

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1966

**Optimizirane izlazne funkcije
klasifikatora temeljenog na
umjetnim neuronskim mrežama u
domeni implementacijskih napada
na kriptografske uređaje**

Juraj Fulir

Zagreb, svibanj 2019.

Umjesto ove stranice umetnite izvornik Vašeg rada.
Da bi ste uklonili ovu stranicu obrišite naredbu \izvornik.

ZAHVALA'n'STUFF

SADRŽAJ

1. Uvod	1
2. Implementacijski napadi na kriptografske uređaje	2
2.1. Side-channel napadi	2
2.2. Izvedba napada	2
2.3. DPA skupovi podataka	2
2.3.1. DPAv2	2
2.3.2. DPAv4	3
3. Klasifikator temeljen na umjetnim neuronskim mrežama	4
3.1. Umjetne neuronske mreže	4
3.1.1. Građa	4
3.1.2. Optimizacija umjetne neuronske mreže	6
3.1.3. Svojstva	12
3.1.4. Problemi	12
3.2. Izlazne funkcije	13
3.2.1. Funkcija identiteta	13
4. Optimizacija simboličkom regresijom (tehnički genetskim programiranjem...)	22
4.1. Simbolička regresija	22
4.2. Taboo evolucijski algoritam	22
4.3. Korišteni čvorovi (prostor pretraživanja)	22
5. Implementacija	24
5.1. Razvojna okolina i alati	24
5.2. Parametri	24
5.3. Evolucijski algoritmi	24
5.4. Neuronske mreže	24
5.5. Paralelizacija	25

5.6. Loggovi	25
6. Rezultati	26
6.1. 9class	26
6.1.1. Uobičajene izlazne funkcije	26
6.1.2. Utjecaj parametra veličine taboo liste	26
6.2. 256class	26
6.2.1. Uobičajene izlazne funkcije	26
6.2.2. Utjecaj parametra veličine taboo liste	26
7. Stvari koje sam probao, ali nisu ispale korisne	27
8. Buduća istraživanja	28
9. Zaključak	29
Literatura	30

1. Uvod

TODO: Opis problema

2. Implementacijski napadi na kriptografske uređaje

2.1. Side-channel napadi

TODO: *Postoji nekoliko vrsta.*

TODO: *Ovdje se obrađuje DPA.*

2.2. Izvedba napada

TODO: *Uštekaj uređaj, osciloskop na to i to mjesto i snimaj*

TODO: *Provjeri mogućnosti i zaključi najvjerojatniju*

TODO: *Problem netraktabilnosti postupka -> neuralke <3*

2.3. DPA skupovi podataka

TODO: *Tko i cilj**

TODO: *Ne zaboravi referencu na stranicu!*

2.3.1. DPAv2

TODO: *Kada je napravljen i ko ga je radil*

TODO: *Jel HW ili onaj pravi*

IMAGE: *PCA redukcija iz.jn*

IMAGE: Statistike iz jn

TODO: Mjere dobrote klasifikacije

2.3.2. DPAv4

TODO: Kada je napravljen i ko ga je radil

TODO: Jel HW ili onaj pravi

IMAGE: PCA redukcija iz jn

IMAGE: Statistike iz jn

TODO: Mjere dobrote klasifikacije

3. Klasifikator temeljen na umjetnim neuronskim mrežama

3.1. Umjetne neuronske mreže

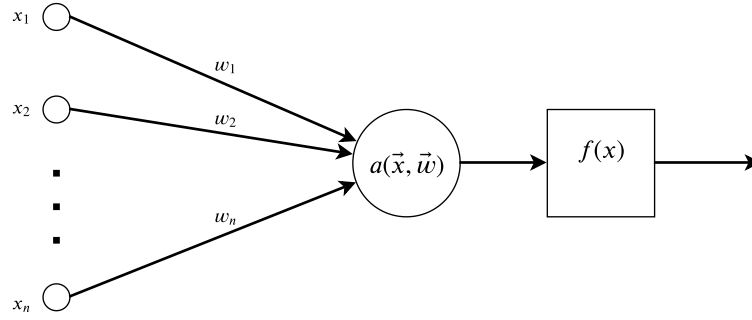
Umjetne neuronske mreže (nadalje „neuronske mreže“) koristimo za modeliranje više-dimenzijske funkcije ili distribucije kojom se aproksimira rješenje zadanog problema iz konačnog broja primjera. Vrlo su moćan alat za savladavanje teških zadataka u raznim područjima, često dostižući ljudske performanse na zadanom problemu. Danas su vrlo raširene u raznim područjima od kojih su samo neke: računalni vid (Krizhevsky et al., 2012; Redmon et al., 2016), prirodna obrada jezika (Mikolov et al., 2013; Kim, 2014) i podržano učenje (Mnih et al., 2013; Fang et al., 2017).

