

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2231

Obrana dubokih konvolucijskih modela od neprijateljskih primjera

Matej Dobrovodski

Zagreb, lipanj 2020.

**SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

Zagreb, 13. ožujka 2020.

DIPLOMSKI ZADATAK br. 2231

Pristupnik: **Matej Dobrovodski (0036491377)**

Studij: Računarstvo

Profil: Računarska znanost

Mentor: izv. prof. dr. sc. Zoran Kalafatić

Zadatak: **Obrana dubokih konvolucijskih modela od neprijateljskih primjera**

Opis zadatka:

Duboke neuronske mreže postižu visoku točnost u mnogim primjenama poput klasifikacije slika. Međutim, naučene duboke modele teško je interpretirati, a često su i osjetljivi na tzv. neprijateljske primjere. To svojstvo otvara mogućnost zlonamjernog oblikovanja neprijateljskih primjera koji bi mogli zavarati sustav temeljen na dubokom modelu. U okviru diplomskog rada potrebno je proučiti metode generiranja neprijateljskih primjera opisane u literaturi, kao i postupke obrane od takvih napada. Implementirati nekoliko odabralih metoda suparničkih napada i obrane od takvih napada. Pripremiti bazu uzoraka za učenje i testiranje sustava te analizirati dobivene rezultate. Radu priložiti izvorni i izvršni kôd razvijenih postupaka, ispitne slike i rezultate, uz potrebna objašnjenja i dokumentaciju.

Rok za predaju rada: 30. lipnja 2020.

SADRŽAJ

1. Uvod	1
1.1. Raspoznavanje objekata	1
1.2. Neprijateljski primjeri	2
2. Programska potpora	5
2.1. Odabir biblioteke za duboko učenje	5
2.2. Biblioteke za neprijateljske primjere	5
2.3. Skupovi podataka	6
2.4. Konvolucijski modeli	7
3. Neprijateljski primjeri I	9
3.1. Model prijetnje	9
3.2. Pojava prvih neprijateljskih primjera	12
3.3. Brza metoda temeljena na gradijentima	13
3.4. DeepFool	15
4. Obrana dubokih konvolucijskih modela I	19
4.1. Jednostavne obrane	19
4.1.1. JPEG kompresija	20
4.1.2. Stiskanje značajki	22
4.2. Neprijateljsko treniranje - FGSM	23
4.3. Termometar kodiranje	25
4.4. Obračbena destilacija	27
5. Neprijateljski primjeri II	29
5.1. Neučinkovitost obrana	29
5.2. Projicirani gradijentni spust	30
5.3. Napadi <i>Carlini and Wagner</i>	31
5.3.1. Napad na termometar kodiranje	35

5.4. Granični napad	36
5.5. Prenosivost napada	41
5.5.1. Prenosivost neprijateljskih primjera između modela	41
5.5.2. Prenosivost neprijateljskih primjera na obrambenu destilaciju	45
6. Obrana dubokih konvolucijskih modela II	46
6.1. Preduvjeti uspješnih obrana	46
6.2. Neprijateljsko treniranje - PGD	48
6.2.1. <i>Fast is better than free</i>	50
6.3. Dokazivost obrane od neprijateljskih napada	51
6.4. Budući rad	51
7. Zaključak	52
Literatura	53
8. Dodatak	57
8.1. Osobni skup slika	57
8.2. Izlazi modela na nepromijenjenim slikama iz osobnog skupa	57

1. Uvod

1.1. Raspoznavanje objekata

Raspoznavanje objekata jedan je od ključnih problema područja računalnog vida. Pri rješavanju problema raspoznavanja objekata se na ulaz nekog sustava dovede slika nekog objekta, a na izlazu se očekuje ispravna klasifikacija u neki od predodređenih razreda. Čovjeku ovaj zadatak ne predstavlja veliki problem, no još uvijek ne postoji zadovoljavajuće rješenje problema koje bi vrijedilo za opći slučaj. Trenutno najbolja takva rješenja temelje se na konvolucijskim neuronskim mrežama.

Razvoj konvolucijskih mreža počeo je osamdesetih godina prošlog stoljeća pojavom *neocognitron-a* – neuronske mreže inspirirane biološkim stanicama vidne kore mozga. Krajem devedesetih godina se pojavljuje konvolucijska neuronska mreža LeNet5. LeNet5 mreža je vrlo uspješno raspoznavala rukom pisane znamenke te je ova mreža bila početna točka za daljnja istraživanja drugih neuronskih mreža.

ImageNet^[1] projekt je velika baza podataka predviđena za istraživanje područja raspoznavanja objekata. S više od 14 milijuna slika podijeljenih u 20000 kategorija, *ImageNet* skup je daleko najveći slobodno dostupni skup. Počevši od 2010. godine, *ImageNet* projekt organizira godišnje natjecanje, *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC). Veliki skok u točnosti pri raspoznavanju dogodio se 2012. godine kada je konvolucijska neuronska mreža *AlexNet*^[2] postigla top-5 pogrešku od samo 15.3%, što je bilo 10.8% manje od sljedeće mreže. To je postignuto korištenjem grafičkih procesora pri treniranju, što je potaknulo svojevrsnu revoluciju u području dubokog učenja.

Do 2017. godine, većina timova u natjecanju je imala top-5 točnost veću od 95%. Danas se u raznim bibliotekama mogu naći unaprijed istrenirane mreže koje postižu vrlo dobre rezultate, primjerice *ResNet*, *Xception* i *VGG*. Sve spomenute mreže postižu vrlo zadovoljavajuće točnosti pri ispitivanju (top-5 točnosti iznad

90%) i čini se da mogu dobro generalizirati. No u nastavku rada će biti pokazan oblik napada na konvolucijske modele koji dovodi u pitanje činjenicu da današnje konvolucijske mreže dobro generaliziraju.

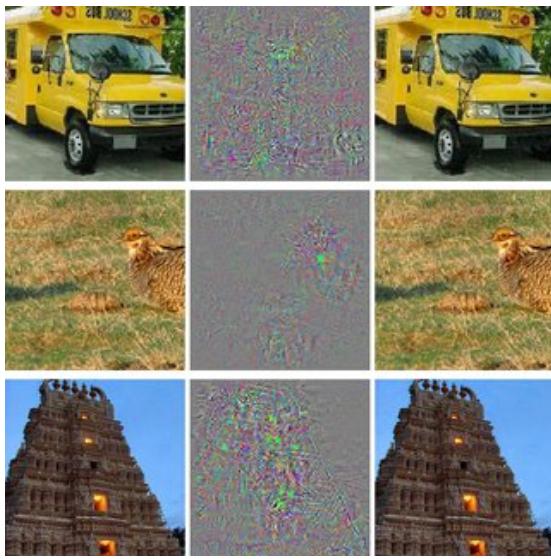
1.2. Neprijateljski primjeri

Krajem 2013. godine pojavljuje se prvi izravni “napad” na duboke neuronske mreže^[3], gdje je jedna od meta bila prethodno spomenuta uspješna mreža *AlexNet*. Polazna pretpostavka je da duboki modeli, usprkos tome što dobro generaliziraju, imaju ugrađene svojevrsne *slike pjege* koje se isplati istražiti.

Vrijedi da za neki ispravno klasificirani ulaz \mathbf{x} postoji područje $\mathbf{x} + \mathbf{r}$ u blizini ulaza te je uobičajeno da modeli ulazne vrijednosti iz tog područja također ispravno klasificiraju, isto kao i \mathbf{x} . U općem slučaju vrijedi da neprimjetne perturbacije iz tog područja (npr. nasumični šum slabog intenziteta) ne mijenjaju izlaz modela. To je pretpostavka lokalne generalizacije i tipično vrijedi za probleme iz područja računalnogvida.

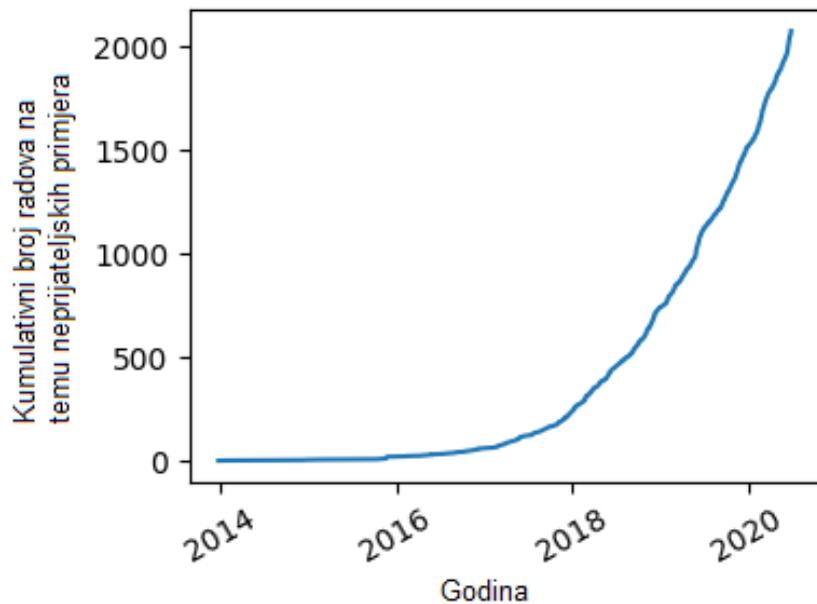
Međutim, ispostavilo se da pretpostavka lokalne generalizacije ne vrijedi uvi-jek. Moguće je konstruirati perturbaciju \mathbf{r} koja dovodi do pogrešne klasifikacije slike $\mathbf{x} + \mathbf{r}$, a ljudskom oku nije uočljiva. Takve slike se nazivaju neprijateljskim primjerima, a taj pojam se može generalizirati i na mnoga druga područja i na sličan način se mogu napasti sustavi pretvaranja teksta u govor, sustavi za de-tekciju zločudnih programa i praktički svi sustavi koji se oslanjaju na dosadašnje modele dubokog učenja.

