SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 6846

NEPRIJATELJSKI NAPADI NA KLASIFIKACIJSKE MODELE UZ VIŠEKRITERIJSKU OPTIMIZACIJU

Bruna Dujmović

Zagreb, lipanj 2020.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 6846

NEPRIJATELJSKI NAPADI NA KLASIFIKACIJSKE MODELE UZ VIŠEKRITERIJSKU OPTIMIZACIJU

Bruna Dujmović

Zagreb, lipanj 2020.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 13. ožujka 2020.

ZAVRŠNI ZADATAK br. 6846

Pristupnica:	**Bruna Dujmović (0036505665)**
Studij:	Računarstvo
Modul:	Računarska znanost
Mentor:	prof. dr. sc. Domagoj Jakobović

Zadatak: **Neprijateljski napadi na klasifikacijske modele uz višekriterijsku optimizaciju**

Opis zadatka:

Opisati i istražiti vrste neprijateljskih napada na klasifikacijske modele, white-box i black-box napadi te njihove inačice. Posebnu pažnju posvetiti postojećim napadima na modele za klasifikaciju slike i osvrnuti se na njihovu robusnost. Proučiti područje višekriterijske optimizacije te opisati odabrane algoritme poput NSGA-II/NSGA-III i PPES. Ostvariti programski sustav za generiranje black-box napada na modele za klasifikaciju slike koristeći odabrane algoritme za višekriterijsku optimizaciju. Ispitati rad implementacije s obzirom na različite parametre i kriterije optimizacije te usporediti rezultate dobivene primjenom pojedinih algoritama. Radu priložiti izvorne tekstove programa, dobivene rezultate uz potrebna objašnjenja i korištenu literaturu.

Rok za predaju rada: 12. lipnja 2020.

SADRŽAJ

1. Uvod

Želimo li razvrstati svoju elektroničku poštu na spam i običnu poštu, prepoznati životinjsku vrstu prikazanu na fotografiji ili dijagnosticirati pacijente na temelju njihovih karakteristika, zapravo trebamo odrediti kojoj od nama poznatih kategorija (klasa) pripada predočeni "uzorak" (e-poruka, fotografija, karakteristike pacijenta). Razvijeni su brojni modeli za automatsko rješavanje takvih tzv. klasifikacijskih problema, jedan od kojih su i umjetne neuronske mreže. Točnost tih modela nije nužno savršena, a možemo ih pokušati i namjerno natjerati na pogrešnu klasifikaciju. U tom slučaju govorimo o neprijateljskim napadima na klasifikacijske modele, koji se temelje na izmjeni uzoraka do te mjere da ih model više ne uspijeva ispravno razvrstati.

Ovaj će se rad baviti upravo izradom takvih neprijateljskih uzoraka ne bi li se zavarale umjetne neuronske mreže naučene na MNIST skupu slika rukom pisanih znamenki. Neprijateljski napadi bit će opisani kao problemi višekriterijske optimizacije jer će se pokušati proizvesti uzorci koji su istovremeno "odlični" u odnosu na više postavljenih zahtjeva.

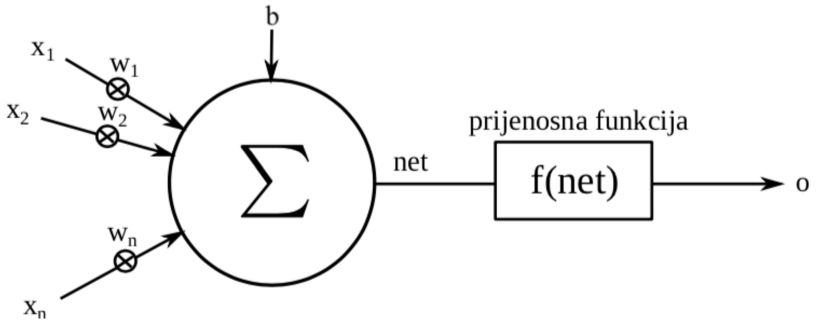
Pregled osnovnih koncepata vezanih uz umjetne neuronske mreže dan je u poglavlju 2. Treće poglavlje pobliže definira neprijateljske napade, opisuje metode obrane od njih i načine povećanja otpornosti napada na te obrane. Problemi višekriterijske optimizacije i algoritmi njihova rješavanja tema su 4. poglavlja. U 5. poglavlju opisano je popratno programsko ostvarenje, a rezultati dobiveni primjenom istog izneseni su u 6. poglavlju. Zaključak, popis literature te sažetci na hrvatskom i engleskom jeziku nalaze se na kraju rada.

2. Umjetne neuronske mreže

Umjetna neuronska mreža računalni je sustav nadahnut arhitekturom i funkcionalnošću biološkog mozga. Čini je skup jednostavnih, međusobno povezanih procesnih jedinica (umjetnih neurona) koje omogućavaju masovno raspodijeljenu obradu podataka [3].

2.1. Umjetni neuron

Umjetni neuron osnovna je gradivna jedinica umjetnih neuronskih mreža.



Slika 2.1: Model umjetnog neurona [13].

Analogno dendritima, krajevima tijela stanice biološkog neurona koji primaju i prosljeđuju podražaje iz okoline, model umjetnog neurona prima informacije kroz svoje ulaze $x_1, ..., x_n$. Djelovanje svakog od ulaza x_i modulirano je skalarima w_i, tzv. težinama (engl. *weights*), pa se utjecaj određenog ulaza na neuron može predočiti umnoškom $x_i \cdot w_i$. Ukupan utjecaj svih ulaza jest suma

$$net = \sum_{i=1}^{n} x_i \cdot w_i + b,$$

pri čemu je težina b poznata kao pomak ili prag (engl. *bias weight*). Taj se zbroj propušta kroz prijenosnu funkciju f (engl. *transfer function*, *activation function*) koja modelira prolaz signala kroz akson biološkog neurona i određuje izlaz neurona $o = f(net)$ [3, 13].

2.2. Arhitektura umjetnih neuronskih mreža

Kako bi se mogla obavljati složenija obrada podataka, više se umjetnih neurona međusobno povezuje u strukture čime tvore umjetne neuronske mreže. Oblik, odnosno arhitektura mreže uvelike određuje njezino ponašanje i sposobnost učenja [14].

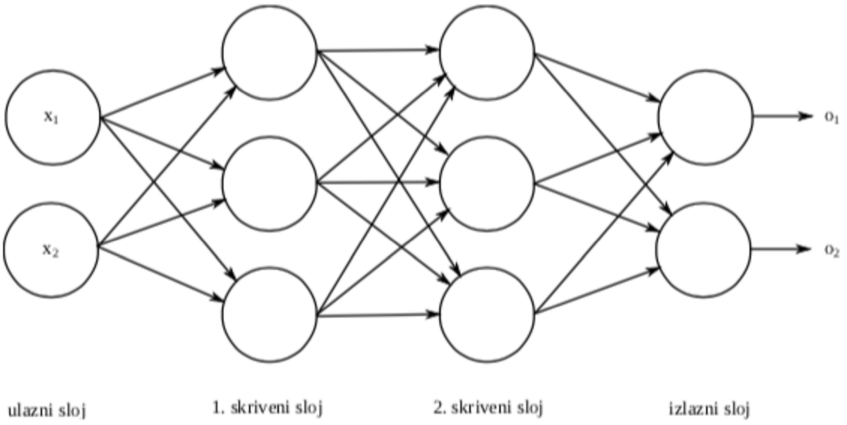
2.2.1. Unaprijedne slojevite umjetne neuronske mreže

Kažemo da je umjetna neuronska mreža unaprijedna (engl. *feedforward*) ako u njoj ne postoje ciklusi u obradi podataka.

Ako je mreža ujedno i slojevita (engl. *multi-layered*), njezini su neuroni grupirani u jasno određene slojeve, pri čemu se izlazi neurona u k-tom sloju računaju na temelju informacija primljenih isključivo od neurona u sloju $k-1$. Sloj koji sadrži ulazne neurone poznat je kao ulazni sloj (engl. *input layer*), izlazni neuroni čine izlazni sloj (engl. *output layer*), a slojevi između ulaznog i izlaznog nazivaju se skrivenim slojevima (engl. *hidden layers*) [14, 13].

Potpuno povezana umjetna neuronska mreža

Potpuno povezana (engl. *fully-connected*) unaprijedna slojevita mreža sastoji se od potpuno povezanih slojeva. Svaki od neurona u k-tom potpuno povezanom sloju na ulaz dobiva pobude od svih neurona iz sloja $k-1$ [13].



Slika 2.2: Potpuno povezana mreža dimenzija 2x3x3x4 [13].

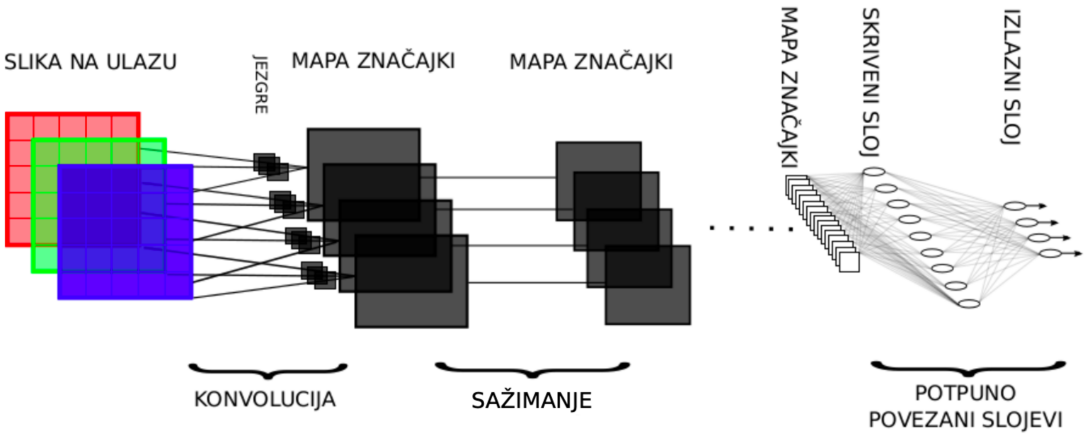
Konvolucijska umjetna neuronska mreža

Konvolucijske mreže specijaliziran su oblik unaprijednih slojevitih mreža za obradu podataka rešetkaste (engl. *grid-like*) topologije poput vremenskih sljedova (1-D re-

šetka podataka uzorkovanih u pravilnim vremenskim intervalima) ili slika (2-D rešetka piksela) [7, 6].

Prednost konvolucijskih mreža u odnosu na potpuno povezane jest u manjoj međusobnoj povezanosti neurona (engl. *sparse connectivity*) i dijeljenju težina među neuronima (engl. *parameter sharing*). Time se smanjuje potreba za učenjem i pohranom velike količine parametara te broj operacija potreban za izračun izlaza [7, 6].

Klasičnu strukturu konvolucijske mreže čine tri tipa slojeva: konvolucijski slojevi, slojevi sažimanja te potpuno povezani slojevi.