3.1.1. Građa

Neuronske mreže građene su od međusobno povezanih jedinica, tzv. neurona, modeliranih prema pojednostavljenom modelu biološkog neurona. Neuron očitava ulazne značajke sustava ili izlaze drugih neurona te ažurira svoje unutarnje stanje (aktivaciju) i stvara odziv (izlaz). Utjecaj ulaza na neuron vrednuje se težinama (engl. *weights*) koje definiraju kako se neuron ponaša u ovisnosti o pojedinim ulazima. Aktivacijski prag neurona (engl. *bias*) određuje jedinstvenu osjetljivost neurona na jačinu podražaja. Težine i prag neurona nazivamo parametrima neurona.

TODO: Što sve biolozi vele o neuronima? <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3812748/>

Način na koji iz ulaza gradimo unutarnje stanje (aktivaciju) neurona opisujemo aktivacijskom funkcijom. Najpopularnije aktivacijske funkcije jesu afina funkcija i unakrsna korelacija. Afina funkcija je skalarni produkt vektora ulaza s vektorom težina neurona uz dodatak vrijednosti praga. Parametri neurona definiraju nagib i pomak ravnine u prostoru ulaza koja opisuje aktivaciju neurona. Primijenjuje se kada se ulazi



Slika 3.1: Prikazani osnovni dijelovi neurona su težine dendrita (w_i), aktivacijska funkcija ($a(\vec{x}, \vec{w})$) i izlazna funkcija ($f(x)$). Prag neurona nije prikazan zbog jednostavnosti dijagrama.

u model mogu zapisati vektorom značajki čiji raspored nije bitan.

$$f(\vec{x}; \underline{W}, \vec{b}) = \underline{W}^T \cdot \vec{x} + \vec{b} \quad (3.1)$$

TODO: Spomeni distance based aktivacije (ANFIS?)

TODO: Spomeni i složenije metode: (Lin et al., 2014)

Pretvorbu aktivacije neurona u izlazni signal opisujemo izlaznom funkcijom koja se detaljnije obrađuje u poglavlju 3.2. Aktivacijska i izlazna funkcija definiraju prijenosnu funkciju koja ujedno opisuje ponašanje cijelog neurona (Duch i Jankowski, 1999). U praksi se pojam aktivacijske i prijenosne vrlo često ekvivalentno koristi na mjestu pojma izlazne funkcije, no ovaj rad se drži prethodno navedene i jasnije notacije.

$$t(x) = (f \circ a)(x) = f(a(x)) \quad (3.2)$$

Povezivanjem neurona gradi se arhitektura mreže koja određuje kako podatci i gradijenti teku kroz mrežu, a time utječu na brzinu učenja i inferencije neuronske mreže. Najčešće se koriste slojevite unaprijedne arhitekture zbog jednostavnosti izvedbe. Unaprijedne arhitekture propuštaju podatke samo u jednom smjeru odnosno već izračunati neuroni se ne izračunavaju ponovno, što je posebno pogodno za optimizaciju širenjem unatrag, detaljnije opisanu u poglavlju 3.1.2. Slojevite arhitekture omogućuju paralelizaciju izvođenja operacija na grafičkim karticama što značajno ubrzava postupke učenja i inferencije. Pri definiciji slojevite arhitekture najčešće je dovoljno navesti samo redoslijed slojeva, no ponekad je potrebno definirati i način povezivanja slojeva npr. pri uporabi preskočnih veza (Srivastava et al., 2015; He et al., 2016; Huang et al., 2017). Prvi sloj služi za postavljanje ulaza mreže i nazivamo ga ulaznim slojem mreže. Posljednji sloj mreže služi nam za ekstrakciju izlaza te mjere nje kakvoće mreže i nazivamo ga izlaznim slojem mreže. Svi slojevi između ulaznog i

izlaznog sloja nazivaju se skrivenim slojevima.

Potpuno povezana arhitektura je najjednostavnija arhitektura za zadatak klasifikacije. Svaki neuron u potpuno povezanom sloju aktivira se pomoću svih izlaza iz prethodnog sloja. Za naučeni potpuno povezani sloj kažemo da vrši ekstrakciju značajki iz svojih ulaza. Geometrijski gledano, svaki neuron vrši mapiranje značajki iz dimenzije prethodnog sloja u novu dimenziju s ciljem modeliranja boljih značajki.

TODO: Daj neki dokaz za ovo gore.

Unakrsna korelacija, za razliku od afine funkcije, koristi informaciju o susjednosti ulaznih značajki. Ulaz za takav model je definiran n-dimenzijskim tenzorom, a neuron uzima samo jedan pod-tenzor ulaza (vidljiva regija) i nad njime računa skalarni produkt s n-dimenzijskim tenzorom parametara (jezgrom). Kada su ulazi slike u boji ulazni tenzor ima 3 dimenzije (visina, širina, RGB kanali) pa stoga i jezgra ima 3 dimenzije, no znatno manje visine i širine. Unakrsna korelacija prozvana je konvolucijom jer radi na istom principu, a jedina razlika je da se elementi jezgre indeksiraju zrcaljeno po obje osi. S obzirom da se parametri jezgre uče automatski, nije nam bitno definirati orijentaciju jezgre.

$$f(\underline{X}; \underline{W}) = \underline{X} \circledast \underline{W} \quad (3.3)$$

TODO: Je li uopće potrebno spominjat konvoluciju?

