. Taip pat svarbi yra ir jos išvestinė - $\sigma(z)' = z(1-z)$. *Sigmoid* funkcija pavaizduota 3 paveiksle (grafikas nupieštas su Python Matplotlib biblioteka).

Turėdami kokį nors tašką (neurono įvesčių, padaugintų iš atitinkamų svorių, sumą) šios funkcijos grafike, pasinaudoję išvestine, galime sužinoti funkcijos gradientą tame taške, reikalingą minimizuojant paklaidai tarp neuroninio tinklo sluoksnių. Minimaliai paklaidai pasiekti bus keičiami svoriai, o tai, kiek svoris turi būti didinamas arba mažinamas, nurodys *sigmoid* funkcijos gradientas. Toks mokymosi tipas vadinamas gradientinio nusileidimo (*gradient descent*) mokymusi. Reikėtų atkreipti dėmesį, kad šio metodo paklaidų



3 pav.: *Sigmoid* funkcijos grafikas.

minimizacijos tikslas yra paklaidų funkcijos globalaus minimumo radimas, neretai kyla situacija, kai tinklas pakliūna į kokį nors lokalų minimumą ir nebegali iš jo išeiti. Mokymosi mechanizmas smulkiau paaiškintas kartu su pavyzdinio neuroninio tinklo programiniu kodu.

Galimi keli mokymo pavyzdžių pateikimo tinklui variantai - kiekvienoje iteracijoje visi pavyzdžiai tinklui gali būti pateikiami po vieną (stochastinis gradientas), visi kartu (*Full-Batch* gradientas) arba dalimis, t.y. visų pavyzdžių aibės pasirinkto dydžio poaibiais (*Mini-Batch* gradientas)[3]. Šių tipų įtaka mokymosi rezultatams apžvelgiama sekančiame skyriuje.

2.2 Neuroninio tinklo modelio realizacija

Šiame skyriuje bus aprašyti keli dirbtinio neuroninio tinklo realizacijos pavyzdžiai su skirtingais parametrais (sluoksnių skaičiumi ir t.t.) ir pateikti mokymosi rezultatai, jų priklausomybė nuo gradiento nusileidimo mokymo tipo. Šių tinklų užduotis - suskaičiuoti vienetų skaičių pateiktuose pavyzdžiuose: atsitiktinai sudarytuose nulių arba vienetų masyvuose. Kiekvieno masyvo dydis - 8 skaičiai, taigi, pavyzdžiai atspindi vieną baitą (8 bitai). Neuroninio tinklo įvesties sluoksnio neuronų skaičius sutampa su masyvo dydžiu, o išvesties sluoksnio dydis - 9 neuronai, iš kurių tik vienas (mokymo metu) turės vieneto reikšmę (kiti - nulio). Šio tinklo išvestis atspindi masyve esančių vienetų skaičių - taigi, jei pirmojo neurono reikšmė yra 1, vadinasi, tinklo įvestyje nėra nė vieno vieneto, jei antrojo reikšmė yra 1, tai tinklo įvestyje yra 1 vienetas ir t.t. iki paskutiniojo 9-to išvesties

neurono, kuris, jei įgis reikšmę 1, atspindės situaciją, kai visi įvesties masyvo elementai yra vienetai. Visi tinklai parašyti Python programavimo kalba (papildoma biblioteka - Numpy matricų, vektorių operacijoms ir t.t.).

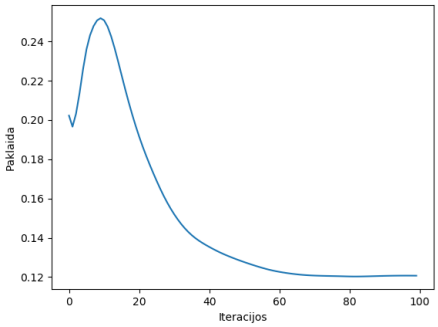
2.2.1 3-jų sluoksnių neuroninis tinklas

Pirmasis nagrinėjamas tokio modelio neuroninis tinklas turi 3 sluoksnius - 8 neuronų įvesties, 25 neuronų antrą sluoksnį (dar vadinamas paslėptuoju sluoksniu) ir 9 neuronų išvesties. Tinklo mokymui sukurti 200 pavyzdžių su atitinkamais taikiniais, o testavimui - 50 įvesčių, kurių vienetų skaičių bandys spėti pats tinklas, remdamasis tik išmoktais pavyzdžiais. Mokymas su visais pavyzdžiais vykdomas 5000 kartų, t.y. iteracijų. Mano paties parašytas (su Python programavimo kalba) šio neuroninio tinklo programinis kodas su paaiškinimais pateiktas 1 priede.