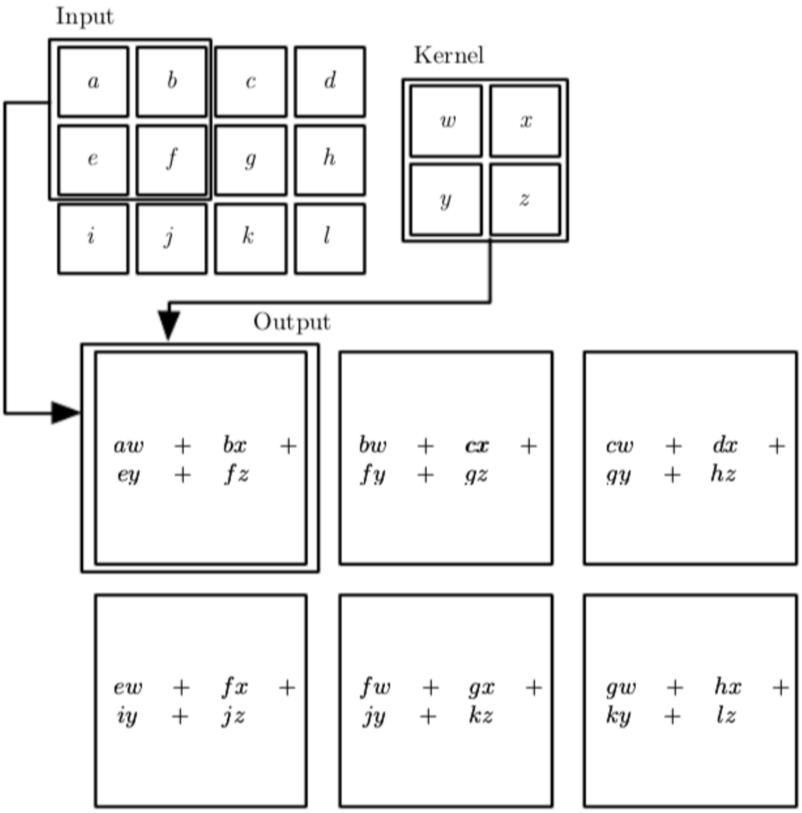
Slika 2.3: Struktura konvolucijske mreže [7].

Konvolucijski slojevi Konvolucijski slojevi naziv su dobili po matematičkoj operaciji konvolucije

$$h(t) = x(t) * w(t),$$

gdje, u terminologiji konvolucijskih mreža, prvi argument $x(t)$ predstavlja ulaz (engl. *input*), drugi argument $w(t)$ čine slobodni parametri poznati kao jezgra (engl. *kernel*), a rezultat operacije $h(t)$ je mapa značajki (engl. *feature map*) [6].

U primjeru 2-D konvolucije sa slike 2.4 ulaz dimenzija 3x4 konvoluira s 2x2 jezgrom i dalje mapu značajki dimenzija 2x3. Jezgra "klizi" preko 2-D ulaznih podataka, izvodeći množenje po elementima s dijelom ulaza nad kojim se u tom trenutku nađe. Dobiveni umnošci zbrajaju se u jedan element izlazne mape značajki. Ta se izlazna vrijednost može smatrati težinskom sumom lokalnog susjedstva elemenata ulaza, gdje elementi jezgre predstavljaju težine. Postupak se ponavlja globalno za sve skupine ulaznih elemenata (tj. za sva susjedstva) nad kojima se jezgra nađe, konačno rezultirajući 2x3 mapom značajki [7, 9].



Slika 2.4: Primjer 2-D konvolucije [6].

Slojevi sažimanja Slojevi sažimanja koriste funkciju sažimanja (engl. *pooling function*) kako bi mapirali skup prostorno bliskih značajki na ulazu u jednu značajku na izlazu [7]. Primjerice, sažimanjem maksimalnom vrijednošću (engl. *max pooling*) pravokutno se susjedstvo ulaznih značajki svodi na izlaznu značajku koja predstavlja maksimalnu vrijednost tog susjedstva [6].

2.3. Učenje umjetnih neuronskih mreža

Na ulaze umjetne neuronske mreže iterativno se dovode podaci kojima se mreža postupno prilagođava promjenom težina veza između neurona i tako uči [3].

2.3.1. Vrste učenja

Pojedinačno i grupno učenje

Težine mreže mogu se ažurirati nakon svakog predočenog uzorka (pojedinačno učenje, engl. *on-line learning*) ili tek nakon što prođe jedna epoha tj. nakon predočavanja svih

uzoraka iz danog skupa uzoraka (grupno učenje, engl. *batch learning*) [13].

Nadzirano i nenadzirano učenje

Ako mreža uči s učiteljem (nadzirano učenje, engl. *supervised learning*) tada joj se dovode uzorci u obliku parova ulaznih podataka i željenih izlaza. Pri tome se definira funkcija gubitka (engl. *loss function*) koja mjeri kvalitetu mreže, primjerice kao polovičnu sumu srednjih kvadratnih odstupanja dobivenih izlaza od željenih, i koja se nastoji minimizirati. Kod učenja bez učitelja (engl. *unsupervised learning*) neuronskoj se mreži na ulaz dovode uzorci sastavljeni samo od ulaznih podataka [13, 3].

2.3.2. Podjela podataka na skupove

Osim skupa podataka za učenje (engl. *training set*) koji se koristi za ažuriranje težina mreže, obično se raspolaže i skupom za provjeru (engl. *validation set*) te skupom za testiranje (engl. *testing set*). Primjenom skupa za provjeru kontrolira se rad mreže da se previše ne prilagodi skupu za učenje i počne odstupati od općih trendova u podacima (pretreniranost, engl. *overfitting*). Konačna kvaliteta rada mreže provjerava se skupom za testiranje [13].

2.3.3. Algoritmi učenja

Postoje brojni algoritmi učenja umjetnih neuronskih mreža, najpoznatiji od kojih je algoritam propagacije pogreške unatrag (engl. *backpropagation*) namijenjen unaprijednim mrežama. Primjenom tog algoritma iznos korekcije za svaku od težina dobiva se izračunom gradijenata funkcije gubitka [3, 14].

2.4. Umjetne neuronske mreže kao klasifikacijski modeli

Klasifikacija je postupak određivanja razreda (klase, labele) kojoj određeni uzorak pripada na temelju svojih značajki. Klasifikacijskim problemom možemo smatrati, primjerice, razvrstavanje skupa slika mačaka, pasa i papiga u tri razreda (mačka, pas, papiga) ovisno o sadržaju slike [3, 13].

Umjetne neuronske mreže možemo koristiti kao klasifikacijske modele primjenom nadziranog učenja. Skup uzoraka za učenje čine podaci koje želimo klasificirati upareni s odgovarajućim labelama. Mreža na izlazu predviđa kojem razredu pripada dani

ulazni uzorak. Cilj je dovoljno dobro naučiti mrežu da ona može samostalno klasificirati nove uzorke.

Jednojedinično kodiranje Ako je riječ o klasifikacijskom problemu s više razreda gdje svaki uzorak pripada samo jednom od njih, obično se koristi jednojedinično kodiranje (engl. *one-hot encoding*) za prikaz labela. Duljina kodne riječi jednaka je broju različitih labela. Riječ sadrži jednu jedinicu, na onoj poziciji koja predstavlja trenutno kodiranu labelu, dok je ostatak riječi popunjen nulama [3].

3. Neprijateljski napadi

Neprijateljski napadi (engl. *adversarial attacks*) nastoje namjerno oblikovanim ulazim podacima zavarati modele strojnog učenja. U kontekstu klasifikacijskih modela, napadač želi natjerati model da pogrešno klasificira dani ulazni podatak neovisno o kojoj je točno pogrešnoj labeli riječ (tzv. neusmjereni napad, engl. *untargeted attack*) ili pak da ga klasificira nekom unaprijed određenom pogrešnom labelom (usmjereni napad, engl. *targeted attack*) [2].

Ovisno o količini informacija koju napadač posjeduje o modelu, razlikujemo dvije vrste napada: napade bijele kutije (engl. *white-box attacks*) i napade crne kutije (engl. *black-box attacks*).

3.1. Napadi bijele kutije

Kod *white-box* napada, napadač je u potpunosti upoznat s arhitekturom modela i njegovim parametrima. U takvom scenariju moguće je primijeniti *backpropagation*, što vodi do učinkovitih napada temeljenih na izračunu gradijenata [2].

3.2. Napadi crne kutije

U stvarnom svijetu, napadač najčešće nema uvid u arhitekturu i parametre mreže, već mu model koji napada predstavlja crnu kutiju. Tada govorimo o *black-box* napadima, koji na raspolaganju imaju jedino informacije o ulazno-izlaznim parovima modela [2].

Jedan pristup ovoj vrsti napada uključuje treniranje zamjenske mreže koja se napada *white-box* napadima. Pritom generirani neprijateljski uzorci iskoristili bi se za napad na mrežu nepoznate arhitekture. Ova se metoda oslanja na svojstvo prenosivosti (engl. *transferability*) koje tvrdi da neprijateljske uzorke koje generira jedan model, drugi model često pogrešno klasificira [2].

Drugi se pristupi, kao što je *ZOO* (optimizacija nultog reda, engl. *zeroth order optimization*), oslanjaju na procjenu gradijenata [2].

Od metoda koje se uopće ne bave gradijentima ističe se generiranje neprijateljskih uzoraka genetskim algoritmima, gdje se iterativno evoluira populacija potencijalnih uzoraka primjenom genetskih operatora selekcije, mutacije i križanja dok kriteriji zaustavljanja algoritma ne budu zadovoljeni [2].

3.2.1. Model *black-box* napada

Napadač ne posjeduje znanje o arhitekturi, parametrima, skupu za učenje, niti bilo kakvim međurezultatima dobivenim u mreži koju napada. Mreža za njega predstavlja crnu kutiju oblika

$$g : \mathbb{R}^n \rightarrow [0,1]^{|L|}$$

gdje je n dimenzija uzorka i $|L|$ broj labela (klasa) u skupu labela L. Funkcija g preslikava uzorak u vektor vjerojatnosti njegove klasifikacije svakom od $|L|$ labela. Vjerojatnost klasifikacije uzorka $\mathbf{x}$ labelom $l \in L$ označit ćemo s $[g(\mathbf{x})]_l$.

Model neusmjerenog napada

Ako je riječ o neusmjerenom napadu, napadač želi na temelju danog uzorka $\mathbf{x}$ pronaći izmijenjeni uzorak $\mathbf{x_{adv}}$ tako da je vjerojatnost klasifikacije ispravnom labelom $l_{orig} \in L$ minimalna:

$$\min(g(\mathbf{x_{adv}})) = [g(\mathbf{x_{adv}})]_{l_{orig}},$$

tj. tako da najmanja vjerojatnost u $|L|$-dimenzionalnom vektoru vjerojatnosti $g(\mathbf{x_{adv}})$ pripada upravo ispravnoj labeli l_{orig}.

Model usmjerenog napada

Kod usmjerenog napada, cilj je izmijenjeni uzorak $\mathbf{x_{adv}}$ klasificirati točno određenom pogrešnom labelom $l_{adv} \in L$ s maksimalnom vjerojatnošću u vektoru vjerojatnosti:

$$\max(g(\mathbf{x_{adv}})) = [g(\mathbf{x_{adv}})]_{l_{adv}}.$$

3.3. Robusnost napada

Postojanje raznih metoda obrane od neprijateljskih napada, poput treniranja mreže na skupu za učenje proširenom neprijateljskim uzorcima ili manipuliranja gradijentima tako da budu neiskoristivi napadaču, dovodi u upit njihovu učinkovitost [2].

Postupci kojima se nastoji povećati otpornost neprijateljskih napada na pokušaje suprotstavljanja uključuju: ograničavanje broja izmjena napravljenih na originalnom uzorku (npr. promjena samo jednog piksela kod napada na modele za klasifikaciju slika [10]) i postepeno generiranje neprijateljskih uzoraka uz paralelne pokušaje uklanjanja izmjena nanesenih na originalni uzorak (npr. iterativno pojačavanje količine šuma na CAPTCHA slici sve dok je model uspijeva ispravno klasificirati nakon filtriranja šuma [8]).

4. Višekriterijska optimizacija

Za razliku od problema jednokriterijske optimizacije, gdje se traži rješenje koje je bolje po nekom jedinstvenom kriteriju f (tzv. funkcija cilja, engl. *objective function*) od svih ostalih rješenja u prostoru prihvatljivih rješenja, u višekriterijskoj optimizaciji cilj je pronaći najbolja rješenja na temelju M različitih kriterija.