TODO: Raspisat konvoluciju po elementima?

3.1.2. Optimizacija umjetne neuronske mreže

Optimizacijom parametara neuronska mreža prilagođava se danom zadatku, odnosno kažemo da mreža 'uči'. Optimizaciju parametara najčešće izvodimo gradijentnim spustom, uz pretpostavku derivabilnosti svih komponenata neuronske mreže. Kada ta pretpostavka ne vrijedi koriste se algoritmi kombinatorne optimizacije poput evolucijskih algoritama. U ovom radu optimizacija se vrši gradijentnim spustom.

Gradijentni spust

IMAGE: gradijentni spust unimodalna vs višemodalna (gdje preskoči brdo i uleti u bolji optimum)

Gradijentni spust je algoritam pronalaska minimuma funkcije vođen gradijentom te funkcije. Za zadanu početnu točku iterativno se pomiče u smjeru suprotnom od

gradijenta funkcije u toj točki dok ne zadovolji neki od uvijeta zaustavljanja. Na strmim funkcijama gradijent je često prevelik i može izazvati oscilaciju ili divergenciju (slika 3.1.2). Stoga se gradijent pri pomaku skalira s koeficijentom pomaka μ . Dobro odabran koeficijent pomaka može osigurati bržu konvergenciju, a kod višemodalnih funkcija i pronalazak boljeg optimuma (slika 3.1.2).

Početna točka utječe na ishod algoritma. Kod višemodalnih funkcija s optimumima različitih kvaliteta, početna točka može definirati u koji će lokalni optimum algoritam konvergirati (slika 3.1.2).

Input: funkcija $f(\vec{x})$

Input: početna točka \vec{x}_0

Input: koeficijent pomaka η

Input: broj iteracija n

for n iteracija **do**

$\vec{g}_i \leftarrow \vec{\nabla}_{\vec{x}} f(\vec{x}_i)$;

$\vec{x}_{i+1} \leftarrow \vec{x}_i - \eta \cdot \vec{g}_i$

end

Result: konačna točka \vec{x}_i je pronađeni optimum

Algorithm 1: Gradijentni spust

IMAGE: gradijentni spust stope spuštanja (velka, mala, taman)

Broj iteracija definira koliko se puta pomičemo iz početne točke, što definira i trajanje algoritma. Generalno želimo skratiti vrijeme pretraživanja te povećati koeficijent spusta kako bismo koristili manje pomaka. No u praksi najčešće nailazimo na višemodalne funkcije sa strmim regijama koje izazivaju oscilacije i mogu izazvati divergenciju. Stoga se češće koriste manji pomaci kroz više iteracija. Dodatno se mogu dodati modifikacije gradijenta koje nude ograničavaju veličinu gradijenta (odsijecanje gradijenta i sl.).

IMAGE: gradijentni spust sa početnim točkama (jedna ode u lok, jedna ode u glob, jedna zapne desno na platou)

Problem se javlja ako algoritam "odluta" u visoravan na kojoj su gradijenti vrlo mali, a sama regija je s obzirom na pomake ogromna (slika 3.1.2). Kad gradijent postane neupotrebljivo malen kažemo da je "iščeznuo". U takvim slučajevima pomaže dodavanje momenta koji se akumulira kroz više iteracija i dodaje vektoru gradijenta. Kad algoritam naiđe na regiju s vrlo malim gradijentima, moment pokušava izvući algoritam iz visoravni pomičući ga u smjeru koji je akumuliran. Kako moment ne

bi izvukao algoritam iz optimuma, dodaje mu se koeficijent "zaboravljanja" kojim se stari vektor momenta djelomično zaboravlja u korist novog vektora pomaka (jednadžba 3.4). Moment može pomoći i pri zaobilaženju lokalnih optimuma (slika 3.1.2).

$$\begin{aligned}\vec{v} &\leftarrow \alpha \cdot \vec{v} - \eta \cdot \vec{g} \\ \vec{x} &\leftarrow \vec{x} + \vec{v}\end{aligned}\tag{3.4}$$

IMAGE: *moment prije i na visoravni, moment za savladavanje brda*

Algoritam je primijenjiv na funkcije proizvoljne dimenzionalnosti uz pretpostavku derivabilnosti u svakoj točki. Za proizvoljnu realnu funkciju, uz dobro odabrane hiperparametre, algoritam će konvergirati u jedan od lokalnih optimuma, no algoritam generalno nema garanciju konvergencije u globalni optimum. Garanciju pronalaska globalnog optimuma nudi jedino za trivijalne unimodalne funkcije uz odgovarajuće hiperparametre algoritma (slika 3.1.2).