Ono što je iznenadjuće i što je potaklo daljnje istraživanje je to što je zapravo iznimno lako za pronaći takve neprijateljske primjere na *state of the art* modelima kod kojih je perturbacija \mathbf{r} potpuno neprimjetna i to što nije nimalo očito zašto mreže neispravno klasificiraju takve ulaze. Jedan od originalnih napada je prikazan na slici 1.1 gdje ranije spomenuta *AlexNet* mreža predviđa da su novonastale slike zapravo slike noja.



Slika 1.1: Primjer suparničkog napada na *AlexNet* mrežu^[3]. U lijevom stupcu su originalne, ispravno klasificirane slike. U srednjem stupcu se nalazi perturbacija koja se nadodaje na originalnu sliku, a u desnom stupcu su sve tri novonastale slike klasificirane kao noj.

Motivacija za istraživanje neprijateljskih primjera je višestruka. Jedan od razloga je zaštita postojećih dubokih modela od zlonamjernih neprijateljskih napada. Uz to, vrlo bitan razlog je i produbljivanje postojećeg teorijskog znanja o dubokim modelima. Pojava neprijateljskih primjera nije izolirana niti rijetka, a jednostavnost kojom se konstruiraju je svakako zanimljiva. Dapače, u zadnjih nekoliko godina je područje neprijateljskih primjera doživjelo svojevrsnu eksploziju. U trenutku pisanja ovog rada je približno jednak broj radova objavljen u prethodnoj godini dana kao i svih godina prije toga. Ovo se vidi i na slici 1.2 koja prikazuje graf ukupnog broja radova na temu neprijateljskih primjera počevši od 2014. godine.



Slika 1.2: Ukupan broj radova na temu neprijateljskih primjera kroz godine.^[4]

U radu je dan pregled nekoliko metoda generiranja neprijateljskih primjera. Neke metode su prikazane zbog njihove povijesne važnosti i utjecaja na daljnji razvoj metoda, dok su neke metode iznimno snažne i mjerilo uspješnosti obrane od suparničkih napada. Pokazano je i koliko su postojeći napadi uspješni protiv poznatih mreža te kako niti jedan od korištenih modela nije unaprijed otporan na napade. Uz napade su pokazane i neke obrane, s naglaskom na njihovu (ne)uspješnost pri odupiranju od postojećih suparničkih napada, probleme koji su gotovo svim obranama zajednički te potencijalnu budućnost razvoja uspješnijih obrana.

2. Programska potpora

2.1. Odabir biblioteke za duboko učenje

Postoji mnogo biblioteka koje pružaju sve potrebno za duboko učenje i računalni vid. U nastavku rada se koristi *Tensorflow 2^[5]* u kombinaciji s bibliotekom *Keras^[6]*. Zbog ogromnog dobitka u brzini izvođenja, ove biblioteke su korištene zajedno s platformom *CUDA* koja omogućava iskorištanje grafičkog procesora za obradu opće namjene (engl. *graphics processing unit for general purpose processing*, GPGPU). Grafička kartica korištena u sklopu generiranja rezultata u radu je NVIDIA GeForce RTX 2060 SUPER.

2.2. Biblioteke za neprijateljske primjere

Usprkos tome što su suparnički primjeri relativno nov koncept, već postoji mnogo biblioteka koje pružaju implementaciju velikog broja suparničkih napada, a često su i napadi implementirani izravno od strane autora napada. Istaknute su se tri biblioteke za generiranje suparničkih napada: *CleverHans^[7]*, *Foolbox Native^[8]* i *Adversarial Robustness Toolbox (ART)^[9]*.

Za odabir biblioteke je razmatrano nekoliko stvari: dostupnost i ekstenzivnost dokumentacije, raznovrsnost implementiranih napada, jednostavnost korištenja, zahtijevana programska potpora te postoji li implementacija obrana. Za svaku biblioteku je implementirano generiranje neprijateljskih primjera napadom FGSM koji je opisan u potpoglavlju 3.3, a napad je proveden na model opisan u potpoglavlju 2.4.

Cleverhans ima kratku dokumentaciju za sve napade i poveznicu na relevantni rad koji opisuje napad, međutim ne postoji dokumentacija u formatu koji se lako pretražuje. *Foolbox* i *ART* imaju dokumentaciju dostupnu u takvom formatu, međutim *Foolbox* dokumentacija ne opisuje kako se napad poziva i s kojim argumentima, što otežava korištenje bez detaljnijeg proučavanja izvornog kôda, dok

je *ART* dokumentacija eksplisitna kod toga.

Što se tiče jednostavnosti korištenja, *Cleverhans* je bio najjednostavniji za primjenu u ovom jednostavnom primjeru. *Foolbox* zahtjeva da se slike pretvore u određen format prije pokretanja napada, što otežava korištenje. *ART*, međutim, zahtjeva dodatne informacije pri konstruiranju napada kao što su broj razreda, dimenzije ulaza, funkcija gubitka i granične vrijednosti, što druge biblioteke ne traže. K tome je vrijeme izvođenja bilo najbrže za *ART*. Ovo je bitno jer se korišteni napad često koristi pri evaluaciji određenih obrana.

Cleverhans i *Foolbox* imaju vrlo specifične zahtjeve za programsku potporu, iako će *Cleverhans* u budućnosti podupirati više od samo *Tensorflow*. *ART* pruža potporu za mnoštvo biblioteka: *Tensorflow* (v1 i v2), *Keras*, *PyTorch*, *MXNet* i još njih.

Od navedenih biblioteka, *ART* je jedina koja već sada ima implementirane neke od obrana u literaturi. *Cleverhans* biblioteka ima planove za implementaciju obrana u budućnosti, dok *Foolbox* podržava samo napade.

Zbog svega navedenog, u nastavku rada se koristi *Adversarial Robustness Toolbox*. Dodatno, autori biblioteke su vrlo aktivni na *GitHub*-u i iznimno brzo reagiraju kada se postavi pitanje ili prijavi problem. Pri izradi rada otkriveno je nekoliko *bug*-ova koji su popravljeni u najkraćem roku.

2.3. Skupovi podataka

ImageNet^[1] je široko korištena baza podataka slika s preko 14 milijuna slika raspoređenih u više od 20000 razreda. *ImageNet* je praktički postao standard za treniranje i evaluaciju rada modela pri klasifikaciji objekata. Neki od modela korišteni u radu su unaprijed trenirani na *ImageNet* bazi podataka na kojima postižu iznimno visoku točnost.

Neke obrane i posljedično napadi će u nastavku rada biti evaluirani na *CIFAR-10*^[10] skupu podataka. *CIFAR-10* sadrži samo 10 disjunktnih razreda, te 50000 slika za treniranje mreže. Nužno je koristiti i ovaj skup podataka jer pojedine obrane još uvijek nije moguće skalirati na *ImageNet* razinu jer vrijeme izvođenja nije razumno.

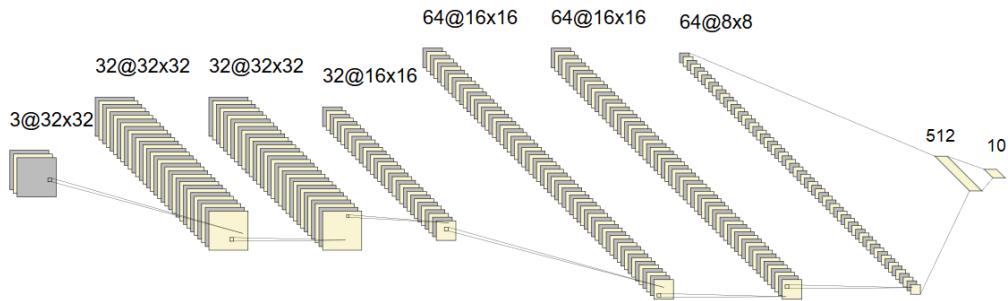
U svrhu rada je također odabранo 16 slika koje bi *ImageNet* modeli trebali ispravno klasificirati. Popis slika i izlazi određenih modela nalaze se u dodatku⁸.