4.1. Problem višekriterijske optimizacije

Općenito, problem višekriterijske optimizacije možemo definirati kao

$$\begin{aligned} \text{Minimiziraj / maksimiziraj} \quad & f_m(\mathbf{x}), && m = 1, 2, ..., M; \\ \text{uz zadovoljenje uvjeta} \quad & x_i^L \leq x_i \leq x_i^U, && i = 1, 2, ..., n; \\ & g_j(\mathbf{x}) \geq 0, && j = 1, 2, ..., J; \\ & h_k(\mathbf{x}) = 0, && k = 1, 2, ..., K. \end{aligned}$$

Rješenje $\mathbf{x} \in \mathbb{R}^n$ vektor je od n decizijskih varijabli $\mathbf{x} = (x_1, x_2, ..., x_n)^T$. Rješenja koja zadovoljavaju ograničenja problema čine prostor prihvatljivih rješenja S unutar prostora decizijskih varijabli $\mathbb{R}^n$ (engl. *decision variable space*). [12, 4].

S obzirom na to da rješenja želimo optimirati po M kriterija, u višekriterijskoj optimizaciji razmatramo i M-dimenzionalan prostor ciljnih funkcija $Z \subset \mathbb{R}^M$ (engl. *objective space*). Za svako rješenje $\mathbf{x}$ u prostoru decizijskih varijabli definirano je preslikavanje $\mathbf{f}(\mathbf{x}) = \mathbf{z} = (f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_m(\mathbf{x}))^T$ u prostor ciljnih funkcija [12, 4].

4.2. Relacija dominacije

Zbog višedimenzionalnosti prostora ciljnih funkcija, postavlja se pitanje kako odrediti koje je od dvaju danih rješenja $\mathbf{x}^{(1)}$ i $\mathbf{x}^{(2)}$ bolje, ako se oni preslikavaju u vektore ciljnih funkcija $\mathbf{z}^{(1)}$ i $\mathbf{z}^{(2)}$ koji općenito nisu međusobno usporedivi. Uvodi se pojam dominacije nad rješenjima na sljedeći način:

Dominacija Neka rješenje $\mathbf{x}^{(1)}$ ima pridružen vektor ciljnih funkcija $\mathbf{z}^{(1)}$, a rješenje $\mathbf{x}^{(2)}$ vektor $\mathbf{z}^{(2)}$, pri čemu je $\mathbf{z}^{(i)} = \mathbf{f}(\mathbf{x}^{(i)})$. Kažemo da rješenje $\mathbf{x}^{(1)}$ dominira nad rješenjem $\mathbf{x}^{(2)}$ i pišemo $\mathbf{x}^{(1)} \succ \mathbf{x}^{(2)}$ ako vrijedi

1. rješenje $\mathbf{x}^{(1)}$ je u svim komponentama vektora ciljnih funkcija bolje ili jednako dobro kao rješenje $\mathbf{x}^{(2)}$ tj. $\forall j \in \{1, ..., M\}\ z_j^{(1)} \geq z_j^{(2)}$,

2. rješenje $\mathbf{x}^{(1)}$ je u barem jednoj komponenti vektora ciljnih funkcija strogo bolje od rješenja $\mathbf{x}^{(2)}$ tj. $\exists j \in \{1, ..., M\}\ z_j^{(1)} > z_j^{(2)}$.

Ova relacija može se iskoristiti za procjenu kvalitete rješenja u optimizacijskim algoritmima. Za svako rješenje prati se koliko postoji rješenja koja ga dominiraju. Što je taj broj manji, rješenje je kvalitetnije, a najbolje je rješenje ono nad kojim nitko ne dominira [12].

Nedominirani skup Ako skup rješenja označimo sa P, nedominirani skup P' podskup je skupa P i sastoji se od onih rješenja nad kojima niti jedno rješenje iz P ne dominira.

4.3. Pareto optimalnost

Zadaća optimizacijskih algoritama bit će, uz pomoć relacije dominacije, pronaći nedominirani skup (tj. njegovu što bolju aproksimaciju) nad prostorom prihvatljivih rješenja za dani problem višekriterijske optimizacije. Uvodimo sljedeća dva pojma:

Pareto-optimalan skup Nedominirani skup skupa svih prihvatljivih rješenja jest Pareto-optimalan skup.

Pareto fronta Za dani Pareto-optimalan skup, skup pripadnih vektora ciljnih funkcija naziva se Pareto fronta.

4.4. Algoritmi višekriterijske optimizacije

4.4.1. NSGA-II

NSGA-II poboljšana je inačica algoritma NSGA (engl. *Non-dominated Sorting Genetic Algorithm*) koja izravnom primjenom relacije dominacije kroz postupak tzv. nedo-

miniranog sortiranja (engl. *non-dominated sort*) nastoji riješiti probleme višekriterijske optimizacije [12].

Osim promjena u korištenim operatorima i povećanju efikasnosti, prednost ove verzije algoritma u odnosu na osnovnu jest u tome što koristi koncept elitizma (engl. *elitism*) kako bi se izbjegao gubitak jednom pronađenih dobrih rješenja tijekom evolucije [5].

Nedominirano sortiranje

Nedominiranim sortiranjem nastojimo razvrstati populaciju rješenja u podskupove (tzv. fronte, engl. *fronts*) prema tome koliko su rješenja "blizu" nedominiranom skupu [12].

Postupak se provodi u dva dijela. Prvo, za svako rješenje p u populaciji P potrebno je odrediti broj rješenja koja nad njim dominiraju n_p (engl. *domination count*)

$$n_p = |\{q : q \in P \wedge q \succ p\}|$$

i skup S_p rješenja nad kojim p dominira

$$S_p = \{q : q \in P \wedge p \succ q\}.$$

Moguće je zatim prijeći na određivanje fronti. Rješenja p za koja je u prethodnom koraku utvrđeno da vrijedi $n_p = 0$ čine prvu frontu. Obilazimo skupove S_p tih rješenja i za svaki $q \in S_p$ umanjimo njihov n_q za jedan. Ako pri tome n_q postane jednak nuli, odvojimo to rješenje u zasebni skup Q koji predstavlja drugu frontu. Obilazak se nastavlja po skupovima S_q rješenja u Q itd. dok se ne odrede sve fronte [5].

Očuvanje raznolikosti rješenja

Kako bi se očuvala raznolikost rješenja u Pareto-optimalnom skupu, NSGA-II uvodi sortiranje prema udaljenosti grupiranja (engl. *crowding distance sort*) koje osigurava dobru raspršenost rješenja unutar fronti.

Udaljenost grupiranja Za svako rješenje p u danoj fronti i svaku ciljnu funkciju f_m izračuna se suma normiranih udaljenosti do najbližeg rješenja koje ima manji odnosno veći iznos te funkcije cilja. Dobivena vrijednost poznata je kao udaljenost grupiranja d_p (engl. *crowding distance*) danog rješenja p i omogućuje procjenu "gustoće" rješenja u njegovoj okolini. Što je vrijednost d_p niža, to je rješenje više okruženo ("grupirano") drugim rješenjima.

Operator usporedbe $\prec_n$ Nakon određivanja udaljenosti grupiranja, rješenja u frontama mogu se međusobno uspoređivati operatorom usporedbe $\prec_n$ (engl. *crowded-comparison operator*) na sljedeći način:

$$p \prec_n q \quad \textbf{ako} \quad (p_{rank} < q_{rank}) \ \textbf{ili} \ ((p_{rank} = q_{rank}) \ \textbf{i} \ (d_p > d_q)).$$

Drugim riječima, od dvaju rješenja p i q preferira se ono koje je u fronti što bližoj prvoj fronti, pri čemu rank (engl. *rank*) predstavlja broj fronte. Ako su rješenja u istoj fronti, bolje je ono rješenje koje se nalazi u području manje gustoće (tj. ono koje je okruženo s manje drugih rješenja pa ima veću udaljenost grupiranja) [5].

Algoritam

Algoritam započinje generiranjem populacije nasumičnih rješenja P veličine N. Na tu se početnu populaciju primijeni nedominirano sortiranje, pri čemu se svakom rješenju dodijeli rank i udaljenost grupiranja.

U i-toj iteraciji algoritma, trenutna se populacija P koristi za stvaranje N novih rješenja Q primjenom genetskih operatora selekcije, križanja i mutacije. Koristi se grupirajuća turnirska selekcija, inačica binarne turnirske selekcije koja koristi operator usporedbe $\prec_n$ za odabir boljeg od nasumično izvučena dva rješenja.

Slijedi nedominirano sortiranje (uz dodjelu ranka i udaljenosti grupiranja) unije $P \cup Q$ trenutne populacije (elitizam) i novostvorenih rješenja. Na temelju dobivenih fronti, stvara se sljedeća populacija P_{nova} veličine N tako da se u nju redom dodaju sva rješenja iz prve fronte, zatim druge, itd. do popunjenja populacije ili pronalaska fronte koja ne može u cijelosti stati u P_{nova}.

Ako je veličina populacije P_{nova} i dalje manja od N nakon prethodnog koraka, fronta koja nije uspjela biti u cijelosti dodana sortira se uz pomoć operatora usporedbe $\prec_n$ te se iz nje izabire prvih $N - |P_{nova}|$ rješenja kojima se upotpunjuje sljedeća populacija.

Postupak se ponavlja dok se ne zadovolje uvjeti zaustavljanja algoritma. Rezultat algoritma jest prva fronta koja, u idealnom slučaju, predstavlja Pareto-optimalan skup za zadani problem višekriterijske optimizacije [5].

Algoritam 1 NSGA-II

Ulaz: N (veličina populacije)

Rezultat: $fronte[0]$ (nedominirani skup)

$P \leftarrow$ INICIJALIZIRAJ(N)
NEDOMINIRANO_SORTIRAJ(P) ▷ uz dodjelu ranka i udaljenosti grupiranja

dok $nisu\ zadovoljeni\ uvjeti\ zaustavljanja$ **čini**
    $Q \leftarrow$ STVORI_DJECU(P)

    $fronte \leftarrow$ NEDOMINIRANO_SORTIRAJ$(P \cup Q)$

    $P_{nova} \leftarrow \emptyset$
    za sve $fronta \in fronte$ **čini**
        ako $|P_{nova}| + |fronta| > N$ **onda**
            SORTIRAJ_SILAZNO$(fronta, \prec_n)$
            $P_{nova} \leftarrow P_{nova} \cup fronta[0 : (N - |P_{nova}|)]$
            prekini za
        kraj ako

        $P_{nova} \leftarrow P_{nova} \cup fronta$
    kraj for

    $P \leftarrow P_{nova}$
kraj dok

4.4.2. SPEA2

SPEA2 unaprjeđenje je algoritma SPEA (engl. *Strength Pareto Evolutionary Algorithm*) koje se temelji na vrednovanju kvalitete rješenja uz relaciju dominacije, procjeni gustoće okoline rješenja pomoću njegova k-tog najbližeg susjeda te održavanju arhive (engl. *archive*) nedominiranih rješenja [11].

Dodjela dobrote

Svakom se rješenju p u populaciji P i arhivi $\overline{P}$ dodijeli snaga $S(p)$ (engl. *strength value*) koja je jednaka broju rješenja nad kojima p dominira:

$$S(p) = |\{q : q \in (P \cup \overline{P}) \wedge p \succ q\}|.$$

Na temelju snage računa se sirova vrijednost dobrote $R(p)$ rješenja p (engl. *raw fitness*) kao suma snaga onih rješenja u $P \cup \overline{P}$ koja dominiraju nad p:

$$R(p) = \sum_{q \in (P \cup \overline{P}) \wedge q \succ p} S(q).$$

Procjena gustoće $D(p)$ okoline rješenja p svodi se na pronalazak inverzne vrijednosti udaljenosti p od njegovog k-tog najbližeg susjeda. Prvo je potrebno u prostoru ciljnih funkcija odrediti udaljenosti svakog rješenja p do svih ostalih rješenja u $P \cup \overline{P}$ te spremiti dobivene vrijednosti u listu. Nakon sortiranja liste, njezin k-ti element upravo je tražena udaljenost σ_p^k do k-tog najbližeg susjeda, pri čemu je $k = \sqrt{|P| + |\overline{P}|}$. Gustoću dobivamo sljedećim izrazom:

$$D(p) = \frac{1}{\sigma_p^k + 2}.$$

Konačna dobrota jedinke $F(p)$ jednaka je zbroju sirove dobrote $R(p)$ i izračunate gustoće $D(p)$:

$$F(p) = R(p) + D(p).$$

Rješenje p je nedominirano ako za njega vrijedi $R(p) = 0$ odnosno $F(p) < 1$ [11].