Problem odabira hiperparametara proizlazi iz činjenice da u generalno praksi nemamo definiranu funkciju koju minimiziramo (već samo skup uzoraka te funkcije) i/ili ju ne možemo jasno vizualizirati (kada radimo s funkcijama visoke dimenzionalnosti). Čak i da imamo definiranu funkciju najčešće ne znamo koju vrijednost poprima globalni optimum, a pohlepna pretraga je netraktabilna. Unatoč tome, gradijentni spust efikasno i učinkovito pronalazi optimume koji su dovoljno dobri za većinu praktičnih primjena (Redmon et al., 2016).

Funkcija gubitka

Pri učenju umjetnih neuronskih mreža potrebno je definirati funkciju gubitka. Funkcija gubitka, za dani ulaz, uspoređuje predikciju mreže sa željenim vrijednostima te dodjeljuje magnitudu pogreške (realan broj). Funkciju gubitka potrebno je pažljivo odabrati jer ona definira što je ishod učenja te utječe na učenje svakog parametra (kao što je opisano u poglavlju 3.1.2).

Najčešće ne znamo definiciju funkcije gubitka na čitavoj promatranoj domeni već posjedujemo samo uzorke te funkcije u podacima koje smo izmjerili i koje smatramo reprezentativnim. Ovdje pretpostavljamo da će neuronska mreža ostvariti svojstvo generalizacije, koje je detaljnije objašnjeno u poglavlju 3.1.3. Stoga se u narednim formulama koristi notacija sumiranja.

Funkcija gubitka često je usko vezana uz vrstu problema koji se rješava (klasifikacija, regresija i ostali), način učenja (nadzirano, polu-nadzirano, nenadzirano, podržano) i izlaznu funkciju posljednjeg sloja neuronske mreže. U ovom radu vrši se

klasifikacija nadziranom učenjem, no u nastavku se navodi i primjer funkcije gubitka za regresiju.

Uobičajeno se za funkciju gubitka koristi negativna logaritamska izglednost ...

TODO: oba imaju neg.log.izglednost, no pretpostavljaju razl. distribucije da dobiješ kvadratni (gauss) ili ovaj drugi loss (kategorička iliti Bernoullijeva)

U problemima regresije, najčešće se koristi funkcija kvadratnog gubitka koja računa odstupanje izlaza neuronske mreže od željenih vrijednosti. Uz ovaj gubitak najčešće se koristi funkcija identiteta u izlaznom sloju. Ova funkcija je brza i jednostavna

$$L(\theta) = \mathbb{E}_{(\vec{x}, \vec{y}) \in (X, Y)} \sum_i^{|\vec{y}|} (f(\vec{x}; \theta)_i - \vec{y}_i)^2 \quad (3.5)$$

U problemima klasifikacije, koristi se funkcija negativne logaritamske izglednosti.

$$L(\theta) = -\mathbb{E} \dots \quad (3.6)$$

TODO: prebaci očekivanja u SGD, ovdje spomeni samo fje gubitka

Optimizacija širenjem unatrag

Funkcija gubitka opisuje pogrešku čitave mreže te ovisi o svakom ugodljivom parametru mreže. Takva formulacija problema omogućuje nam da svaki parametar mreže ugađamo gradijentnim spustom. Dakle, za parametriziranu funkciju $f(x; \theta)$ tražimo one parametre θ^* za koje je gubitak najmanji na danim parovima ulaznih i izlaznih podataka.

$$\theta^* = \operatorname{argmin}_{\theta} L(x, y; \theta), \quad \forall (x, y) \in (X, Y) \quad (3.7)$$

Optimiranje gradijentnim spustom zahtjeva derivabilnost funkcije koju optimiziramo po ulazima, što izrazi (3.5) i (3.6) zadovoljavaju. Pri tome koristi se pravilo ulančavanja parcijalne derivacije kompozicije funkcija.

$$\frac{d}{dx} f(g(x)) = \frac{\partial f(g(x))}{\partial g(x)} \cdot \frac{\partial g(x)}{\partial x} \quad (3.8)$$

Derivacijom funkcije gubitka za regresiju (3.5) po ulazima, uz pretpostavku derivabilnosti čitave neuronske mreže po ulazima, dobivamo sljedeći izraz:

$$tODO \quad (3.9)$$

S obzirom da se mreža sastoji od ulančanih nelinearnih neurona s parametrima, gradijent gubitka moramo proslijediti sekvencijalno širenjem unazad (prema ulazima

u mrežu). Pojedini neuron možemo smatrati parametriziranom funkcijom koju je moguće prikazati grafom 3.1.2. Vidimo da se ulazni gradijent prolaskom kroz neuron širi na ostale elemente i na ulaze neurona koji vode do prethodnih neurona. Primijetimo i da se širi u suprotnom smjeru od toka podataka. Iz toga proizlazi naziv "širenjem unazad" (engl. *backpropagation*).