2.4. Konvolucijski modeli

Primarna meta napada u radu je mreža *ResNet V2*^[11], i to verzija s 50 slojeva koja je unaprijed trenirana na *ImageNet* skupu. Mreža postiže top-1 točnost od 76%, te top-5 točnost od 93%. U početnim fazama izrade rada razmatrano je više mreža, međutim *ResNet* mreža je puno brža pri evaluaciji što ubrzava i olakšava evaluaciju napada i obrana. Korištenje ove mreže ne smanjuje općenitost ideja predstavljenih u radu, pošto su sve konvolucijske mreže jednako ranjive na neprijateljske napade. Lakoća provođenja neprijateljskih napada je “problem” koji sve konvolucijske mreže dijele u jednakoj mjeri, i trenutno ne postoji niti jedna takva mreža koja je sama po sebi otporna na njih. Dodatno, kôd priložen uz rad podupire provođenje napada na sljedeće mreže: *DenseNet121*, *VGG16*, *VGG19*, *MobileNetV2* te *Xception*.

Osim *ResNet* mreže, dodatno je konstruirana i jednostavna konvolucijska mreža za klasifikaciju *CIFAR-10* slika. Mreža se sastoji od 11 slojeva, redom:

- konvolucijski sloj oblika $32 \times 32 \times 32$ s filtrom veličine 3×3
- konvolucijski sloj oblika $32 \times 32 \times 32$ s filtrom veličine 3×3
- sloj sažimanja oblika 2×2
- sloj ispadanja s vjerojatnošću 0.25
- konvolucijski sloj oblika $16 \times 16 \times 64$ s filtrom veličine 3×3
- konvolucijski sloj oblika $16 \times 16 \times 64$ s filtrom veličine 3×3
- sloj sažimanja oblika 2×2
- sloj ispadanja s vjerojatnošću 0.25
- potpuno povezani sloj veličine 512
- sloj ispadanja s vjerojatnošću 0.50
- potpuno povezani sloj veličine 10, pošto se klasificira u 10 razreda



Slika 2.1: Skica jednostavne mreže za *CIFAR-10* mrežu. Nisu prikazani slojevi ispananja.

Ukupno mreža ima 2, 168, 362 parametara koji se mogu naučiti. Na slici 2.1 je skica mreže. Aktivacijska funkcija između relevantnih slojeva je *ReLU*. Mreža već nakon 15 epoha lako postiže točnost od 75% na skupu za testiranje korištenjem optimizacijskog algoritma Adam^[12] uz prepostavljene zadane hiperparametre od strane *Tensorflow* biblioteke. Nakon 25 epoha postiže točnost od 79.47%, no i u tom trenutku mreža nije pretrenirana. Za potrebe rada nije nužno maksimizirati točnost jer ne bi bitno utjecalo na rezultate. Jednako je lagano za pronaći suparničke primjere i na vrlo dobro istreniranim dubokim mrežama kao i na ovakvim jednostavnim mrežama.

3. Neprijateljski primjeri I

3.1. Model prijetnje

Model prijetnje (engl. *threat model*) je proces kojim se potencijalne prijetnje mogu nabrojati i identificirati te se mogu odrediti određene mjere kao prioritet. Neki od dijelova modela prijetnje mogu biti: frekvencija interakcije s metom napada, željena vrsta pogrešne klasifikacije, količina znanja o meti napada i specifičnost napada. Prije opisivanja navedenih aspekata modela prijetnje uvedeno je nekoliko relevantnih simbola koji se učestalo pojavljuju u literaturi i u ovom radu.

Osnovni pojmovi i simboli

- $f(\cdot)$ – model dubokog učenja
- \mathbf{x}, l – originalni ulaz te pripadajuća labela
- \mathbf{x}', l' – neprijateljski primjer i pripadajuća labela
- $J(\cdot)$ – funkcija gubitka, u većini slučajeva gubitak unakrsne entropije
- $\|\cdot\|_p$ – p-norma, p je najčešće 0, 2 ili ∞ . Dodatna oznaka za norme je ℓ_p .

Norme

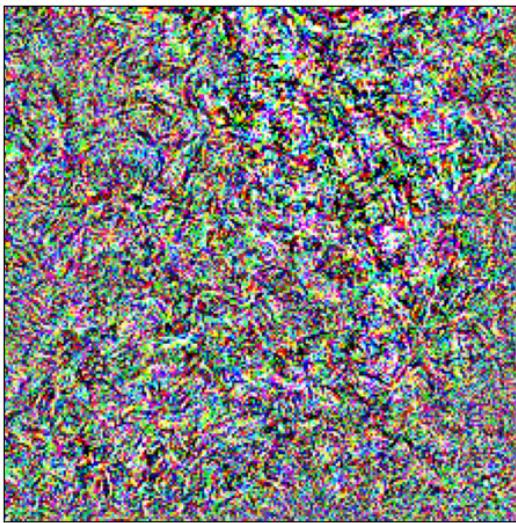
- ℓ_0 – $\|\mathbf{x}\|_0$ predstavlja broj ne-nula elemenata vektora \mathbf{x} .
- ℓ_2 – $\|\mathbf{x}\|_2 := \sqrt{x_1^2 + \dots + x_n^2}$, odnosno Euklidska norma.
- ℓ_∞ – $\|\mathbf{x}\|_\infty := \max_i |x_i|$. U kontekstu neprijateljskih napada ova norma predstavlja maksimalnu vrijednost perturbacije r .

Frekvencija interakcije s metom napada

- Jednokratni napad – jednokratni napadi (engl. *one-time*) su napadi kojima je potreban samo jedan pristup modelu da generiraju neprijateljski primjer. Ovi napadi su brzi, ali i mnogo slabiji od iterativnih napada te nisu u fokusu istraživanja. Na primjer, napad opisan u potpoglavlju 3.3 je jednokratni napad.
- Iterativni napad – iterativni napadi zahtijevaju više pristupa modelu da bi generirali neprijateljski primjer. Ovakvi napadi generiraju daleko bolje neprijateljske primjere. Većina napada pripada ovoj kategoriji.

Vrsta pogrešne klasifikacije

- Lažno pozitivni napad – lažno pozitivni (engl. *false positive*) primjeri u kontekstu neprijateljskih napada kod klasifikacijskih problema su čovjeku potpuno nepoznatljivi (npr. šum), dok mreža s visokom vjerojatnošću klasificira sliku.
- Lažno negativni napad – lažno negativni (engl. *false negative*) primjeri su oni koje čovjek vrlo lako prepozna, a mreža pogrešno klasificira zbog nevidljive perturbacije. Fokus rada su od početka lažno negativni primjeri. Usporedba napada dana je u slici 3.1.



(a) Lažno pozitivni napad. Mreža klasificira sliku kao prostirka za molitvu (engl. *prayer rug*) s vjerojatnošću 98.66%.



(b) Lažno negativni napad. Mreža klasificira sliku kao zmaj (engl. *kite*) s vjerojatnošću 91.89%.

Slika 3.1: Primjeri lažno pozitivnog i lažno negativnog napada. Napad proveden na *ResNet50V2* mreži korištenjem *Carlini and Wagner* ℓ_2 napada opisanog u potpoglavlju 5.3.

Znanje o meti napada

- Bijela kutija – model se naziva bijela kutija (engl. *white box*) ako je sve o modelu unaprijed poznato napadaču, uključujući: arhitekturu, parametre mreže, aktivacijske funkcije, hiperparametre i sve druge moguće detalje mreže. Napadi koji se temelje na modelu bijele kutije često iskorištavaju gradijente mreže pri konstruiranju neprijateljskog primjera.
- Crna kutija – suprotno tome, model crne kutije (engl. *black box*) pretpostavlja nedostatak svih mogućih informacija, osim izlaza iz mreže. Na primjer, ako se napada neka mreža “u oblaku”, njoj se pristupa tako da joj se preda ulaz te nije moguće direktno saznati dodatne detalje o mreži. Začudo, moguće je konstruirati neprijateljske primjere samo na temelju izlaza mreže.

Specifičnost napada

- Ciljani napad – ciljni napad (engl. *targeted attack*) je oblik napada gdje se za neki neprijateljski primjer pokušava dobiti unaprijed određen izlaz mreže. Uz to, dodatni zahtjev može biti i da se maksimizira vjerojatnost odabranog razreda.
- Neciljni napad – neciljni napad (engl. *untargeted attack*) zahtjeva jedino da je klasifikacija neprijateljskog primjera neispravna. Općenito je lakše i brže konstruirati neciljni napad.

3.2. Pojava prvih neprijateljskih primjera

U uvodu je opisana osnovna ideja neprijateljskih primjera iz jednog od najranijih radova na temu neprijateljskih primjera^[3], a u nastavku je ideja dodatno razrađena i ukratko opisana optimizacijska metoda generiranja suparničkih primjera.