Održavanje arhive nedominiranih rješenja

Arhiva nedominiranih rješenja $\overline{P}$ konstantne je veličine $\overline{N} = |\overline{P}|$, a održava se postupkom okolišne selekcije (engl. *environmental selection*).

Okolišna selekcija Arhiva za sljedeću iteraciju algoritma $\overline{P}_{nova}$ gradi se tako da se u nju kopiraju sva nedominirana rješenja iz trenutne populacije P i arhive $\overline{P}$.

$$\overline{P}_{nova} = \{p : p \in (P \cup \overline{P}) \wedge F(p) < 1\}$$

Ako je kopirano točno $\overline{N}$ rješenja ($|\overline{P}_{nova}| = \overline{N}$), nova arhiva je popunjena i postupak završava.

Dopuna arhive Ako u arhivi nema dovoljno rješenja ($|\overline{P}_{nova}| < \overline{N}$), u nju se dodaje još $\overline{N} - |\overline{P}_{nova}|$ rješenja p s $F(p) \geq 1$ dobivenih sortiranjem $P \cup \overline{P}$ silazno po dobroti.

Skraćivanje arhive Ako u arhivi postoji previše rješenja ($|\overline{P}_{nova}| > \overline{N}$), potrebno je napraviti skraćivanje (engl. *archive truncation*). Iterativno se izbacuju rješenja iz $\overline{P}_{nova}$ dok se ne dosegne veličina $\overline{N}$. Za izbacivanje se odabire ono rješenje p za koje vrijedi $p \leq_d q, \forall q \in \overline{P}_{nova}$, pri čemu

$$\begin{aligned} p \leq_d q \qquad & \textbf{ako} \quad \forall\, 0 < k < |\overline{P}_{nova}| : o_p^k = o_q^k \\ & \textbf{ili} \quad \exists\, 0 < k < |\overline{P}_{nova}| : ((\forall\, 0 < l < k : o_p^l = o_q^l)\ \textbf{i}\ o_p^k < o_q^k). \end{aligned}$$

Drugim riječima, u svakoj iteraciji skraćivanja izbacuje se ono rješenje koje ima najmanju udaljenost do nekog drugog rješenja. Ako postoji više takvih rješenja, razmatraju se druge najmanje udaljenosti itd. [11].

Algoritam

Algoritam započinje generiranjem populacije nasumičnih rješenja P veličine N te stvaranjem prazne arhive $\overline{P} = \emptyset$.

U i-tom koraku algoritma dodjeljuje se dobrota rješenjima u populaciji i arhivi $P \cup \overline{P}$. Postupkom okolišne selekcije (uz dopunu i skraćivanje, po potrebi) stvara se nova arhiva $\overline{P}_{nova}$. Sljedeća populacija dobiva se iz rješenja u novostvorenoj arhivi $\overline{P}_{nova}$ primjenom genetskih operatora binarne turnirske selekcije te križanja i mutacije.

Postupak se ponavlja dok se ne zadovolje uvjeti zaustavljanja algoritma. Rezultat algoritma je nedominirani skup rješenja za dani problem višekriterijske optimizacije [11].

Algoritam 2 SPEA2

Ulaz: N (veličina populacije), $\overline{N}$ (veličina arhive)

Rezultat: $\{p : p \in \overline{P} \wedge F(p) < 1\}$ (nedominirani skup)

$P \leftarrow$ INICIJALIZIRAJ(N)
$\overline{P} \leftarrow \emptyset$

dok $nisu\ zadovoljeni\ uvjeti\ zaustavljanja$ **čini**
    DODIJELI_DOBROTU$(P \cup \overline{P})$

    $\overline{P}_{nova} = \{p : p \in (P \cup \overline{P}) \wedge F(p) < 1\}$
    ako $|\overline{P}_{nova}| < \overline{N}$ **onda**
        DOPUNI_ARHIVU$(\overline{P}_{nova})$
    inače ako $|\overline{P}_{nova}| > \overline{N}$ **onda**
        SKRATI_ARHIVU$(\overline{P}_{nova})$
    kraj ako

    $P \leftarrow$ STVORI_DJECU$(\overline{P}_{nova})$
    $\overline{P} \leftarrow \overline{P}_{nova}$
kraj dok

5. Programsko ostvarenje

Rad je ostvaren u programskom jeziku Python[1], uz uporabu sustava Git[2] za praćenje promjena u izvornom kodu.

Biblioteka NumPy[3] korištena je jer nudi objekte za modeliranje višedimenzionalnih matrica uz bogat skup efikasnih matematičkih operacija.

Klasifikacijski modeli za neprijateljske napade izgrađeni su pomoću*open-source* biblioteke Keras[4], koja predstavlja API visoke razine za biblioteku TensorFlow 2.0[5].

Crtanje rezultata učenja klasifikacijskih modela te višekriterijske optimizacije ostvareno je uz biblioteku Matplotlib[6]. Prikaz uzoraka kao slika omogućila je biblioteka PIL (Pillow)[7]. Rezultati su obrađeni uz pomoć biblioteke Pandas[8].

5.1. Izgradnja klasifikacijskih modela

Klasifikacijski modeli korišteni u neprijateljskim napadima izvedeni su iz apstraktnog razreda `AttackModel` koji nudi metode za treniranje modela, predviđanje labele danog uzorka te pohranu težina i njihovo učitavanje iz datoteke.

5.1.1. Potpuno povezani model

Potpuno povezani model predstavljen je podrazredom `SimpleModel`. Izgrađen je na temelju `Sequential` modela biblioteke Keras dodavanjem niza potpuno povezanih slojeva (`Dense` podrazred razreda `Layer` u Kerasu) i pripadnih aktivacijskih funkcija (podrazred `Activation`).

[1] https://www.python.org/
[2] https://git-scm.com/
[3] https://numpy.org/
[4] https://keras.io/
[5] https://www.tensorflow.org/
[6] https://matplotlib.org/
[7] https://pillow.readthedocs.io/en/stable/
[8] https://pandas.pydata.org/

Default parametri razreda `SimpleModel` grade neuronsku mrežu s jednim skrivenim slojem od 128 neurona te izlaznim slojem od 10 neurona. Aktivacijska funkcija skrivenog sloja jest zglobnica (engl. *Rectified Linear Unit*, *ReLU*), a za izlazni sloj koristi se funkcija *softmax* koja izlaz pretvara u vektor kategorijskih vjerojatnosti.

5.1.2. Konvolucijski model

Podrazred `ConvolutionalModel` predstavlja konvolucijsku mrežu koja se također temelji na Kerasovom `Sequential` modelu.

Model sadrži dva skupa od po dva konvolucijska sloja (`Conv2D`) i jednog *max pooling* sloja (`MaxPooling2D`). Sloj `Flatten` koristi se za transformiranje višedimenzijskog izlaza tih skupova na jednodimenzionalni vektor koji `Dense` slojevi zahtijevaju na svom ulazu. Između `Dense` slojeva dodan je `Dropout` sloj kojim se nastoji smanjiti vjerojatnost pretreniranja modela zanemarivanjem utjecaja nasumično odabranih neurona tijekom svake epohe.

Prema *default* parametrima, gradi se mreža s 2 konvolucijska sloja s 32 jezgre veličine 3x3 te 2 konvolucijska sloja s 64 jezgre veličine 3x3. Veličina sloja sažimanja je 2x2, a potpuno povezani slojevi imaju 128 odnosno 10 neurona. Aktivacijske funkcije skrivenih slojeva su zglobnice, a izlazni sloj koristi *softmax* funkciju. Jačina `Dropout` sloja (engl. *dropout rate*) postavljena je na 0.4.

5.2. Skup podataka MNIST

Za treniranje klasifikacijskih modela te generiranje neprijateljskih uzoraka odabran je MNIST skup podataka koji sadrži slike rukom pisanih znamenki 0-9. Podijeljen je na skup uzoraka za učenje od 60.000 slika te skup za testiranje od 10.000 slika.

5.2.1. Struktura podataka

Izvorne crno-bijele slike znamenki u MNIST-u su normalizirane na 20x20 piksela te sadrže i razine sive kao rezultat tehnike *anti-aliasing*a provedene pri normalizaciji. Znamenke su centrirane unutar 28x28 okvira izračunavanjem središta mase piksela. Pikseli su pohranjeni kao brojčane vrijednosti 0-255 (0 = crno, 255 = bijelo), a labele kao vrijednosti 0-9 ovisno o znamenci koju opisuju [1].

5.2.2. Prilagodba podataka

Podaci su dohvaćeni u već vektoriziranom, NumPy obliku pomoću Kerasova modula `tf.keras.datasets`. Implementirana je funkcija `load_mnist` za preoblikovanje podataka na dimenzije koje zahtijeva željeni klasifikacijski model, njihovu normalizaciju na raspon $[0, 1]$ te pretvorbu labela u vektore kodirane jednojediničnim kodom.

5.3. Modeliranje problema višekriterijske optimizacije

Apstraktan razred `Problem` predstavlja općeniti problem višekriterijske optimizacije. Sadrži atribute za pohranu zahtijevane veličine rješenja (dimenzije vektora u prostoru decizijskih varijabli), veličine vektora ciljnih funkcija (broj kriterija) te intervalnih ograničenja na vrijednosti rješenja i vektora ciljnih funkcija. Razred nudi apstraktnu metodu `evaluate` za procjenu kvalitete populacije.

5.3.1. Neprijateljski napadi kao problem višekriterijske optimizacije

Neprijateljski napad na klasifikacijski model možemo formulirati kao problem višekriterijske optimizacije na sljedeći način: originalni uzorak $\mathbf{x}$ želimo algoritmom višekriterijske optimizacije izmijeniti u neprijateljski uzorak $\mathbf{x_{adv}}$ tako da model uspijemo natjerati na pogrešnu klasifikaciju.

Konkretno, vektoriziranoj slici $\mathbf{x}$ iz MNIST skupa podataka pridodat ćemo šum $\mathbf{e}$ (engl. *noise*) dobiven evolucijskim postupkom i dobiti neprijateljsku sliku

$$\mathbf{x_{adv}} = \mathbf{x} + \mathbf{e}.$$

Pri tome, zbog normalizacije vektoriziranih slika na $[0, 1]$, potrebno je i vrijednosti vektora $\mathbf{x_{adv}}$ održavati u tom rasponu rezanjem (engl. *clipping*) vrijednosti vektora šuma $\mathbf{e}$ po potrebi.

Zahtijevamo da generirana slika sadrži što manje moguću količinu šuma kako bi se povećala robusnost napada na pokušaje uklanjanja šuma i osigurala prepoznatljivost znamenke s originalne slike ljudskom oku. Uz to, želimo da model sa što većom vjerojatnošću pogrešno klasificira taj neprijateljski uzorak.

Za očekivati je da će se poboljšanjem jednog kriterija pogoršati onaj drugi, odnosno da će model češće pogrešno klasificirati slike s većim šumom i obratno.

Količina šuma Količinu šuma na slici mjerit ćemo funkcijom cilja jednakoj L_2 (Euklidskoj) normi vektora šuma

$$f_1(\mathbf{e}) = ||\mathbf{e}||_2 = \sqrt{e_1^2 + ... + e_n^2}$$

koju nastojimo minimizirati.