IMAGE: neuron kao graf + tok gradijenata

Želimo li učiti mrežu gradijentnim spustom, svaki parametar mreže treba imati pristup gradijentu funkcije gubitka. S obzirom da je neuron parametrizirana funkcija, pomoću koje ulančavanjem gradimo mrežu, dovoljno je pokazati da pojedini neuron osigurava svojim parametrima pristup gradijentu te da gradijent šalje svojim prethodnicima s kojima je povezan. Na slici 3.1.2 vidimo da je to ostvarivo, što dokazuju i izrazi:

$$\begin{aligned}\frac{\partial L(x, y; \theta)}{\partial s(x; w)} &= \frac{\partial L(x, y; \theta)}{\partial o(x; w)} \cdot \frac{\partial o(x; w)}{\partial s(x; w)} \\ &= \frac{\partial L(x, y; \theta)}{\partial o(x; w)} \cdot f'(x)\end{aligned}\tag{3.10}$$

$$\begin{aligned}\frac{\partial L(x, y; \theta)}{\partial x_i} &= \frac{\partial L(x, y; \theta)}{\partial s(x; w)} \cdot \frac{\partial s(x; w)}{\partial x_i} \\ &= \frac{\partial L(x, y; \theta)}{\partial o(x; w)} \cdot f'(x) \cdot w_i\end{aligned}\tag{3.11}$$

$$\begin{aligned}\frac{\partial L(x, y; \theta)}{\partial w_i} &= \frac{\partial L(x, y; \theta)}{\partial s(x; w)} \cdot \frac{\partial s(x; w)}{\partial w_i} \\ &= \frac{\partial L(x, y; \theta)}{\partial o(x; w)} \cdot f'(x) \cdot x_i\end{aligned}\tag{3.12}$$

$$\begin{aligned}\frac{\partial L(x, y; \theta)}{\partial w_0} &= \frac{\partial L(x, y; \theta)}{\partial s(x; w)} \cdot \frac{\partial s(x; w)}{\partial w_0} \\ &= \frac{\partial L(x, y; \theta)}{\partial o(x; w)} \cdot f'(x) \cdot 1\end{aligned}\tag{3.13}$$

Stohastički gradijentni spust

TODO: ne možemo čitav ds odjednom jer je prevelik

TODO: ne možemo stohastički jer je prešumovito, a i skupo na gpu

TODO: all hail the mini-batch, aproksimacija taman šumovita da pomaže, spomeni

usrednjavanje gradijenta po batchu

TODO: može i sampliranjem, ali ako imamo dataset kako znamo koliko koje semplirat

-> bolje po epohama

TODO: algoritam SGD

Optimizer

U prethodnim poglavljima opisan je postupak stohastičkog gradijentnog spusta i definicije gradijenta po parametrima, koji su potrebni za ispravno računanje

TODO: uporaba momenta i momenta na kvadrat (interpretacija)

TODO: Adam

TODO: smanjivanje stope učenja

Inicijalizacija parametara

Inicijalizacija parametara mreže je sinonim za odabir početne točke pri gradijentnom spustu.

TODO: važnost dobre inicijalizacije

TODO: Xavier

Regularizacija

TODO: geometrijski opis decizijske granice

TODO: podnaučena, generalizira, prenaučena

IMAGE: podnaučena, generalizira, prenaučena

Regularizacija parametara *TODO: L1*

TODO: L2

TODO: Spomeni pokoju još (adversarial iz Hintona, one za style transfer, ...)

Regularizacija šumom *TODO: inherentno zbog šumovitih ulaznih podataka*

TODO: može povećati dataset

Odabir hiperparametara

TODO: nepristranost

TODO: pretraživanje po rešetci

3.1.3. Svojstva

Univerzalna aproksimacija

TODO: univerzalna aproksimacija

TODO: VC dimenzija

TODO: proizvoljna složenost

Metoda optimizacije širenjem unazad nam omogućava izgradnju neuronskih mreža proizvoljne složenosti i raznovrsnih arhitektura (Krizhevsky et al., 2012; He et al., 2016; Huang et al., 2017).

Kompresija

TODO: kompresija

Generalizacija

TODO: generalizacija

Ovo je posebno izraženo kod klasifikacije slika gdje za jednu generičku klasu (npr. automobil) postoji neprebrojivo mogućih slika s različitim modelom, bojom ili kutom gledanja automobila. No mi raspolazemo sa svega nekoliko stotina tisuća uzoraka koje smatramo reprezentativnim za tu klasu i želimo izgraditi klasifikator koji je robustan na većinu perturbacija slike.