Implicitno je pretpostavljeno da mreže imaju svojstvo lokalne generalizacije. Za neki dovoljno mali radius $\epsilon > 0$ u blizini ulaza \mathbf{x} (epsilon okolina), postoji ulaz $\mathbf{x} + \mathbf{r}$ takav da je $\|\mathbf{r}\| < \epsilon$ koji će također imati veliku vjerojatnost pripadanja ispravnom klasifikacijskom razredu na izlazu mreže. Slabo vidljive promjene u pravilu ne mijenjaju drastično izlaz mreže, što se može i pokazati dodavanjem šuma na neku ulaznu sliku. Dapače, neke mreže su nasumičnom deformacijom ulaza pri treniranju povećavale robusnost modela. Pokazalo se da svojstvo lokalne generalizacije zapravo u velikoj mjeri nije prisutno kod tadašnjih (a i današnjih) modela dubokog učenja i da je moguće osmisiliti optimizacijski proces koji će pronaći primjere sa slabo vidljivim promjenama koje ipak drastično mijenjaju izlaz mreže i prisile model na pogrešnu klasifikaciju. Dodatno, spomenuti oblik treniranja deformiranjem ulaza nije nimalo osporavao traženje neprijateljskih primjera.

Slijedi formalni opis optimizacijskog problema koji je potrebno riješiti.

Klasifikator koji na ulazu prima sliku, a na izlazu daje pripadnu labelu označen je s $f : \mathbb{R}^m \rightarrow \{1\dots k\}$. Pripadajuća funkcija gubitka definirana je s $loss_f : \mathbb{R}^m \times \{1\dots k\} \rightarrow \mathbb{R}^+$. Za neku sliku $\mathbf{x} \in \mathbb{R}^m$ i neku labelu $l \in \{1\dots k\}$, potrebno je riješiti sljedeći optimizacijski problem:

- minimizirati $\|\mathbf{r}\|_2$ uz ograničenja
 - $f(\mathbf{x} + \mathbf{r}) = l$
 - $\mathbf{x} + \mathbf{r} \in [0, 1]^m$

Problem postaje netrivijalan za $f(\mathbf{x}) \neq l$ i traženje egzaktnog \mathbf{r} je težak problem. Autori su riješili približni problem:

- minimizirati $c|\mathbf{r}| + loss_f(\mathbf{x} + \mathbf{r}, l)$ uz ograničenje $\mathbf{x} + \mathbf{r} \in [0, 1]^m$

Korištenjem iterativnog optimizacijskog algoritma L-BFGS (engl. *Limited memory Broyden–Fletcher–Goldfarb–Shanno algorithm*) u svakoj iteraciji se minimizira $loss_f(\mathbf{x} + \mathbf{r}, l)$ te se dodatno linijskim pretraživanjem odredi i minimalni c za koji je izlaz takav da je klasifikacija pogrešna. Za rad algoritma L-BFGS potrebno je unaprijed poznati i vrijednost gradijenta funkcije koja se optimizira, što ovaj napad čini napadom bijele kutije.

Jedno od ponuđenih objašnjenja postojanja neprijateljskih primjera je to što su konvolucijske mreže vrlo nelinearne po prirodi. To je zapravo poželjno svojstvo dubokih mreža, jer nelinearnost omogućuje rješavanje vrlo nelinearnih optimizacijskih problema kao što je klasifikacija slika. No čini se da je upravo zbog toliko visoke nelinearnosti lako za pronaći neprijateljske primjere koji su se sakrili u “džepovima” u blizini nekog ulaza, koje je vrlo teško pronaći nasumičnim pretraživanjem.

3.3. Brza metoda temeljena na gradijentima

Iako su neprijateljski primjeri otkriveni već krajem 2013., idući bitni rad na temu neprijateljskih primjera pojavio se tek 2015. godine^[13]. Taj rad je direktni nastavak na prethodni te nudi nove načine generiranja neprijateljskih načina, jedno novo bitno i zanimljivo svojstvo neprijateljskih primjera te potpuno novo i neočekivano objašnjenje postojanja neprijateljskih primjera.

Neprijateljski primjeri su se originalno pojavili pod pretpostavkom da su duboki modeli previše nelinearni te je njihovo postojanje objašnjeno teorijom da modeli imaju “slijedeće pjege” u kojima se neprijateljski primjeri teško pronalaze. Usprkos tome što je prethodno ponuđeno objašnjenje postojanja neprijateljskih primjera vrlo logično, sljedeći napad je osmišljen počevši od potpuno obrnute pretpostavke.

Ispostavilo se i da su linearne modeli podložni neprijateljskim primjerima, stoga je prvo potrebno objasniti kako je to moguće.

Digitalne slike uglavnom koriste samo 8 bitova za reprezentaciju pojedinog piksela, i svaka dodatna informacija manja od 1/255 je odbačena. Razumno je za očekivati da klasifikator nema različit izlaz za \mathbf{x} i $\tilde{\mathbf{x}} = \mathbf{x} + \boldsymbol{\eta}$, ako je svaki element perturbacije $\boldsymbol{\eta}$ manji od spomenute preciznosti. Formalnije, klasifikator bi trebao dati isti izlaz za \mathbf{x} i $\tilde{\mathbf{x}}$ dokle god je $\|\boldsymbol{\eta}\|_\infty < \epsilon$, gdje je ϵ dovoljno mal da bude odbačen.

Skalarni umnožak vektora težina \mathbf{w}^T i primjera s perturbacijom $\tilde{\mathbf{x}}$ može se raspisati ovako:

$$\mathbf{w}^T \tilde{\mathbf{x}} = \mathbf{w}^T \mathbf{x} + \mathbf{w}^T \boldsymbol{\eta} \quad (3.1)$$

Dakle, perturbacija $\boldsymbol{\eta}$ uzrokuje porast aktivacije za $\mathbf{w}^T \boldsymbol{\eta}$. Ovaj porast se može maksimizirati postavljanjem $\boldsymbol{\eta} = \epsilon \operatorname{sign}(\mathbf{w})$. Ako je prosječna vrijednost vektora težina m , onda je porast iznosa ϵmn . Norma $\|\boldsymbol{\eta}\|_\infty$ ne raste s porastom dimenzionalnosti problema, dok porast aktivacije raste linearno s n . Dakle, za visoko dimenzionalne probleme moguće je dodati nevidljive promjene koje onda zajedno mogu drastično promijeniti izlaz.

Ideja je zato napasti klasifikator “gdje najviše боли”. Potrebno je maksimizirati promjenu izlaza uz minimalnu promjenu svakog pojedinačnog elementa. Za nelinearne modele, ideja je identična:

ako su $\boldsymbol{\theta}$ parametri modela, \mathbf{x} ulaz, y izlaz te $J(\boldsymbol{\theta}, \mathbf{x}, y)$ funkcija gubitka korištena za treniranje mreže, maksimalna perturbacija za koju je uvjet norme zadovoljen je:

$$\boldsymbol{\eta} = \epsilon \operatorname{sign}(\nabla_x J(\boldsymbol{\theta}, \mathbf{x}, y)) \quad (3.2)$$

Ova metoda generiranja neprijateljskih primjera se zove metoda temeljena na predznaku gradijenta (engl. *fast gradient sign method*). Metoda je brza jer je potreban samo jedan pristup mreži, i također je napad na bijelu kutiju. Činjenica da je nelinearne klasifikatore moguće napasti s istom pretpostavkom kao i linearne, te da su nelinearne modeli jednako podložni neprijateljskim napadima kao i linearni modeli dodatno dokazuje da problem nije to što su duboki modeli previše nelinearni, nego to da su previše linearni. Na sličan način se može dobiti maksimalna perturbacija pod uvjetima normi 1 i 2. Ne uzme se funkcija predznaka sign nego je potrebno gradijent podijeliti s određenim faktorom koji osigurava da uvjet norme ostane zadovoljen. Ovako proširena skupina neprijateljskih napada se zove brza metoda temeljena na gradijentima (engl. *fast gradient method*).



Slika 3.2: FGSM napad za $\epsilon \in \{1, 5, 10\}$. Torba je predviđena kao nogometna lopta (99.87%, 71.24%, 48.63%), orao kao zmaj (99.76%, 99.96%, 98.98%), i tržnica kao banana (99.79%, 99.99%, 99.99%).

Na slici 3.3 se nalazi primjer FGSM napada. Već za $\epsilon > 2$ FGSM uspješno nalazi neprijateljski primjer za 14/16 slika iz skupa podataka 8.1. FGSM ne nađe neprijateljski primjer za slike 8.1a i 8.11 za razumni ϵ . Dodatno je zanimljivo kako kad mreža pogriješi, greška je s vrlo visokom vjerojatnošću.

3.4. DeepFool

Algoritmi FGM uspješno nalaze neprijateljske primjere iz jednog pokušaja. Očito je da bi neka iterativna metoda sigurno pronašla neprijateljske primjere s još

manjim perturbacijama.

2016. godine se pojavljuje sljedeći bitan algoritam stvaranja neprijateljskih primjera: DeepFool.

DeepFool je iterativni algoritam baziran na modelu bijele kutije. Kao i FGM, DeepFool iskorištava pretpostavljeni svojstvo prevelike linearnosti modela te u svakom koraku aproksimira nelinearni klasifikator na linearan način. Slijedi opis rada DeepFool algoritma na binarnom klasifikatoru.