Ograničenje količine šuma Ograničit ćemo količinu šuma time što ćemo sve vrijednosti vektora šuma $\mathbf{e}$ održavati unutar zadanog intervala $[-\delta, \delta]$. To ograničenje možemo formalno zapisati uz L_∞ normu na sljedeći način:

$$||\mathbf{e}||_\infty = \max(|e_1|, ..., |e_n|) \leq \delta,\ \delta \in [0, 1].$$

Maksimalna količina šuma pritom iznosi $||\mathbf{e}||_{2,max} = \delta \cdot \sqrt{n}$ i dobije se kad su sve vrijednosti vektora $\mathbf{e}$ jednake upravo δ ili $-\delta$.

Vjerojatnost pogrešne klasifikacije Oslanjajući se na model *black-box* napada opisan u odjeljku 3.2.1, označimo vjerojatnost klasifikacije neprijateljskog uzorka $\mathbf{x_{adv}}$ labelom l iz skupa labela L s $[g(\mathbf{x_{adv}})]_l$. Ovisno o tipu napada, vjerojatnost pogrešne klasifikacije neprijateljskog uzorka formulira se različitim funkcijama cilja.

Neusmjereni napad

Razred `SimpleAttack` implementacija je neusmjerenog napada koji predstavlja problem višekriterijske optimizacije definiran na sljedeći način:

$$\begin{aligned} \text{Minimiziraj} \quad & f_1(\mathbf{e}) = ||\mathbf{e}||_2; & \\ \text{i minimiziraj} \quad & f_2(\mathbf{e}) = [g(\mathbf{x} + \mathbf{e})]_{l_{orig}}, & l_{orig} \in L; \\ \text{uz zadovoljenje uvjeta} \quad & ||\mathbf{e}||_\infty \leq \delta, & \delta \in [0, 1]. \end{aligned}$$

Usmjereni napad

Usmjereni napad predstavljen je razredom `TargetedAttack` i definira sljedeći problem višekriterijske optimizacije:

$$\begin{aligned} \text{Minimiziraj} \quad & f_1(\mathbf{e}) = ||\mathbf{e}||_2; & \\ \text{i maksimiziraj} \quad & f_2(\mathbf{e}) = [g(\mathbf{x} + \mathbf{e})]_{l_{adv}}, & l_{adv} \in L; \\ \text{uz zadovoljenje uvjeta} \quad & ||\mathbf{e}||_\infty \leq \delta, & \delta \in [0, 1]. \end{aligned}$$

"Poboljšani" usmjereni napad

"Poboljšani" usmjereni napad `ImprovedTargetedAttack` uvodi dodatnu funkciju cilja kojom se nastoji minimizirati suma vjerojatnosti svih labela različitih od željene neprijateljske labele $l_{adv} \in L$:

$$\begin{aligned}
&\text{Minimiziraj} && f_1(\mathbf{e}) = ||\mathbf{e}||_2; && \\
&\text{i maksimiziraj} && f_2(\mathbf{e}) = [g(\mathbf{x}+\mathbf{e})]_{l_{adv}}, && l_{adv} \in L; \\
&\text{i minimiziraj} && f_3(\mathbf{e}) = \sum_{l} [g(\mathbf{x}+\mathbf{e})]_l, && l \in L, l \neq l_{adv}; \\
&\text{uz zadovoljenje uvjeta} && ||\mathbf{e}||_\infty \leq \delta, && \delta \in [0,1].
\end{aligned}$$

5.4. Modeliranje rješenja

Općenito rješenje optimizacijskog problema predstavljeno je apstraktnim razredom `Solution`, koji sadrži atribute za pohranu vektora decizijskih varijabli, vektora ciljnih funkcija te reference na `Problem` čije rješenje predstavlja. Relacija dominacije implementirana je kroz metodu `dominates`.

5.4.1. NSGA-II rješenje

Podrazred `NSGA2Solution` ima dodatne atribute za pohranu broja rješenja koja dominiraju nad trenutnim rješenjem, skupa rješenja nad kojima to rješenje dominira, udaljenosti grupiranja te ranga rješenja. Nadgrađen je operator > kako bi se implementirala negacija operatora usporedbe $\prec_n$.

5.4.2. SPEA2 rješenje

Analogno, podrazred `SPEA2Solution` sadrži atribute za pohranu snage, gustoće, sirove dobrote i dobrote rješenja te skupa rješenja koja dominiraju nad tim rješenjem. Nadgrađen je operator > ne bi li se omogućila usporedba instanci ovog razreda po dobroti.

5.5. Genetski operatori

Genetski operatori korišteni u algoritmima višekriterijske optimizacije implementirani su kao funkcije u modulu `moo.operators`.

5.5.1. Selekcija

Rješenja se odabiru postupkom binarne turnirske selekcije. Prvo se sastavi turnir od dvaju nasumično odabranih rješenja iz dane populacije. Konačni rezultat jest najbolje rješenje u turniru, određeno uz pomoć nadgrađenog operatora > u `Solution` podrazredima.

5.5.2. Mutacija

Ostvareni operator mutacije vektoru decizijskih varijabli pridodaje vektor nasumično izabranih brojeva iz uniformne distribucije uz zadanu vjerojatnost mutacije svake varijable postavljenu na $p = 0.02$.

5.5.3. Križanje

Dva roditeljska rješenja križaju se uniformnim križanjem (engl. *uniform crossover*) i daju dva rješenja-potomka. Svaka varijabla u vektoru decizijskih varijabli potomka naslijeđena je ili od prvog ili od drugog roditelja s jednakom vjerojatnošću križanja ($p = 0.5$). Ako je jedan potomak naslijedio varijablu na određenoj poziciji od prvog roditelja, onda će drugi potomak naslijediti tu varijablu od drugog roditelja, i obratno.

5.6. Parametri i pokretanje

Glavni program, ovisno o argumentima komandne linije, učitava klasifikacijski model te pokreće izgradnju neprijateljskih uzoraka pomoću zadanog algoritma višekriterijske optimizacije.

Može se pokrenuti u željenom razvojnom okruženju (npr. PyCharm[9]) ili preko komandne linije, na sljedeći način:

```
$ python3 main.py algorithm attack_type noise_size model
    weights [-h] [--maxiter MAXITER] [--popsize POPSIZE]
```

[9] https://www.jetbrains.com/pycharm/

argument	**dozvoljene vrijednosti**	**opis**
`algorithm`	nsga2, spea2	algoritam za izradu uzoraka
`attack_type`	simple_attack, targeted_attack, improved_attack	tip neprijateljskog napada
`noise_size`	$[0, 1]$	ograničenje količine šuma
`model`	simple_model, convolutional_model	vrsta klasifikacijskog modela
`weights`	niz znakova	putanja do datoteke s težinama

Tablica 5.1: Opis obaveznih argumenata komandne linije.

Opcionalnim argumentima `--maxiter` i `--popsize` može se postaviti maksimalni broj iteracija te veličina populacije, čije su podrazumijevane vrijednosti inače 100 i 100.

Parametri genetskih operatora, poput vjerojatnosti mutacije, donje i gornje granice intervala za izbor brojeva iz uniformne distribucije te vjerojatnosti križanja, zadani su kao *default* vrijednosti parametara pripadnih funkcija u modulu `moo.operators`.

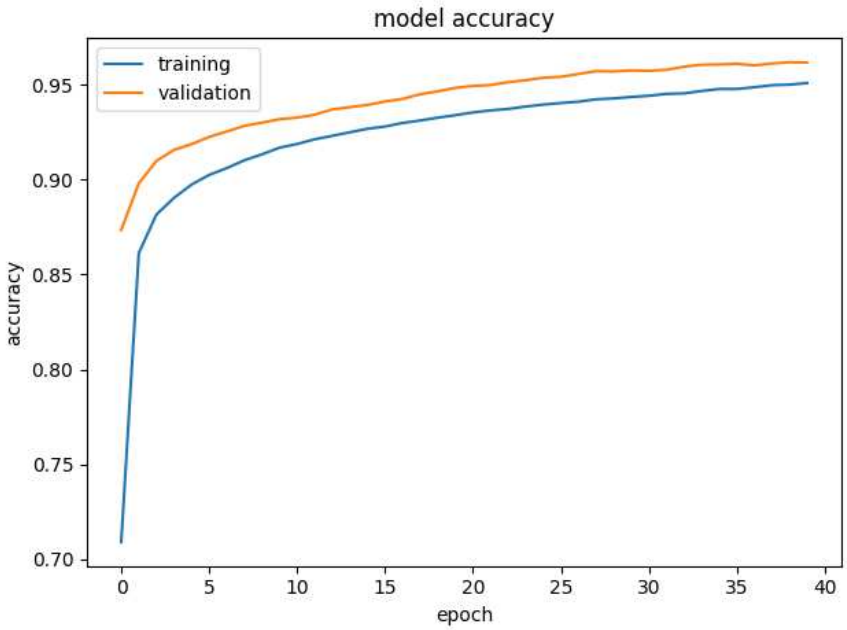
6. Rezultati

6.1. Rezultati učenja klasifikacijskih modela

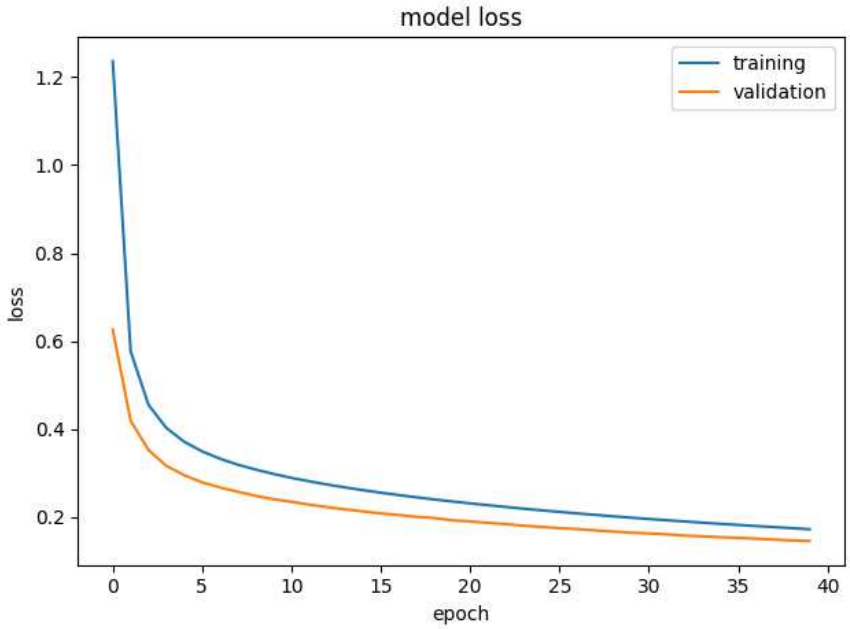
Klasifikacijski modeli trenirani su na MNIST skupu za učenje od 60.000 slika, uz skup za provjeru definiran kao 10% veličine skupa za učenje. Testirani su na skupu za testiranje od 10.000 uzoraka.

6.1.1. Učenje potpuno povezanog modela

Potpuno povezani model treniran je kroz 40 epoha uz veličinu minigrupa (engl. *mini-batches*) od 128. Za funkciju gubitka, koja govori o odstupanju predviđenih labela od stvarnih i koju nastojimo minimizirati, korištena je kategorijska križna entropija (engl. *categorical cross-entropy*). Stohastički gradijentni spust (engl. *stochastic gradient descent*) odabran je kao optimizacijski algoritam za podešavanje težina.



Slika 6.1: Točnost potpuno povezanog modela na skupovima za učenje i provjeru po epohama.



Slika 6.2: Vrijednosti funkcije gubitka potpuno povezanog modela na skupovima za učenje i provjeru po epohama. Smanjivanje vrijednosti kroz epohe podrazumijeva smanjivanje odstupanja između predviđenih i željenih labela.