3.1.4. Problemi

Odabir hiperparametara

TODO: Problem odabira arhitekture, aktivacijske funkcije i hiperparametara

Arhitektura, prijenosne funkcije i parametri definiraju neuronsku mrežu te njihov odabir značajno utječe na performanse neuronske mreže. Učenje

Pretréniranost i neprijateljski primjeri

TODO: Problem pretréniranosti + neprijateljski primjeri

Derivabilne neuronske mreže optimiziraju se optimizatorom koji određuje kako se mijenjaju parametri. Za ugađanje parametara najčešće se koristi gradijentni spust, uz pretpostavku derivabilnosti čitave neuronske mreže. Kada pretpostavka ne vrijedi najčešće se koriste evolucijski algoritmi.

3.2. Izlazne funkcije

TODO: Bitka za odabir izlazne fje (nađi onaj rad di pljuje po sigmoidi i relu (elu rad?))

TODO: Usporedbe funkcije i derivacije

TODO: Navedene su funkcije koje su razmatrane

3.2.1. Funkcija identiteta

(engl. *Identity function*)

TODO: Dokaz da cijela mreža postaje linearna

TODO: koji problem rješava

TODO: svojstva

TODO: problemi

Ispravljena linearna jedinica (ReLU)

(engl. *Rectified linear unit*)

TODO: bez i sa cutoff

TODO: mrtvi neuroni

TODO: i dalje nije loša

IMAGE:

$$f(x) = \begin{cases} x, & \text{ako } x > 0 \\ 0, & \text{inače} \end{cases} \quad f'(x) = \begin{cases} 1, & \text{ako } x > 0 \\ 0, & \text{inače} \end{cases} \quad (3.14)$$

TODO: koji problem rješava

TODO: svojstva

TODO: problemi

Propusna ispravljena linearna jedinica (LReLU)

(engl. *Leaky ReLU*)

IMAGE:

$$f(x) = \begin{cases} x, & \text{ako } x > 0 \\ \alpha x, & \text{inače} \end{cases} \quad f'(x) = \begin{cases} 1, & \text{ako } x > 0 \\ \alpha, & \text{inače} \end{cases} \quad (3.15)$$

TODO: koji problem rješava

TODO: svojstva

TODO: problemi

Ispravljena linearna jedinica s pragom (ThReLU)

(engl. *Thresholded ReLU*)

IMAGE:

$$f(x) = \begin{cases} x, & \text{ako } x > \theta \\ 0, & \text{inače} \end{cases} \quad f'(x) = \begin{cases} 1, & \text{ako } x > \theta \\ 0, & \text{inače} \end{cases} \quad (3.16)$$

TODO: koji problem rješava

TODO: svojstva

TODO: problemi

(RReLU)

(engl. *Randomized leaky ReLU*)

IMAGE:

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Eksponencijalno-linearna jedinica (ELU)

(engl. *Exponential linear unit*)

IMAGE:

$$f(x) = \begin{cases} x, & \text{ako } x > 0 \\ \alpha(e^x - 1), & \text{inače} \end{cases} \quad f'(x) = \begin{cases} 1, & \text{ako } x > 0 \\ \alpha e^x, & \text{inače} \end{cases} \quad (3.17)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Skalirana eksponencijalno-linearna jedinica (SELU)

(engl. *Scaled exponential linear unit*)

IMAGE:

$$f(x) = \lambda \begin{cases} x, & \text{ako } x > 0 \\ \alpha(e^x - 1), & \text{inače} \end{cases} \quad f'(x) = \lambda \begin{cases} 1, & \text{ako } x > 0 \\ \alpha e^x, & \text{inače} \end{cases} \quad (3.18)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

(GELU)

(engl. *Gaussian error linear unit*)

IMAGE:

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Sigmoida (σ)

(engl. *Sigmoid*)

IMAGE:

$$f(x) = \frac{1}{1 + e^{-x}} \quad f'(x) = \sigma(x)(1 - \sigma(x)) \quad (3.19)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Tvrda sigmoida

(engl. *Hard sigmoid*)

IMAGE:

$$f(x) = \min(1, \max(0, 0.2x + 0.5)) \quad f'(x) = \begin{cases} 0.2, & \text{ako } x \in [-2.5, 2.5] \\ 0, & \text{inače} \end{cases} \quad (3.20)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Swish

IMAGE:

$$\begin{aligned} f(x) &= x\sigma(\beta x) & f'(x) &= \sigma(\beta x) + \beta x \cdot \sigma(\beta x)(1 - \sigma(\beta x)) \\ & & &= \frac{e^x \cdot (e^x + x + 1)}{(e^x + 1)^2} \end{aligned} \quad (3.21)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Tangens hiperbolni (tanh)

IMAGE:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad f'(x) = 1 - \tanh^2(x) \quad (3.22)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Tvrđi tangens hiperbolni