Za neki klasifikator f , izlazna labela za neki ulaz \mathbf{x} označena je s $\hat{k}(\mathbf{x})$. Minimalna perturbacija \mathbf{r} je ona koja je dovoljna da promijeni vrijednost izlaza klasifikatora:

$$\Delta(\mathbf{x}; \hat{k}) := \min_{\mathbf{r}} \|\mathbf{r}\|_2 \text{ uz uvjet } \hat{k}(\mathbf{x} + \mathbf{r}) \neq \hat{k}(\mathbf{x}) \quad (3.3)$$

Minimalna perturbacija potrebna za pogrešnu klasifikaciju nekog uzorka se također naziva i robusnost \hat{k} u točki \mathbf{x} . Usput se može i definirati robusnost cijelog modela:

$$\rho_{\text{adv}}(\hat{k}) = \mathbb{E}_{\mathbf{x}} \frac{\Delta(\mathbf{x}; \hat{k})}{\|\mathbf{x}\|_2} \quad (3.4)$$

Robusnost nekog klasifikatora definirana je kao očekivana potrebna perturbacija za stvaranje neprijateljskog primjera preko (nekog) cijelog skupa podataka.

Za binarni klasifikator $f(\mathbf{x})$ se pretpostavlja da vrijedi $\hat{k}(\mathbf{x}) = \text{sign}(f(\mathbf{x}))$. Dodatno se definira skup $\mathcal{F} := \{\mathbf{x} : f(\mathbf{x}) = 0\}$ – skup vektora \mathbf{x} za koje je izlaz klasifikatora 0.

Za klasifikator oblika $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$ robusnost u točki \mathbf{x}_0 je jednaka udaljenosti od \mathbf{x}_0 do hiperravnine definirane s $\mathcal{F} = \{\mathbf{x} : \mathbf{w}^T \mathbf{x} + b = 0\}$. Ovo je prikazano na slici 3.3a.

Prema tome, da bi se klasifikacija promijenila, potrebna je perturbacija koja će točku $\mathbf{x} = \mathbf{x}_0 + \mathbf{r}$ staviti na drugu stranu hiperravnine. Minimalna takva perturbacija jednaka je ortogonalnoj projekciji \mathbf{x}_0 na ravninu \mathcal{F} . Ova projekcija može se izračunati u zatvorenom obliku:

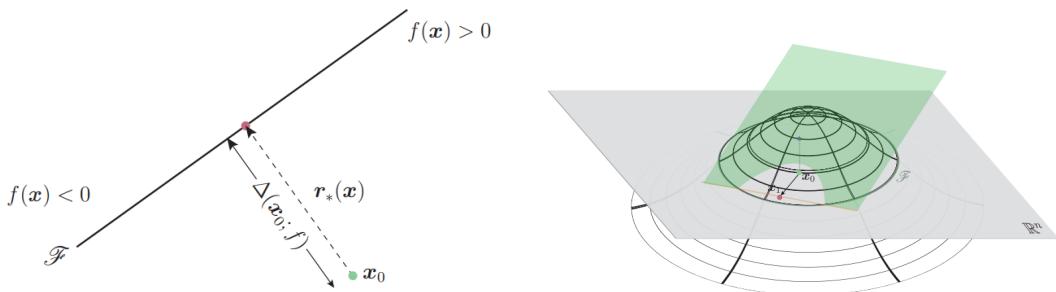
$$\mathbf{r}(\mathbf{x}_0) := -\frac{f(\mathbf{x}_0)}{\|\mathbf{w}\|_2^2} \mathbf{w} \quad (3.5)$$

U generalnom slučaju za bilo kakav diferencijabilni klasifikator ne može se vrijednost perturbacije izračunati u zatvorenom obliku. Ovdje se ponovno iskorištava svojstvo klasifikatora da su previše linearni, te se u svakoj iteraciji klasifikator f linearizira u točki \mathbf{x}_i . Minimalna perturbacija takvog linearног klasifikatora u koraku i računa se na sljedeći način:

$$\arg \min_{\mathbf{r}_i} \|\mathbf{r}_i\|_2 \text{ uz uvjet } f(\mathbf{x}_i) + \nabla f(\mathbf{x}_i)^T \mathbf{r}_i = 0 \quad (3.6)$$

Izraz uvjeta predstavlja tangencijalnu hiperravninu na funkciju klasifikatora u točki \mathbf{x}_i . Algoritam opisan riječima bi glasio ovako: u svakom koraku algoritma se za trenutnu točku napravi linearna aproksimacija klasifikatora, zatim se napravi ortogonalna projekcija trenutne točke na sjecište tangencijalne hiperravnine i \mathbb{R}^n . Algoritam se ponavlja dokle god klasifikator ne pogriješi, odnosno dokle god točka ne dođe na granicu klasifikatora. Ilustracija iz rada prikana na slici 3.3b vizualno opisuje jedan korak algoritma. Zanimljivo je odmah uočiti kako je često i potreban samo jedan korak algoritma.

Kako se višerazredni klasifikator može promatrati kao skupina binarnih klasifikatora, algoritam je moguće poopćiti na višerazredne klasifikatore. Poopćenje na višerazredne diferencijabilne klasifikatore može se naći u izvornom radu^[14].



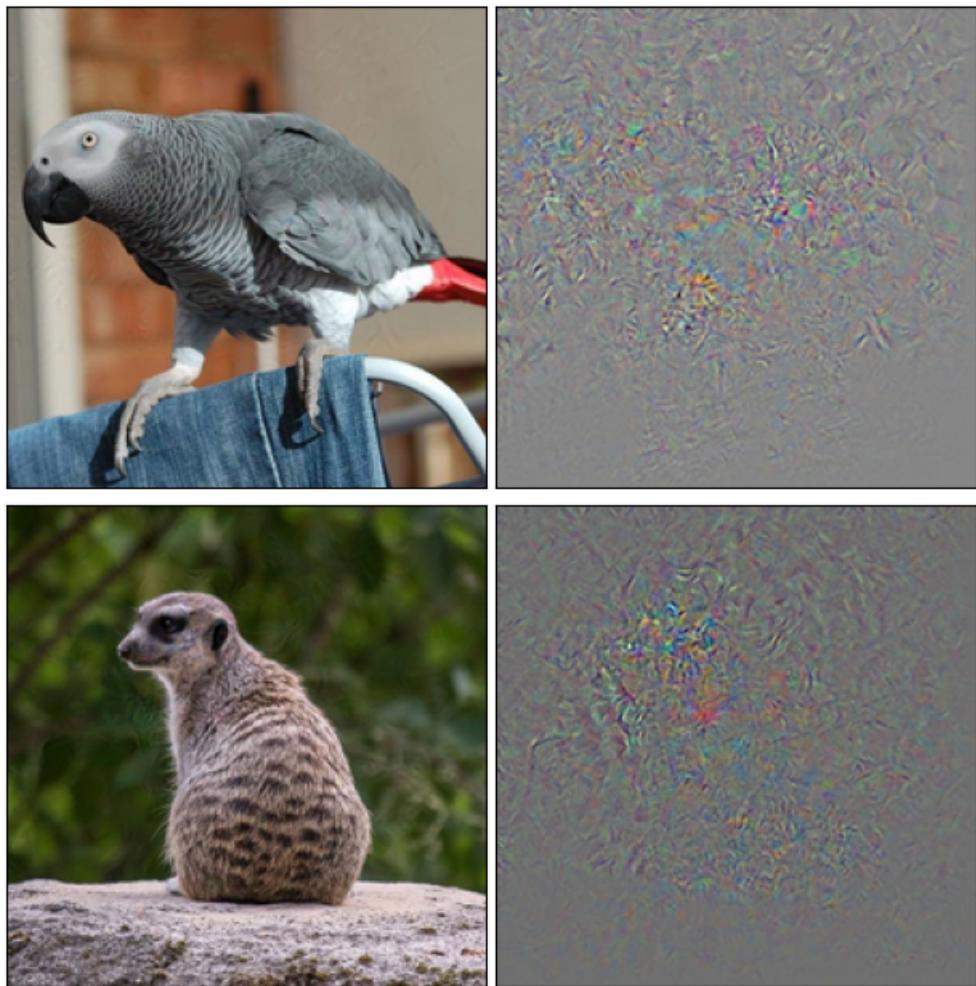
(a) Linearni binarni klasifikator. Minimalna perturbacija potrebna za promijeniti izlaz klasifikatora je ortogonalna projekcija na pravac koji dijeli razrede.

(b) Diferencijabilni binarni klasifikator. U svakom koraku se za trenutnu točku aproksimira funkcija klasifikatora s hiperravninom i napravi projekcija na pravac koji je rezultat presjeka dobivene hiperravnine i \mathbb{R}^n (označen narančastom bojom).

Slika 3.3: Ilustracija rada DeepFool algoritma na binarnim klasifikatorima.