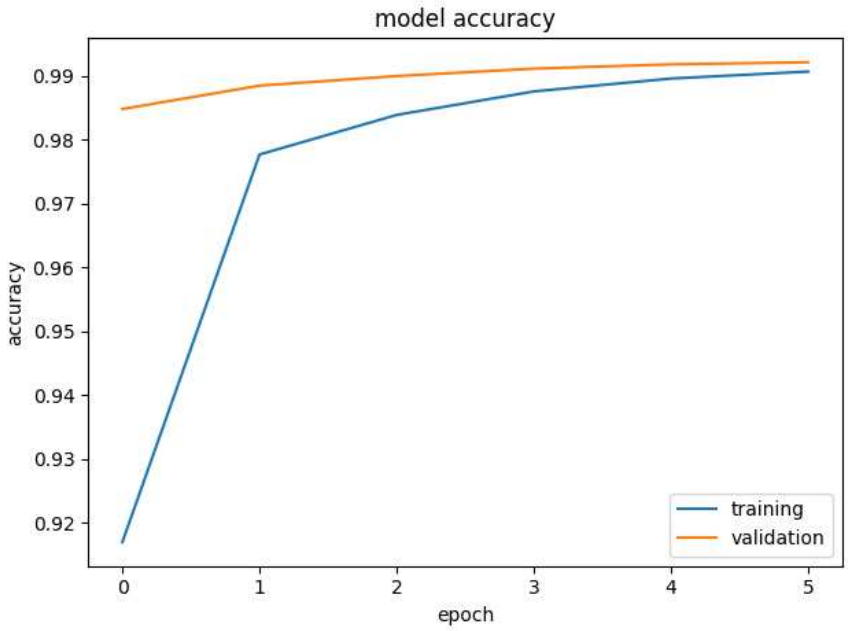
Konačni rezultati učenja i testiranja potpuno povezanog modela prikazani su u sljedećoj tablici.

skup	točnost	funkcija gubitka
skup za učenje	0.9509	0.1719
skup za provjeru	0.9617	0.1452
skup za testiranje	0.9500	0.1720

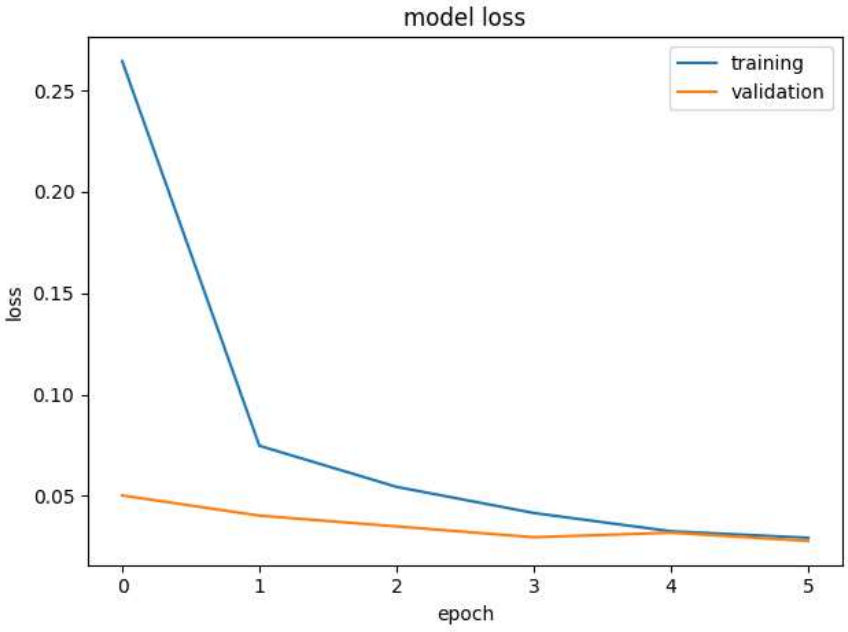
Tablica 6.1: Točnost i konačna vrijednost funkcije gubitka potpuno povezanog modela na različitim skupovima.

6.1.2. Učenje konvolucijskog modela

Učenje konvolucijskog modela trajalo je 6 epoha uz veličinu minigrupa od 128. Kategorijska križna entropija ponovno je korištena kao funkcija gubitka, dok je optimizacijski algoritam u ovom slučaju bio Adam.



Slika 6.3: Točnost konvolucijskog modela na skupovima za učenje i provjeru po epohama.



Slika 6.4: Vrijednosti funkcije gubitka konvolucijskog modela na skupovima za učenje i provjeru po epohama.

skup	točnost	funkcija gubitka
skup za učenje	0.9907	0.0292
skup za provjeru	0.9922	0.0277
skup za testiranje	0.9940	0.0185

Tablica 6.2: Točnost i konačna vrijednost funkcije gubitka konvolucijskog modela na različitim skupovima.

6.2. Rezultati neprijateljskih napada

Za testiranje neprijateljskih napada korišten je skup od 30 nasumično odabranih MNIST slika iz skupa za testiranje. Svaka od znamenki 0-9 predstavljena je s 3 slike. Podaci vezani uz vjerojatnost ispravne klasifikacije odabranih slika prije napada dani su u nastavku.

vjerojatnost ispravne klasifikacije prije napada				
model	**prosjek**	**std**	**min**	**maks**
potpuno povezani	0.9694	0.0521	0.7860	1.0000
konvolucijski	0.9998	0.0007	0.9961	1.0000

Tablica 6.3: Srednja vrijednost, standardna devijacija te minimalna i maksimalna vrijednost vjerojatnosti ispravne klasifikacije 30 nasumično odabranih MNIST slika po modelu.

Svaki od tri ostvarena tipa napada (neusmjereni, usmjereni i "poboljšani" usmjereni) isproban je na potpuno povezanom i konvolucijskom modelu, pri čemu su neprijateljski uzorci generirani i uz NSGA-II i uz SPEA2.

Ograničenje količine šuma postavljeno je na 0.2, što znači da maksimalna dopuštena količina šuma na uzorku iznosi 5.6 (L_2 norma vektora šuma duljine $28 \cdot 28$ gdje su sve vrijednosti jednake 0.2 ili -0.2).

Za svaku od 30 odabranih slika te svaku kombinaciju algoritma, modela i vrste napada izgradnja neprijateljskih uzoraka pokrenuta je 10 puta. Broj iteracija te veličina populacije postavljeni su na 100.

6.2.1. Rezultati neusmjerenih napada

Ako uzmemo u obzir svih 30 slika i svih 10 pokretanja, tablica 6.4 prikazuje omjere ukupnog broja uspješnih neprijateljskih uzoraka i ukupnog broja uzoraka dobivenih u

nedominiranim skupovima po modelu i algoritmu.

generiranje uspješnih neprijateljskih uzoraka		
algoritam	**potpuno povezani model**	**konvolucijski model**
NSGA-II	0.5458	0.1964
SPEA2	0.5062	0.1578

Tablica 6.4: Omjeri ukupnog broja uspješnih neprijateljskih uzoraka i ukupnog broja uzoraka dobivenih u nedominiranim skupovima po modelu i algoritmu.

NSGA-II daje nešto više uspješnih uzoraka od SPEA2 za oba modela. Potpuno povezani model ima veći broj uspješnih uzoraka od konvolucijskog, što je za očekivati jer on u startu ima nižu prosječnu vjerojatnost ispravne klasifikacije pa ga je i lakše zavarati.

Neusmjereni napadi na potpuno povezani model

Zanima nas i količina uspješnih neprijateljskih uzoraka po odabranim slikama. Kako imamo 10 pokretanja svakog algoritma, a svaki algoritam pritom vraća nedominirani skup od više uzoraka, uvest ćemo tri različita omjera.

Prvo, promotrimo količinu slika za koje je generiran barem jedan uspješan neprijateljski uzorak u barem jednom od 10 pokretanja ("1+ u 1+" u tablici 6.5). Zatim, promotrimo koliko ima slika za koje je generiran barem jedan uspješan uzorak u svakom od 10 pokretanja (tablica 6.5, "1+ u svih 10"). Konačno, zanimat će nas i količina slika za koje su nedominirani skupovi sadržavali isključivo uspješne neprijateljske uzorke u svih 10 pokretanja u odnosu na ukupni broj slika (tablica 6.5, "svi u svih 10").

generiranje uspješnih neprijateljskih uzoraka po slikama			
algoritam	**1+ u 1+**	**1+ u svih 10**	**svi u svih 10**
NSGA-II	0.7667	0.6333	0.2667
SPEA2	0.8000	0.6667	0.2000

Tablica 6.5: Različiti omjeri broja slika s uspješnim uzorcima i ukupnog broja slika.

Analiza vrijednosti ciljnih funkcija (kao što su definirane u odjeljku 5.3.1) za uspješne neprijateljske uzorke svih 30 slika u svih 10 pokretanja daje podatke prikazane u tablicama 6.6 i 6.7.

vjerojatnost ispravne klasifikacije nakon napada				
algoritam	**prosjek**	**std**	**min**	**maks**
NSGA-II	0.1991	0.1417	0.0068	0.4982
SPEA2	0.1981	0.1428	0.0057	0.4973

Tablica 6.6: Srednja vrijednost, standardna devijacija te minimalna i maksimalna vrijednost vjerojatnosti ispravne klasifikacije nakon uspješnih neusmjerenih napada na 30 nasumično odabranih slika kroz 10 pokretanja za potpuno povezani model.

količina šuma				
algoritam	**prosjek**	**std**	**min**	**maks**
NSGA-II	1.5492	0.1193	1.2193	1.9664
SPEA2	1.6119	0.1447	1.2321	2.1130

Tablica 6.7: Srednja vrijednost, standardna devijacija te minimalna i maksimalna vrijednost količine šuma nakon uspješnih neusmjerenih napada na 30 nasumično odabranih slika kroz 10 pokretanja za potpuno povezani model.

Neusmjereni napadi na konvolucijski model

Podaci iz tablice 6.8 također ukazuju na općenito manju uspješnost napada na konvolucijski modele u odnosu na potpuno povezani model.

generiranje uspješnih neprijateljskih uzoraka po slikama			
algoritam	**1+ u 1+**	**1+ u svih 10**	**svi u svih 10**
NSGA-II	0.2667	0.2333	0.1000
SPEA2	0.2667	0.1667	0.0333

Tablica 6.8: Različiti omjeri broja slika s uspješnim uzorcima i ukupnog broja slika.

Međutim, srednje vrijednosti ciljnih funkcija za ovu vrstu modela nešto su niže od srednjih vrijednosti za potpuno povezani model.