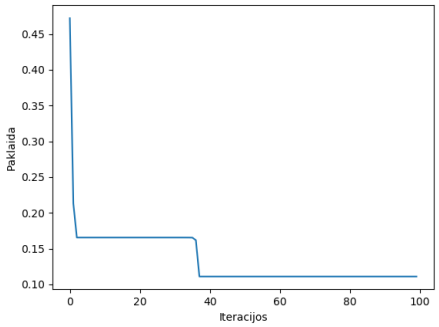
Sinapsės yra inicializuojamos su atsitiktinėmis reikšmėmis, siekiant sumažinti tikimybę, kad neuroninis tinklas nepakliūtų į lokalų minimumą, kaip jau minėjau praeitame skyriuje. Tinklo rezultatų priklausomybei nuo gradientinio nusileidimo įvertinti naudosime tinklo parametrus, nurodytus praeitoje pastraipoje. Atliekami 5 testai (1 testas - pilnas apmokymas ir spėjimai iš įvesties), skaičiuojamas kiekvieno testo metu gautų paskutinės iteracijos paklaidų vidurkis. Taip pat pateikti teisingų spėjimų dalis (procentai) ir pavienių individualių spėjimų taiklumo (kiek spėta reikšmė artima tikrajai) vidurkis. Galima sudaryti tokią lentelę (prie Mini-Batch skliausteliuose parašytas skaičius nurodo pavyzdžių poaibio dydį):

Grad. nusileidimo tipas	Paklaidų vid.	Teisingai atspėta	Ind. spėjimo tikslumas
Stochastinis	0.001	0.764	0.927
Full-Batch	0.111	0.168	0.596
Mini-Batch (20)	0.015	0.784	0.934

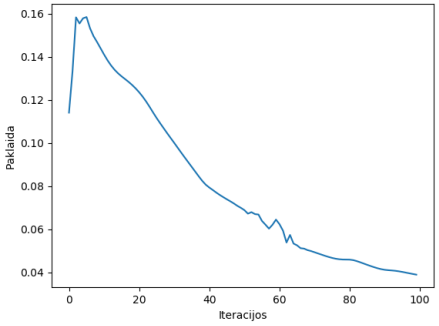
Matyti, kad prasčiausi rezultatai pasiekti naudojant *Full-Batch* gradientinio nusileidimo mokymo tipą. Nors ir paklaida nėra didelė, teisingai atspėtų įvesčių vienetų skaičiaus dalis yra itin maža, be to, negalime teigti, kad ir šie spėjimai nebuvo atsitiktiniai. Panašūs rezultatai pasiekti tarp stochastinio ir *Mini-Batch* gradientinių nusileidimų mokymų, tačiau naudojantis pastaruoju mokymas vyksta daug greičiau. 4 paveiksle pateikiami kiekvieno gradientinio nusileidimo mokymo tipo paklaidų vidurkio (1 testo) kitimo grafikai.



(a) Stochastinis tipas



(b) Full-Batch tipas



(c) Mini-Batch tipas

4 pav.: Paklaidų vidurkio kitimo grafikai

Matyti, kad stochastinio gradientinio nusileidimo mokymo tipo paklaidų vidurkio per 100 iteracijų grafikas yra stabiliausias. *Full-Batch* tipas kai kuriose iteracijose paklaidą mažina dideliais žingsniais. *Mini-Batch* tipo grafikas panašus į stochastinio tipo.

Atkreipkite dėmesį, kad sugeneruotų pavyzdžių yra 200, taigi, turint 8 bitų masyvą, šis skaičius nesiekia visų įmanomų tokio masyvo perstatų skaičiaus ($2^8 = 256$), be to, pavyzdžiai gali kartotis. Padidinus pavyzdžių skaičių, pavyzdžiui, iki 1000, teisingai atspėtų įvesčių dalis lengvai siekia 98 procentus.

2.2.2 4-ių sluoksnių neuroninis tinklas

Literatūra

[1] Michael A. Nielsen. Neural Networks and Deep Learning. Determination Press, 2015

[2] L. Howard Holley, Martin Karplus. *Protein secondary structure prediction with a neural network*. Proc. Natl. Acad. Sci. USA, Vol. 86, pp. 152-156, January 1989

[3] Sebastian Ruder. An overview of gradient descent optimization algorithms. arXiv:1609.04747v1 [cs.LG]

[4] Xueliang Leon Liu. Deep Recurrent Neural Networks for Protein Function Prediction from Sequence. arXiv:1701.08318v1 [q-bio.QM]