Pošto je i za rad ovog algoritma potrebno znanje o mreži, ovo je također napad na model bijele kutije. Za razliku od do sada opisanog FGSM napada, DeepFool pronalazi bolje neprijateljske primjere s daleko manjim perturbacijama.

Što se tiče uspješnosti DeepFool napada na skup 8.1, napad je 100% uspješan na svim slikama. Broj iteracija potreban je također iznenadjuće nizak – za čak sedam slika je potrebna samo jedna iteracija algoritma. Za FGSM napad su se pokazale najzahtjevnije slike 8.1a i 8.1l. Te dvije slike su u slučaju DeepFoola zahtijevale 6 i 8 iteracija. U slici 3.4 se nalazi prikaz napada na te slike.



Slika 3.4: Rezultati DeepFool napada za slike koje FGSM nije mogao pretvoriti u neprijateljske. U lijevom stupcu je neprijateljska slika, a u desnom je razlika između originalne i neprijateljske slike. Izlaz modela za prvu sliku je *prepelica* (quail, 99.46%), a za drugu sliku *šumski kunić* (wood rabbit, 7.8% – moguće je povećati vjerojatnost uz podešavanje hiperparametara napada)

4. Obrana dubokih konvolucijskih modela I

Do sada je predstavljeno nekoliko napada. Originalni napad temeljen na minimizaciji približnog problema uz pomoć L-BFGS algoritam je opisan zbog povijesnih razloga. Napad nije u primjeni jer je u praksi vrlo spor i ne daje dobre rezultate. Međutim, pojavilo se mnoštvo potencijalnih obrana od dosadašnjih napada. U ovom poglavlju su opisane tri glavne kategorije takvih obrana na primjeru pojedinačnih obrana:

- Jednostavne obrane – konceptualno vrlo jednostavne za shvatiti, nije potrebno duboko predznanje o napadima, “jeftine” za implementirati
- Neprijateljsko treniranje – osnovna ideja je poboljšanje robusnosti klasifikatora još pri treniranju
- Prikrivanje gradijenata – vođeni idejom povećanja nelinearnosti mreže, određeni napadi namjerno ili slučajno “prikrivaju” gradijente mreže i time prividno sprječavaju napade koji se temelje na gradijentima

4.1. Jednostavne obrane

U ovom dijelu je ukratko opisano nekoliko vrlo jednostavnih obrana. Dobra strana ovih obrana je što su iznimno jednostavne za implementirati i vrlo efektivne protiv već stvorenih neprijateljskih primjera, te nije potrebno nikakvo mijenjanje samog modela niti detaljno poznavanje potencijalnih metoda napada. Negativno je to što su pre-jednostavne da bi spriječile stvaranje novih neprijateljskih primjera te ne povećavaju robusnost mreže. Općenito, obrane koje su “agnostičke” što se tiče modela kojeg brane, odnosno nezavisne su od modela, su u pravilu neuspješne.

4.1.1. JPEG kompresija

JPEG kompresija u obliku obrane od neprijateljskih primjera se pojavila više puta u različitim oblicima^[15] [16] [17], a slijedi opis najjednostavnijeg načina zaštite.

JPEG je vrsta sažimanja podataka s gubitkom (engl. *lossy*) za slike. Upravo to svojstvo je poželjno pri uništavanju neprijateljske perturbacije. Prepostavka je da su i sami neprijateljski primjeri osjetljivi na perturbacije, odnosno da će male promjene nad pažljivo-konstruiranom perturbacijom vratiti sliku nazad u područje ispravne klasifikacije. Ovo dodatno ima smisla kada se uzme u obzir da su neprijateljski primjeri rijetki, to jest da ih nije moguće lako nasumično pronaći. JPEG kompresija bazira se na diskretnoj kosinusnoj transformaciji (engl. *discrete cosine transform*, DCT). Transformacijom iz prostorne domene u frekvencijsku domenu omogućava se direktna manipulacija frekvencijskom domenom. Ljudski vid nije toliko osjetljiv na komponente visoke frekvencije u slikama, stoga se provodi kvantizacija frekvencija te se visoke frekvencije čuvaju s manjom preciznošću nego niske frekvencije. Preciznost pri kojoj se čuvaju visoke frekvencije ovisi o faktoru kvalitete koji je u rasponu od 0 do 100: za 0 se visoke frekvencije potpuno odbacuju, a za 100 visoke frekvencije su maksimalno očuvane. Važno je uočiti da kvaliteta od 100 ne znači da je kompresija bez gubitka, jer se gubitak djelomično dogodi već pri prelasku u frekvencijsku domenu. Na slici 4.1 se nalazi usporedba kompresije za različite kvalitete.



Slika 4.1: Primjeri JPEG kompresije za različiti parametar kvalitete. Kvaliteta je redom 5, 25, 50, 90.

Najjednostavnija verzija obrane (i prva koja se pojavila) je korištenje JPEG kompresije samo pri evaluaciji. Autori su isprobali FGSM napad uz $\epsilon \in \{1, 5, 10\}$. Koraci su sljedeći:

- izračunati neprijateljske primjere za navedene ϵ
- provesti JPEG kompresiju nad običnim slikama i neprijateljskim primjerima
- usporediti izlaz modela za sve navedene skupove

Dodatno bi bilo zanimljivo vidjeti kako obrana utječe na DeepFool napad, pošto DeepFool generira puno manje perturbacije. Rezultati u tablici 4.1 su dobiveni na 16 slika iz 8.1.

Ulaz	Kompresija	Točnost
Standardno	Bez kompresije	100.00%
	Kvaliteta 50%	100.00%
	Kvaliteta 75%	100.00%
FGSM $\epsilon = 1$	Bez kompresije	31.25%
	Kvaliteta 50%	68.75%
	Kvaliteta 75%	43.75%
FGSM $\epsilon = 5$	Bez kompresije	18.75%
	Kvaliteta 50%	12.50%
	Kvaliteta 75%	12.50%
FGSM $\epsilon = 10$	Bez kompresije	12.50%
	Kvaliteta 50%	18.75%
	Kvaliteta 75%	12.50%
DeepFool	Bez kompresije	0.00%
	Kvaliteta 50%	81.25%
	Kvaliteta 75%	64.50%

Tablica 4.1: Utjecaj JPEG kompresije na različite neprijateljske primjere. JPEG kompresija u ovom obliku jedino može pokvariti napade male magnitude, no čak i tada nije uvijek uspješna.

Zaključak je dakle da je jednostavna obrana temeljena na JPEG kompresiji daleko od korisnog rješenja, bar za FGSM napad. U slučaju DeepFool napada, obrana je puno uspješnija, međutim i tada nije u mogućnosti dosegnuti 100% točnost kao na čistim ulazima, te čak i napad s iznimno malom perturbacijom može zaobići JPEG kompresiju. Dodatno je problematično to što metoda nije u mogućnosti spriječiti nove napade, samo degradirati već postojeće neprijateljske primjere. Dapače, ponovi li se DeepFool napad uz JPEG kompresiju na ulazu, napad je opet uspješan 100% vremena na ovom skupu podataka.

Postoje i sofisticiraniji oblici JPEG kompresije kao metode obrane^[16] od neprijateljskih primjera. Moguće je dodatno mrežu trenirati na slikama različite

kvalitete kompresije kako bi mreža mogla uspješno klasificirati i slike lošije kvalitete (koje često imaju artefakte). Ovaj proces autori nazivaju “cijepljenjem” mreže. Takoder je moguće imati onoliko modela koliko i različitih kvaliteta JPEG slika i konstruirati ansambl takvih modela. Međutim, u pravilu su obrane temeljene na ansamblima jake onoliko koliko i najjača komponenta ansambla, a pošto JPEG kompresija nije jaka obrana takav ansambl isto nije vrlo jak pri obrani od neprijateljskih primjera.

4.1.2. Stiskanje značajki

Stiskanje značajki^[18] (engl. *feature squeezing*) je generalni pojam koji se može definirati neovisno o neprijateljskim primjerima. Prostori u kojima se nalaze značajke su često bespotrebno veliki te je ideja smanjiti stupnjeve slobode pojedinih značajki i tako “istisnuti” manje bitne značajke. JPEG kompresija se isto može smatrati oblikom stiskanja značajki, no u kontekstu neprijateljskih primjera se termin odnosi na dvije metode: redukcija dubine boje slike i zagladivanje slike.

Dva najčešće korištена formata boje slika za klasifikaciju su RGB (npr. *CIFAR-10*, *ImageNet*) i sivi tonovi (engl. *grayscale*, npr. *MNIST*). Za RGB slike, svaki piksel je reprezentiran s $3 \times 8 = 24$ bita, što daje 2^{24} mogućih vrijednosti pojedinog piksela. Međutim, za klasifikaciju slika, nije potrebno imati toliko precizne informacije te ljudi mogu lako prepoznati što se na slici nalazi i s manjom dubinom boje. Smanjenjem broja dostupnih bitova se može pokvariti neprijateljska perturbacija i također u teoriji smanjiti prostor gdje se neprijateljski primjeri mogu nalaziti. Na slici 4.2 je primjer slike s različitim brojem bitova dostupnim za prikaz boje. Korišten broj bitova u originalnom radu je 4 i 5.