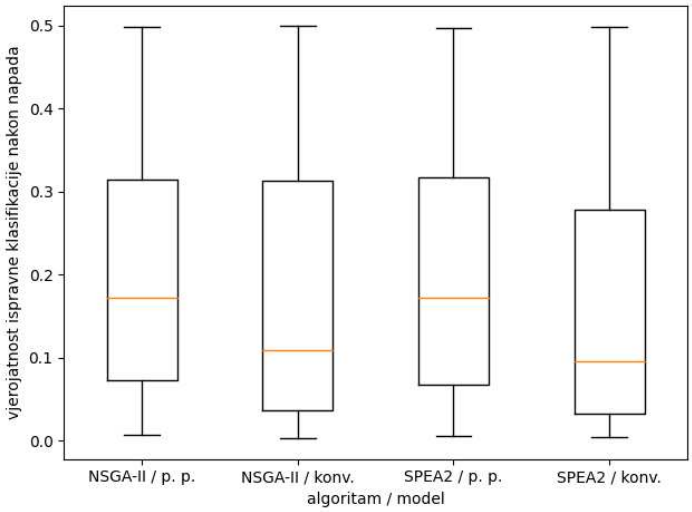
Iako je manje uspješnih neprijateljskih uzoraka generirano za konvolucijski model, oni koji jesu generirani naizgled bolje zavaravaju model i sadrže manju količinu šuma u odnosu na potpuno povezani model i njegove uzorke.

vjerojatnost ispravne klasifikacije nakon napada				
algoritam	**prosjek**	**std**	**min**	**maks**
NSGA-II	0.1736	0.1547	0.0024	0.4993
SPEA2	0.1613	0.1523	0.0041	0.4990

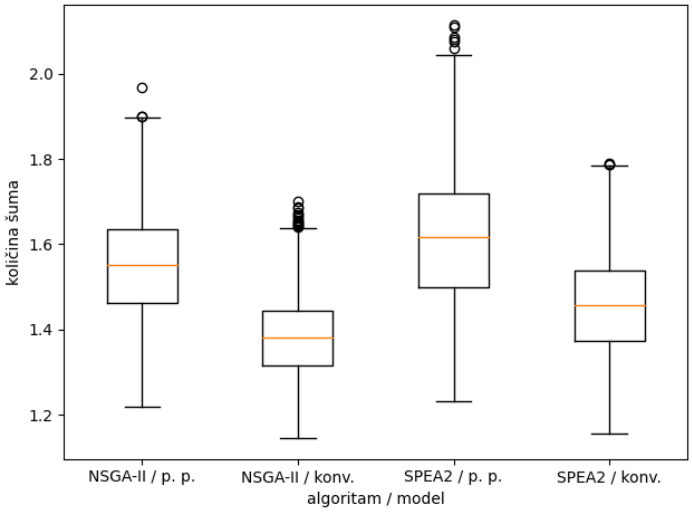
Tablica 6.9: Srednja vrijednost, standardna devijacija te minimalna i maksimalna vrijednost vjerojatnosti ispravne klasifikacije nakon uspješnih neusmjerenih napada na 30 nasumično odabranih slika kroz 10 pokretanja za konvolucijski model.

količina šuma				
algoritam	**prosjek**	**std**	**min**	**maks**
NSGA-II	1.3836	0.0913	1.1447	1.6995
SPEA2	1.4594	0.1140	1.1568	1.7883

Tablica 6.10: Srednja vrijednost, standardna devijacija te minimalna i maksimalna vrijednost količine šuma nakon uspješnih neusmjerenih napada na 30 nasumično odabranih slika kroz 10 pokretanja za konvolucijski model.



Slika 6.5: Grafički prikaz vjerojatnosti ispravne klasifikacije uzorka nakon uspješnog neusmjerenog napada za svaki algoritam i model.



Slika 6.6: Grafički prikaz količine šuma na uzorku nakon uspješnog neusmjerenog napada za svaki algoritam i model.

Primjeri dobivenih neprijateljskih uzoraka

U nastavku su izdvojeni neki od dobivenih neprijateljskih uzoraka. Za pet odabranih slika istaknuta su po tri neprijateljska uzorka dobivena u jednom pokretanju algoritma.

Neprijateljski uzorci posloženi su po rastućim vrijednostima vjerojatnosti pogrešne klasifikacije i količine šuma. Prvi redak u opisu ispod uzorka predstavlja sigurnost modela u ispravnu klasifikaciju uzorka, drugi redak predstavlja najveću sigurnost modela u pogrešnu klasifikaciju, a treći količinu šuma za taj uzorak.

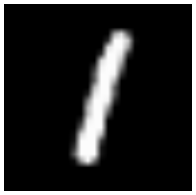
(a) 5: 87.13%,
3: 10.62%,
k. š.: 0.0000

(b) 5: 7.11%,
3: 90.84%,
k. š.: 1.3932

(c) 5: 3.46%,
3: 94.43%,
k. š.: 1.4777

(d) 5: 0.68%,
3: 97.92%,
k. š.: 1.8388

Slika 6.7: Originalni uzorak (a) te izabrani neprijateljski uzorci (b)-(d) dobiveni za potpuno povezani model uz NSGA-II (pokretanje 1 od 10). Svi neprijateljski uzorci dobiveni u ovom pokretanju uspjeli su zavarati model.



(a) 1: 99.49%,
8: 0.27%,
k. š.: 0.0000

(b) 1: 68.63%,
8: 28.56%,
k. š.: 1.3392

(c) 1: 46.30%,
8: 50.86%,
k. š.: 1.4624

(d) 1: 22.49%,
8: 74.77%,
k. š.: 1.7098

Slika 6.8: Originalni uzorak (a) te izabrani neprijateljski uzorci (b)-(d) dobiveni za potpuno povezani model uz NSGA-II (pokretanje 9 od 10). Neprijateljski uzorak (b) nije uspio zavarati model.

(a) 4: 82.92%,
9: 16.01%,
k. š.: 0.0000

(b) 4: 4.66%,
9: 93.88%,
k. š.: 1.4015

(c) 4: 2.86%,
9: 95.93%,
k. š.: 1.5125

(d) 4: 1.27%,
9: 97.67%,
k. š.: 1.7692

Slika 6.9: Originalni uzorak (a) te izabrani neprijateljski uzorci (b)-(d) dobiveni za potpuno povezani model uz SPEA2 (pokretanje 4 od 10). Svi neprijateljski uzorci dobiveni u ovom pokretanju uspješno su zavarali model.

(a) 2: 99.61%,
7: 0.39%,
k. š.: 0.0000

(b) 2: 2.98%,
7: 97.02%,
k. š.: 1.3732

(c) 2: 1.01%,
7: 98.99%,
k. š.: 1.4470

(d) 2: 0.247%,
7: 99.75%,
k. š.: 1.6474

Slika 6.10: Originalni uzorak (a) te izabrani neprijateljski uzorci (b)-(d) dobiveni za konvolucijski model uz NSGA-II (pokretanje 3 od 10). Svi neprijateljski uzorci dobiveni u ovom pokretanju uspješno su zavarali model.

(a) 3: 100.00%,
2: 0.00%,
k. š.: 0.0000

(b) 3: 58.53%,
2: 41.11%,
k. š.: 1.4836

(c) 3: 48.82%,
2: 50.78%,
k. š.: 1.5638

(d) 3: 37.84%,
2: 61.81%,
k. š.: 1.6735

Slika 6.11: Originalni uzorak (a) te izabrani neprijateljski uzorci (b)-(d) dobiveni za konvolucijski model uz SPEA2 (pokretanje 10 od 10). Neprijateljski uzorak (b) nije uspio zavarati model.

6.2.2. Rezultati usmjerenih napada

Implementacija usmjerenog napada ispitana je tako da se modele pokuša natjerati na neispravnu klasifikaciju nasumično odabranom labelom 3. Kod ove vrste napada zato nisu korištene 3 od 30 slika koje su inicijalno ispravno klasificirane tom labelom.

generiranje uspješnih neprijateljskih uzoraka		
algoritam	**potpuno povezani model**	**konvolucijski model**
NSGA-II	0.2681	0.0000
SPEA2	0.2573	0.0000

Tablica 6.11: Omjeri ukupnog broja uspješnih neprijateljskih uzoraka i ukupnog broja uzoraka dobivenih u nedominiranim skupovima po modelu i algoritmu.

Dobivena količina uspješnih neprijateljskih uzoraka značajno je manja od one za neusmjerene napade, što je u skladu s pretpostavkom da je teže natjerati model na pogrešnu klasifikaciju točno određenom labelom nego na bilo kakvu pogrešnu klasifikaciju.

NSGA-II nije značajnije nadmašio SPEA2 kod potpuno povezanog modela, a niti jedan algoritam nije uspio stvoriti neprijateljski uzorak kojim bi se zavarao konvolucijski model. U nastavku su, stoga, detaljnije razrađeni samo rezultati napada na potpuno povezani model.

Usmjereni napadi na potpuno povezani model

Prema omjerima iz donje tablice gotovo i nema razlike između rezultata dvaju algoritama za ovaj tip napada i modela.

generiranje uspješnih neprijateljskih uzoraka po slikama			
algoritam	**1+ u 1+**	**1+ u svih 10**	**svi u svih 10**
NSGA-II	0.4444	0.3333	0.0741
SPEA2	0.4444	0.3704	0.0741

Tablica 6.12: Različiti omjeri broja slika s uspješnim uzorcima i ukupnog broja slika.

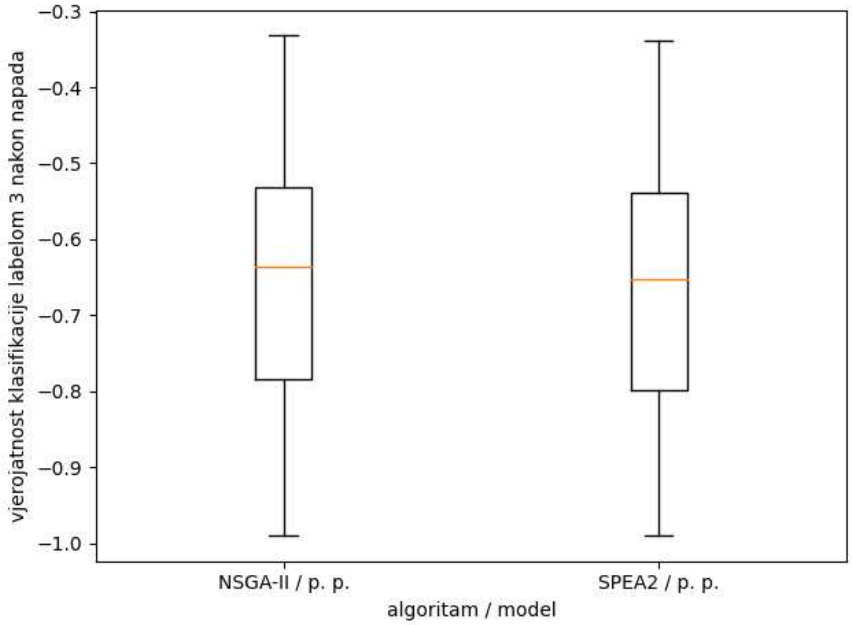
Prema odjeljku 5.3.1, ciljna funkcija vezana uz količinu šuma zajednička je usmjerenim i neusmjerenim napadima, ali se zato, umjesto minimizacije vjerojatnosti klasifikacije ispravnom labelom, u ovom slučaju maksimizira vjerojatnost klasifikacije odabranom pogrešnom labelom (u ovom slučaju, labelom 3). Dobiveni su sljedeći podaci.

vjerojatnost klasifikacije labelom 3 nakon napada				
algoritam	**prosjek**	**std**	**min**	**maks**
NSGA-II	0.6668	0.1650	0.3318	0.9888
SPEA2	0.6751	0.1661	0.3382	0.9905

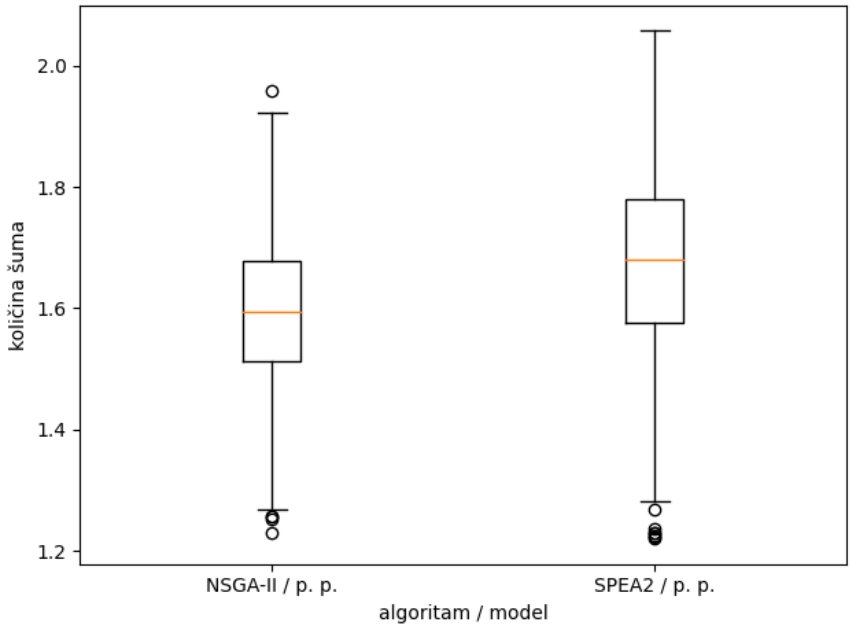
Tablica 6.13: Srednja vrijednost, standardna devijacija te minimalna i maksimalna vrijednost vjerojatnosti klasifikacije pogrešnom labelom 3 nakon uspješnih usmjerenih napada na 30 nasumično odabranih slika kroz 10 pokretanja za potpuno povezani model.

količina šuma				
algoritam	**prosjek**	**std**	**min**	**maks**
NSGA-II	1.5954	0.1176	1.2296	1.9583
SPEA2	1.6747	0.1408	1.2205	2.0569

Tablica 6.14: Srednja vrijednost, standardna devijacija te minimalna i maksimalna vrijednost količine šuma nakon uspješnih usmjerenih napada na 30 nasumično odabranih slika kroz 10 pokretanja za potpuno povezani model.