(engl. *Hard tanh*)

IMAGE:

$$f(x) = \begin{cases} -1, & \text{ako } x < -1 \\ x, & \text{ako } x \in [-1, 1] \\ 1, & \text{inače} \end{cases} \quad f'(x) = \begin{cases} 0, & \text{ako } x < -1 \\ 1, & \text{ako } x \in [-1, 1] \\ 0, & \text{inače} \end{cases} \quad (3.23)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Racionalna aproksimacija tanh

(engl. *Rational tanh*)

TODO: *DL4J ima bug u derivaciji ove fje*

IMAGE:

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

$$f(x) = 1.7159 \cdot \tanh\left(\frac{2}{3}x\right), \quad \text{gdje } \tanh(x) \approx \operatorname{sgn}(x) \left(1 - \frac{1}{1 + |x| + x^2 + 1.41645 \cdot x^4}\right) f'(x) = \quad (3.24)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Ispravljeni tanh

(engl. *Rectified tanh*)

IMAGE:

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Softmax

IMAGE:

$$f(\vec{x}) = \frac{e^{\vec{x}}}{\sum_i e^{\vec{x}_i}} \quad f'(x) = \frac{e^x}{1 + e^x} \quad (3.25)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Softplus

IMAGE:

$$f(x) = \log(1 + e^x) \quad f'(x) = \frac{e^x}{1 + e^x} \quad (3.26)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Softsign

IMAGE:

$$f(x) = \frac{x}{1 + |x|} \quad f'(x) = \frac{1}{(1 + |x|)^2} \quad (3.27)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Sinus (sin)

IMAGE:

$$f(x) = \sin(x) \quad f'(x) = \cos(x) \quad (3.28)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Kosinus (cos)

IMAGE:

$$f(x) = \cos(x) \quad f'(x) = -\sin(x) \quad (3.29)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Parabola x^2

IMAGE:

$$f(x) = x^2 \quad f'(x) = 2x \quad (3.30)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Kubna parabola x^3

IMAGE:

$$f(x) = x^3 \quad f'(x) = 3x^2 \quad (3.31)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

Gauss

IMAGE:

$$f(x) = e^{-x^2} \quad f'(x) = -2x \cdot f(x) \quad (3.32)$$

TODO: *koji problem rješava*

TODO: *svojstva*

TODO: *problemi*

4. Optimizacija simboličkom regresijom (tehnički genetskim programiranjem...)

4.1. Simbolička regresija

TODO: Opis i svojstva SR

TODO: Utjecaj i brojnost parametara u GA (moš linkat i svoj završni rad :P)

4.2. Taboo evolucijski algoritam

TODO: Problem konvergencije i stohastičnosti GP-a

TODO: EA oplemenjen taboo listom iz algoritma Taboo pretraživanja

4.3. Korišteni čvorovi (prostor pretraživanja)

U nastavku je dan popis čvorova koji ujedno definiraju prostor pretraživanja.

Naziv	Funkcija	Broj ulaza	Naziv	Funkcija	Broj ulaza
x	x	0	elu	$ELU(x)$	1
const	$c \in \mathbb{R}$	0	gauss	e^{x^2}	1
+	$x + y$	2	lrelu	$LReLU(x)$	1
-	$x - y$	2	relu	$ReLU(x)$	1
*	$x \cdot y$	2	selu	$SELU(x)$	1
/	$\frac{x}{y+1e-12}$	2	sigmoid	$\frac{1}{1+e^{-x}}$	1
min	$\min(x, y)$	2	softmax	$Softmax(x)$	1
max	$\max(x, y)$	2	sotplus	$Softplus(x)$	1
abs	$ x $	1	softsign	$Softsign(x)$	1
sin	$\sin(x)$	1	swish	$Swish(x)$	1
cos	$\cos(x)$	1	tanh	$\tanh(x)$	1
tan	$\tan(x)$	1			
exp	e^x	1			
log	$\log_e(x)$	1			
pow2	x^2	1			
pow3	x^3	1			
pow	x^y	2			