Slika 4.2: Ista slika s različitim brojem bitova za boju. Broj bitova je redom 8, 6, 4, 2 i 1.

Gaussovo zaglađivanje je proces pri kojemu se slika zamućuje do određene mjere. Kao i kod prethodnih metoda, zamućenje može uništiti pažljivo stvorene perturbacije. Na slici 4.3 se nalazi primjer Gaussovog zamućivanja slike.



Slika 4.3: Zamućenje slike uz veličine prozora 2×2 , 3×3 i 4×4

Obje metode su poprilično neuspješne u uništavanju FGSM perturbacije jer je FGSM napad veće magnitude, dok uspješno uništava primjere drugih, jačih napada (npr. DeepFool) koji generiraju manje perturbacije. Ovo je slično kao i kod obrane uz JPEG kompresiju koja je opisana u potpoglavlju 4.1.1. Međutim, kao što se kasnije ispostavilo^[19], obje obrane je vrlo lako zaobići. Potrebno je jednostavno povećati snagu pojedinih napada da bi se obrane zaobišle, a novonastala perturbacija je također nevidljiva. Postoji i jednostavno objašnjenje zašto Gaussovo zaglađivanje sigurno ne može biti uspješna obrana: zaglađivanje se provodi operacijom konvolucije, što se zapravo može promatrati kao dodatni konvolucijski sloj na ulazu mreže, a konvolucijske mreže su ionako slabe na napade. Stoga još jedan dodatni konvolucijski sloj (s fiksnim težinama) sigurno ne može puno toga napraviti da spriječi nastanak neprijateljskih primjera. Dodatno je zanimljivo i kako se zaglađivanje može koristiti i za generiranje neprijateljskih primjera: dovoljno je linijskim pretraživanjem pronaći najmanji faktor zaglađivanja za koji neka mreža pogriješi. Iznenađujuće je da za puno slika, distorzija potrebna za pogrešnu klasifikaciju nije velika. Međutim, ovo se može poboljšati tako da se pri treniranju uvedu i zamućene slike (engl. *data augmentation*).

4.2. Neprijateljsko treniranje - FGSM

Neprijateljsko treniranje se kao ideja pojavila zajedno s FGSM napadom^[13]. Međutim, neprijateljsko treniranje je tada zamišljeno primarno kao nova metoda regularizacije modela i autori su usporedivali neprijateljsko treniranje s drugim metodama regularizacije (npr. *dropout* i *pretraining*), a ne kao obranu od potencijalnih neprijateljskih napada.

Neprijateljsko treniranje se fundamentalno razlikuje od dosadašnjih metoda povećanja skupa podataka (engl. *data augmentation*). Standardne metode uključuju

operacije kao što su rotacija, zrcaljenje, blago mijenjanje boja, brisanje dijelova slike – no tako transformirane slike i dalje ostaju u originalnoj distribuciji podataka. Zapravo je većina operacija i odabrana upravo iz tog razloga, jer se takve slike očekuju i u skupu podataka za testiranje. No neprijateljsko treniranje trenera model na primjerima koji se nikad ne bi pojavili u skupu za treniranje – neprijateljski primjeri se ne pojavljuju prirodno nego trebaju biti konstruirani.

Najjjednostavniji oblik neprijateljskog treniranja je sljedeći:

- u svakom koraku treniranja se skup treniranja podijeli u dva skupa
- jedan skup ostane netaknut
- drugi skup se pretvori u neprijateljske primjere – ovo se provodi samo za ulaze koji su ispravno klasificirani

Omjer skupova je hiperparametar. U originalnom su radu autori odabrali omjer 1 : 1 koji radi dovoljno dobro, stoga je i ovdje u nastavku korišten isti omjer.

Za model je ovdje korišten konvolucijski model opisan u potpoglavlju 2.4. Model treniran na standardan način nakon 25 epoha postigne točnost od 78.22%, a nakon 50 epoha postigne točnost od 79.80%. Za usporedbu, istreniran je model neprijateljskim treniranjem uz FGSM s $\epsilon = 0.1$ te model uz FGSM s $\epsilon \in \{0.05, 0.1, 0.2, 0.3\}$. U tablici 4.2 se nalaze rezultati uspješnosti FGSM napada uz $\epsilon \in \{0.05, 0.1, 0.2\}$ na različito trenirane modele.

Treniranje	Epoha	Čisti podatci	FGSM $\epsilon = 0.05$	FGSM $\epsilon = 0.1$	FGSM $\epsilon = 0.2$
Standardno	25	78.22%	19.89%	16.99%	16.22%
	50	79.80%	22.05%	16.85%	12.32%
FGSM $\epsilon = 0.1$	25	73.32%	44.33%	66.53%	23.24%
	50	74.96%	45.89%	66.14%	68.33%
FGSM $\epsilon = 0.2$	25	74.04%	15.22%	24.25%	69.71%
	50	75.77%	16.80%	34.47%	71.11%
FGSM $\epsilon = 0.3$	25	74.23%	15.02%	18.88%	56.67%
	50	77.76%	17.12%	18.30%	58.76%
FGSM $\epsilon \in \{0.05, 0.1, 0.2, 0.3\}$	25	77.54%	43.98%	73.52%	30.14%
	50	76.98%	66.82%	69.88%	69.87%

Tablica 4.2: Prikazana je točnost modela na čistim podatcima i uz FGSM za različite ϵ . Neprijateljsko FGSM treniranje ne generalizira preko različitih epsilona, te broj epoha igra bitnu ulogu kod većeg broja napada korištenih pri treniranju.

Neprijateljsko treniranje se čini kao korak u dobrom smjeru, ali ne u ovom obliku. Područje koje je potrebno pokriti je preveliko jer svakako postoji više neprijateljskih primjera nego legitimnih ulaza i jednostavno nije izvodljivo model trenirati na svim normama svakog napada. Bolji oblik neprijateljskog treniranja koji ipak može generalizirati nad različitim normama se dodatno razmatra u potpoglavlju 6.2 s obećavajućim rezultatima.

4.3. Termometar kodiranje

Termometar kodiranje (engl. *thermometer encoding*) je obrana koja je direktno napala problem linearnosti dubokih modela^[20]. Slično kao i prethodne obrane, nije potrebno puno promijeniti postojeći model, ali obrana svejedno nije besplatna kao jednostavnije obrane.

Jedan način da se poveća nelinearnost bi bila mijenjanje aktivacijskih funkcija mreža, no ReLU i ostale aktivacijske funkcije se koriste s razlogom: brze su i jednostavne za optimirati. Druga metoda bi bila da se postavi vrlo nelinearna transformacija na ulaz mreže. Na drugoj metodi se temelji termometar kodiranje.

Prvo je potrebno uvesti kvantizacijsku funkciju b . Potrebno je odabrati vrijednosti b_i takve da vrijedi $0 < b_1 < b_2 < \dots < b_k = 1$, npr. $b_i = \frac{i}{k}$. Za neki realni broj $\theta \in [0, 1]$ definira se funkcija $b(\theta)$ kao najmanji indeks $\alpha \in \{1, \dots, k\}$ takav da je $\theta \leq b_\alpha$.

Slijedi opis *one-hot* kodiranja. Za neki indeks $j \in \{1, \dots, k\}$, neka je $\chi(j) \in \mathbb{R}^k$ *one-hot* vektor od j :

$$\chi(j)_l = \begin{cases} 1 & \text{ako } l = j \\ 0 & \text{inače} \end{cases} \quad (4.1)$$

Diskretizacijska funkcija za neki piksel je onda:

$$f_{\text{onehot}}(x_i) = \chi(b(x_i)) \quad (4.2)$$

Međutim, *one-hot* kodiranje se nije pokazalo dobrom transformacijom ulaza za sprječavanje neprijateljskih primjera. Pretpostavka je da je zbog toga što se gubi svojstvo uređenosti između piksela, odnosno za svaku *one-hot* reprezentaciju vrijedi:

$$\|\chi(b(x_i))\|_2 = \|\chi(b(x_j))\|_2 = 1 \text{ kada vrijedi } b(x_i) \neq b(x_j) \quad (4.3)$$

Termometar kodiranje je vrlo slično *one-hot* kodiranju, no ne gubi se svojstvo uređenosti. Za neki indeks $j \in \{1, \dots, k\}$, neka je $\tau(j) \in \mathbb{R}^k$ termometar vektor od j definiran kao

$$\tau(j)_l = \begin{cases} 1 & \text{ako } l \geq j \\ 0 & \text{inače} \end{cases} \quad (4.4)$$

Diskretizacijska funkcija za neki piksel je onda:

$$f_{\text{therm}}(x_i) = \tau(b(x_i)) \quad (4.5)$$

Za svaku termometar reprezentaciju vrijedi:

$$\|\tau(b(x_i))\|_2 < \|\tau(b(x_j))\|_2 = 1 \text{ kada vrijedi } b(x_i) \neq b(x_j) \text{ i } x_i < x_j \quad (4.6)$$

U tablici 4.3 je nekoliko primjera *one-hot* i termometar kodiranja.