Slika 6.12: Grafički prikaz vjerojatnosti klasifikacije pogrešnom labelom 3 uzorka nakon uspješnog usmjerenog napada za svaki algoritam.



Slika 6.13: Grafički prikaz količine šuma na uzorku nakon uspješnog usmjerenog napada za svaki algoritam.

Primjeri dobivenih neprijateljskih uzoraka

Za ovaj tip napada izdvojeni su neprijateljski uzorci potpuno povezanog modela dobiveni za dvije slike.

Drugi redak u opisu ispod uzorka u ovom slučaju predstavlja sigurnost modela u klasifikaciju uzorka odabranom neprijateljskom labelom 3.

(a) 2: 94.22%,
3: 5.68%,
k. š.: 0.0000

(b) 2: 23.12%,
3: 76.79%,
k. š.: 1.4562

(c) 2: 12.48%,
3: 87.46%,
k. š.: 1.6105

(d) 2: 6.35%,
3: 93.61%,
k. š.: 1.9014

Slika 6.14: Originalni uzorak (a) te izabrani neprijateljski uzorci (b)-(d) dobiveni za potpuno povezani model uz NSGA-II (pokretanje 10 od 10). Svi neprijateljski uzorci dobiveni u ovom pokretanju uspješno su zavarali model.

(a) 9: 97.15%,
3: 0.03%,
k. š.: 0.0000

(b) 9: 38.45%,
3: 52.76%,
k. š.: 1.5077

(c) 9: 22.26%,
3: 70.21%,
k. š.: 1.6142

(d) 9: 9.70%,
3: 86.19%,
k. š.: 1.8712

Slika 6.15: Originalni uzorak (a) te izabrani neprijateljski uzorci (b)-(d) dobiveni za potpuno povezani model uz NSGA-II (pokretanje 1 od 10). Svi neprijateljski uzorci dobiveni u ovom pokretanju uspješno su zavarali model.

6.2.3. Rezultati "poboljšanih" usmjerenih napada

Ova inačica usmjerenih napada ističe se time što uvodi treću ciljnu funkciju (definiranu u 5.3.1) kojom se nastoji minimizirati suma vjerojatnosti klasifikacije uzorka labelama različitim od odabrane pogrešne labele (labela 3).

generiranje uspješnih neprijateljskih uzoraka		
algoritam	**potpuno povezani model**	**konvolucijski model**
NSGA-II	0.2776	0.0000
SPEA2	0.2519	0.0000

Tablica 6.15: Omjeri ukupnog broja uspješnih neprijateljskih uzoraka i ukupnog broja uzoraka dobivenih u nedominiranim skupovima po modelu i algoritmu.

Ispostavlja se da ovakva implementacija "poboljšanih" usmjerenih napada ne predstavlja značajno poboljšanje nad "običnim" usmjerenim napadima. Ponovno nije generiran niti jedan uspješan neprijateljski uzorak za konvolucijski model.

"Poboljšani" usmjereni napadi na potpuno povezani model

Zbog velike sličnosti s rezultatima iz prethodnog odjeljka, u nastavku su dani samo podaci za vrijednosti treće ciljne funkcije kod uspješnih neprijateljskih uzoraka.

suma vjerojatnosti klasifikacije labelama različitim od 3				
algoritam	**prosjek**	**std**	**min**	**maks**
NSGA-II	0.3434	0.1656	0.0145	0.6672
SPEA2	0.3298	0.1676	0.0111	0.6587

Tablica 6.16: Srednja vrijednost, standardna devijacija te minimalna i maksimalna vrijednost sume vjerojatnosti klasifikacije labelama različitim od 3 nakon uspješnih "poboljšanih" usmjerenih napada na 27 nasumično odabranih slika kroz 10 pokretanja za potpuno povezani model.

6.2.4. Napadi na jedan model pomoću uzoraka za drugi model

Pretpostavlja se da bi uzorci generirani za konvolucijski model uspjeli zavarati potpuno povezani model u većem broju nego što bi to mogli napraviti uzorci potpuno povezanog modela primijenjeni na konvolucijski. Zbog veće točnosti konvolucijskog modela dobiveni uzorci trebali bi biti "jači" od onih za potpuno povezani model pa bi, stoga, mogli biti i općenito uspješniji.

Međutim, primjeri neprijateljskih uzoraka prikazani u prošlim odjeljcima nisu mogli navesti modele različite od onih za koje su izgrađeni na pogrešnu klasifikaciju, neovisno o tipu napada. To bi moglo značiti da uspješnost ovako generiranih uzoraka ovisi o sličnosti između modela koji napadaju i modela za koji su razvijeni.

7. Zaključak

Ovaj rad pristupio je neprijateljskim napadima na klasifikacijske modele kao problemima višekriterijske optimizacije. Potpuno povezana te konvolucijska umjetna neuronska mreža trenirane su i ispitane na MNIST skupu podataka, na kojem su postigle visoku točnost klasifikacije (oko 95% odnosno 99%). Ti su modeli zatim napadani neprijateljskim uzorcima stvorenim pomoću odabranih algoritama višekriterijske optimizacije, NSGA-II i SPEA2. Neprijateljski uzorak gradio se tako da je originalnoj MNIST slici, predstavljenoj vektorom realnih brojeva iz intervala $[0, 1]$, dodan šum dobiven evolucijskim postupkom.

Neprijateljski napad definiran je kao problem višekriterijske optimizacije s 2 osnovne ciljne funkcije kojima se nastojala minimizirati količina šuma na uzorku te minimizirati vjerojatnost ispravne klasifikacije danog uzorka (za neusmjereni napad) odnosno maksimizirati vjerojatnost pogrešne klasifikacije uzorka zadanom labelom (za usmjereni napad).

Dobiveno je značajno više uspješnih neprijateljskih uzoraka za neusmjerene napade nego za usmjerene, što je bilo za očekivati zbog jednostavnije prirode neusmjerenih napada. Kod obje vrste napada više uspješnih uzoraka generirano je za potpuno povezani model nego za konvolucijski, što je ponovno bilo u skladu s očekivanjima jer je konvolucijski model teže zavarati zbog njegove veće točnosti. Nije opažena značajna razlika u količini uspješnih napada s obzirom na vrstu algoritma kojim su izgrađeni.

Pokušaj poboljšanja usmjerenih napada nije bio uspješan iako je u tu svrhu uvedena i treća funkcija cilja, za minimizaciju sume vjerojatnosti klasifikacije uzorka labelama različitim od zadane pogrešne labele.

Izabrani neprijateljski uzorci razvijeni za potpuno povezani model nisu uspješno primijenjeni na konvolucijski model, i obratno.

LITERATURA

[1] The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/. Pristupljeno: 2020-05-25.

[2] Moustafa Alzantot, Yash Sharma, Supriyo Chakraborty, Huan Zhang, Cho-Jui Hsieh, i Mani B Srivastava. Genattack: Practical black-box attacks with gradient-free optimization. U *Proceedings of the Genetic and Evolutionary Computation Conference*, stranice 1111–1119, 2019.

[3] Bojana Dalbelo Bašić, Marko Čupić, i Jan Šnajder. Prezentacija Umjetne neuronske mreže. https://www.fer.unizg.hr/_download/repository/UI_12_UmjetneNeuronskeMreze[1].pdf, 2020. Pristupljeno: 2020-05-19.

[4] Kalyan Deb. Multi-objective optimization using evolutionary algorithms: An introduction. Technical Report 2011003, KanGAL, 02 2011.

[5] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, i TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[6] Ian Goodfellow, Yoshua Bengio, i Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org. Pristupljeno: 2020-05-19.

[7] Josip Krapac i Siniša Šegvić. Prezentacija Konvolucijski modeli. http://www.zemris.fer.hr/~ssegvic/du/du2convnet.pdf. Pristupljeno: 2020-05-19.

[8] Margarita Osadchy, Julio Hernandez-Castro, Stuart Gibson, Orr Dunkelman, i Daniel Pérez-Cabo. No bot expects the DeepCAPTCHA! Introducing immutable adversarial examples, with applications to CAPTCHA generation. *IEEE Transactions on Information Forensics and Security*, 12(11):2640–2653, 2017.

[9] Irhum Shafkat. Intuitively understanding convolutions for deep learning. https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1, 06 2018. Pristupljeno: 2020-05-19.

[10] Jiawei Su, Danilo Vasconcellos Vargas, i Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

[11] Eckart Zitzler, Marco Laumanns, i Lothar Thiele. SPEA2: Improving the strength Pareto evolutionary algorithm. *TIK-report*, 103, 2001.

[12] Marko Čupić. Prirodom inspirirani optimizacijski algoritmi. Metaheuristike. http://java.zemris.fer.hr/nastava/pioa/knjiga-0.1.2013-12-30.pdf, 12 2013. Pristupljeno: 2020-05-20.

[13] Marko Čupić. Skripta Umjetna inteligencija: Umjetne neuronske mreže. http://java.zemris.fer.hr/nastava/ui/es/es-20200415.pdf, 2020. Pristupljeno: 2020-05-19.

[14] Marko Čupić, Bojana Dalbelo Bašić, i Marin Golub. Neizrazito, evolucijsko i neuroračunarstvo. http://java.zemris.fer.hr/nastava/nenr/knjiga-0.1.2013-08-12.pdf, 08 2020. Pristupljeno: 2020-05-19.

Neprijateljski napadi na klasifikacijske modele uz višekriterijsku optimizaciju

Sažetak

Ovaj se rad bavi *black-box* napadima na modele za klasifikaciju slike uz odabrane algoritme za višekriterijsku optimizaciju, NSGA-II i SPEA2. Neprijateljski napadi prikazani su kao problemi višekriterijske optimizacije te su kao takvi i implementirani. Klasifikacijski modeli predstavljeni su dvjema umjetnim neuronskim mrežama, potpuno povezanom i konvolucijskom, koje su trenirane i ispitane na MNIST skupu slika rukom pisanih znamenki uz visoku točnost. Generirani su neprijateljski uzorci dodavanjem evoluiranog šuma vektoriziranim MNIST slikama. Značajno više uspješnih neprijateljskih uzoraka dobiveno je za neusmjerene napade u odnosu na usmjerene te za potpuno povezani model u odnosu na konvolucijski. Nije opažena značajna razlika u količini uspješnih napada s obzirom na algoritam višekriterijske optimizacije.

Ključne riječi: Višekriterijska optimizacija, neprijateljski napadi, umjetne neuronske mreže, klasifikacija slike.

Adversarial attacks on classification models using multi-objective optimization

Abstract

This thesis deals with staging *black-box* attacks on image classification models using selected multi-objective optimization algorithms, NSGA-II and SPEA2. Adversarial attacks are presented as multi-objective optimization problems and are implemented as such. A simple, fully-connected artificial neural network and a convolutional one were built and then trained on the MNIST handwritten digits dataset, achieving high accuracy, to be used as targets for adversarial attacks. Adverarial images were generated by adding evolved noise vectors to vectorized MNIST images. Significantly more successful adversarial images were obtained for untargeted attacks versus targeted ones, and for the simple model versus the convolutional one. No significant difference was observed in the number of successful adversarial images with respect to the type of algorithm with which they were constructed.

Keywords: Multi-objective optimization, adversarial attacks, artificial neural networks, image recognition.