Tablica 4.1: Popis korištenih čvorova

5. Implementacija

5.1. Razvojna okolina i alati

TODO: *tulavi DL4J*

TODO: *IntelliJ <3 <3 <3*

TODO: *funkcije iscrtavam u jn*

5.2. Parametri

TODO: *da, možeš definirat parametre u datoteki*

TODO: *dodavanje parametara*

TODO: *grid search mehanizam*

5.3. Evolucijski algoritmi

TODO: *in-house EA okolina*

TODO: *glavne komponente koda*

IMAGE: *IntelliJ generirna UML paketa genetics*

5.4. Neuronske mreže

TODO: *glavne komponente*

TODO: *automatizirana pohrana rezultata*

IMAGE: *IntelliJ generirna UML paketa neurology*

5.5. Paralelizacija

TODO: workarbiter <3

TODO: sinkronizacija u GA i pomoćne klase/metode

5.6. Loggovi

TODO: kaj sve imaš i kak se koristi

6. Rezultati

6.1. 9class

6.1.1. Uobičajene izlazne funkcije

TODO: Opis postupka pretrage

TODO: Tablica

TODO: Komentar

6.1.2. Utjecaj parametra veličine taboo liste

TODO: Tablica

TODO: Komentar

6.2. 256class

6.2.1. Uobičajene izlazne funkcije

TODO: Opis postupka pretrage

TODO: Tablica

TODO: Komentar

6.2.2. Utjecaj parametra veličine taboo liste

TODO: Tablica

TODO: Komentar

7. Stvari koje sam probao, ali nisu ispale korisne

Učeći parametri

TODO: dokaz da na korištene funkcije nema utjecaja (stopi se s težinama ili biasom)

TODO: pokazati primjer fje gdje bi se mogao koristiti

Dropout

TODO: zahtjeva previše iteracija, što nije baš korisno u EA okruženju

TODO: probati maxout?

Tensorflow Java API

TODO: proba, ali je još u razvoju (puno toga je falilo)

8. Buduća istraživanja

TODO: Primjena CNN na vremenskim uzorcima po uzoru na onaj rad

TODO: Ispitivanje učinkovitosti korištene optimizacije na ostalim problemima

TODO: Paralelna evolucija arhitekture i aktivacijskih fja

9. Zaključak

TODO: Radi/Ne radi

TODO: Pronađene zanimljivosti

TODO: Pouka za doma

LITERATURA

Włodzisław Duch i Norbert Jankowski. Survey of neural transfer functions. *Neural Computing Surveys*, 2(1):163–212, 1999. URL ftp://ftp.icsi.berkeley.edu/pub/ai/jagota/vol2_6.pdf.

Meng Fang, Yuan Li, i Trevor Cohn. Learning how to active learn: A deep reinforcement learning approach. U *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, stranic 595–605, Copenhagen, Denmark, Rujan 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1063. URL <https://www.aclweb.org/anthology/D17-1063>.

K. He, X. Zhang, S. Ren, i J. Sun. Deep residual learning for image recognition. U *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, stranic 770–778, June 2016. doi: 10.1109/CVPR.2016.90.

G. Huang, Z. Liu, L. v. d. Maaten, i K. Q. Weinberger. Densely connected convolutional networks. U *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, stranic 2261–2269, July 2017. doi: 10.1109/CVPR.2017.243.

Yoon Kim. Convolutional neural networks for sentence classification. U *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, stranic 1746–1751, Doha, Qatar, Listopad 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1181. URL <https://www.aclweb.org/anthology/D14-1181>.

Alex Krizhevsky, Ilya Sutskever, i Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. U F. Pereira, C. J. C. Burges, L. Bottou, i K. Q. Weinberger, urednici, *Advances in Neural Information Processing Systems* 25, stranic 1097–1105. Curran Associates, Inc., 2012. URL <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-network.pdf>.

- Min Lin, Qiang Chen, i Shuicheng Yan. Network in network. *CoRR*, abs/1312.4400, 2014.
- Tomas Mikolov, Kai Chen, Greg S. Corrado, i Jeffrey Dean. Efficient estimation of word representations in vector space, 2013. URL <http://arxiv.org/abs/1301.3781>.
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, i Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013. URL <http://arxiv.org/abs/1312.5602>.
- J. Redmon, S. Divvala, R. Girshick, i A. Farhadi. You only look once: Unified, real-time object detection. U *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, stranice 779–788, June 2016. doi: 10.1109/CVPR.2016.91.
- Rupesh Kumar Srivastava, Klaus Greff, i Jürgen Schmidhuber. Highway networks. *CoRR*, abs/1505.00387, 2015.

Optimizirane izlazne funkcije klasifikatora temeljenog na umjetnim neuronskim mrežama u domeni implementacijskih napada na kriptografske uređaje

Sažetak

Proučiti postojeće metode u izgradnji izlaznih funkcija u umjetnim neuronskim mrežama. Posebnu pažnju posvetiti evolucijskim algoritmima simboličke regresije za izgradnju ciljanih funkcija. Ustanoviti moguće nedostatke postojećih algoritama ili mogućnost poboljšanja. Primijeniti evoluirane izlazne funkcije u homogenoj ili heterogenoj umjetnoj neuronskoj mreži na skupovima DPAv2 i DPAv4 te odrediti mjere kvalitete izgrađenog klasifikatora: točnost, preciznost, odziv te F mjere. Usporediti učinkovitost ostvarenih postupaka s postojećim rješenjima iz literature. Radu priložiti izvorne tekstove programa, dobivene rezultate uz potrebna objašnjenja i korištenu literaturu.

Ključne riječi: Ključne riječi, odvojene zarezima.

Title

Abstract

Abstract.

Keywords: Keywords.