Ulaz	One hot	Termometar
0.03	[1000000000]	[1111111111]
0.54	[0000100000]	[0000111111]
0.78	[0000000100]	[0000000111]
0.92	[0000000001]	[0000000001]

Tablica 4.3: Primjer kodiranja za $b_i = \frac{i}{k}$ uz $k = 10$.

Da bi se neka mreža mogla koristiti s termometar kodiranjem, potrebno ju je ponovno istrenirati uz transformaciju ulaza. Nakon 30 epoha treniranja, mreža na čistim ulazima postiže točnost od 77.13%. U ovom obliku, mrežu nije više moguće napast uspješno s niti jednim od do sada spomenutih napada na model bijele kutije. Razlog tome je što nije moguće obaviti propagaciju unatrag kroz diskretizacijsku funkciju na ulazu. Svi napadi bijele kutije u standardnom formatu imaju uspješnost od približno 0%, ako ne uzimaju u obzir transformaciju na ulazu.

Autori su zato osmislili dva nova iterativna napada na model bijele kutije koje evaluiraju na istreniranim modelima, te su tako pokazali uspješnost svoje obrane. Ova obrana je bila *state-of-the-art* obrana u trenutku kada se pojavila krajem 2017. godine. Međutim, ova i mnoge slične obrane koje pokušavaju diskretizacijom povećati nelinearnost mreže imaju zajednički problem koji ubrzano dolazi na vidjelo.

4.4. Obrambena destilacija

Obrambena destilacija (engl. *defensive distillation*) je učinkovita obrana koja se temelji na općenitijem pojmu destilacije dubokih neuronskih mreža. Destilacija je oblik treniranja mreže koji se provodi u dva koraka. U prvom koraku se standardno trenira neka mreža, no na ulaz u *softmax* se vrijednosti podijele s faktorom T koji se naziva temperatura. Izlaz *softmax* sloja se onda računa prema izrazu u jednadžbi 4.7. Temperatura T igra veliku ulogu u izlazu mreže – veća temperatura pridaje veću vjerojatnost svim izlazima, i za $T \rightarrow \infty$ vjerojatnost za sve razrede teži k $\frac{1}{K}$.

$$\sigma(\mathbf{x})_i = \frac{e^{\mathbf{x}_i/T}}{\sum_{j=1}^K e^{\mathbf{x}_j/T}} \text{ za } i = 1, \dots, K \text{ i } \mathbf{x} \in \mathbb{R}^K \quad (4.7)$$

Nakon treniranja prve mreže se trenira druga mreža na isti način, ali se u skupu podataka za treniranje labele zamijene s izlazom već istrenirane mreže. Stoga se standardne *one-hot* kodirane labele mijenjaju s vjerojatnostima koje je vratila prethodno istrenirana mreža. Ponovno se ponovi postupak treniranja uz temperaturu T . Nakon treniranja, za vrijeme testiranja, temperatura T se ukloni, odnosno postavi na $T = 1$. Rezultat je destilirana mreža.

U originalnoj formulaciji destilacije mreže, prva mreža je kompleksnija i s većim kapacitetom, dok je druga mreža jednostavnija. Učenje jednostavnije mreže na vjerojatnostima uz temperaturu T se pokazalo korisnim jer jednostavnija mreža uspije postići rezultate neke složenije mreže, a k tome je i brža pri računanju izlaza.

U kontekstu obrambene destilacije, obje mreže mogu biti iste. Temperatura je najbitniji parametar, a autori su pokazali da je dobra temperatura $T > 10$. Najbolji rezultati postignuti su za $T = 100$. U tablici 4.4 su prikazani rezultati različitih FGSM napada te DeepFool napada na destilirane mreže uz temperature $T \in \{50, 100\}$ i rezultati napada na čistu mrežu. Korištena mreža je ponovno *CIFAR-10* mreža opisana u potpoglavlju 2.4, a treniranje je trajalo 50 epoha. Mreža trenirana uz $T = 1$ nije destilirana, tu je provedeno standardno treniranje.

Temperatura	Čisti podatci	FGSM	FGSM	FGSM	DeepFool
		$\epsilon = 2$	$\epsilon = 5$	$\epsilon = 10$	
$T = 1$	78.89%	51.47%	34.87%	27.99%	16.80%
$T = 50$	78.09%	77.50%	77.46%	76.15%	77.40%
$T = 100$	78.41%	78.15%	78.09%	77.73%	76.40%

Tablica 4.4: Točnost za obrambeno destilirane mreže uz različite napade. Mreža za $T = 1$ je trenirana standardno. Napadi imaju iznimno loše rezultate na ovako treniranim mrežama. DeepFool napad je testiran na samo 500 primjera.

I ova obrana se pokazala iznimno uspješnom na napade na model bijele kućice. Za razliku od termometar kodiranja, ovdje je moguće provesti propagaciju unazad, a rezultati su svejedno izvrsni. Nažalost, kao i sve ranije opisane obrane, i ova obrana ima prikrivenu manu.

5. Neprijateljski primjeri II

5.1. Neučinkovitost obrana

Obrane opisane do sada su se sve pokazale neučinkovitim. Jednostavne transformacije ulaza redovito nisu dovoljne da pokvare perturbacije, a čak i kad jesu, pokazalo se da je moguće konstruirati bolje perturbacije koje su otporne na različite transformacije kao što su JPEG kompresija, zaglađivanje i redukcija dubine boje. Neprijateljsko treniranje uz FGSM nije u mogućnosti generalizirati za različite vrijednosti ϵ , iako je korak u dobrom smjeru. No obrane kao što su termometar kodiranje i defenzivna destilacija su se pokazale posebno uspješnima.

Maskiranje gradijenata je pojava kod koje model ne sadrži korisne gradijente. Za obrane koje su dizajnirane tako da (namjerno ili slučajno) maskiraju gradijente se kaže da skrivaju gradijente (engl. *gradient obfuscation*)^[21]. Nije svaki oblik maskiranja gradijenata isto što i skrivanje gradijenata – npr. mreža može naučiti maskirati gradijente, kao što je slučaj kod mnogo oblika neprijateljskog treniranja^[22].

Oblik skrivanja gradijenata kod obrana koje: nisu diferencijabilne (kao termometar kodiranje), su numerički nestabilne, ili na bilo koji način daju neispravne gradijente naziva se *razbijanje gradijenata* (engl. *gradient shattering*)^[21]. U ovu kategoriju spadaju termometar kodiranje i defenzivna destilacija. Specifično, termometar kodiranje nije diferencijabilno i stoga se ne mogu izravno izračunati gradijenti mreže, što sprječava mnogo napada koji se na tome temelje.

Za obrambenu destilaciju je problem malo drugačiji. Temperatura T u *softmax* funkciji efektivno tjera mrežu da izlaze prethodnog sloja skalira za faktor T . Kada se pri testiranju opet uzme temperatura $T = 1$, uz velike vrijednosti prethodnog sloja, izlaz mreže postane ϵ za sve neispravne izlaze, te $1 - N \cdot \epsilon$ za ispravni izlaz. Ispostavilo se da je u većini slučajeva ϵ toliko mal, da se u 32 bitnoj aritmetici s pomicnim zarezom ta vrijednost zaokruži na 0 što posljedično i gradijente pretvori u 0 i tako spriječi napade koji se temelje na gradijentima.

Drugi česti problem kod velikog broja ranijih metoda obrane je također to što je njihova evaluacija bila slaba ili nepotpuna. Obrane bi često bile evaluirane na vrlo jednostavnim mrežama, jednostavnim skupovima podataka (*CIFAR-10* i *MNIST*), korištenjem samo najjednostavnijih napada (L-BFGS, FGSM) s nedovoljno dobrim hiperparametrima. Česti problemi evaluacije robusnosti modela su detaljnije opisani u potpoglavlju 6.1. Kao odgovor na neispravno evaluirane obrane su se pojavili jači napadi koji su korišteni za opovrgavanje uspješnosti tih obrana. U nastavku slijede dva vrlo jaka napada na model bijele kutije, te jedan zanimljivi napad na model crne kutije.

5.2. Projicirani gradijentni spust

Projicirani gradijentni spust (engl. *projected gradient descent*) se pojavio kontekstu neprijateljskog treniranja koje je detaljno opisano u potpoglavlju 6.2. Općenito, projicirani gradijentni spust je oblik gradijentnog spusta koji dodatno uzima u obzir ograničenja. Razlika je u tome što je nakon svakog pomaka po gradijentu potrebno obaviti projekciju takvu da su ograničenja zadovoljena.

FGSM napad u jednom koraku izračuna neprijateljski primjer na sljedeći način:

$$\tilde{\mathbf{x}} = \mathbf{x} + \epsilon \operatorname{sign}(\nabla_{\mathbf{x}} J(\boldsymbol{\theta}, \mathbf{x}, y)) \quad (5.1)$$

FGSM se može promatrati kao jedan korak iterativnog napada:

$$\mathbf{x}^{t+1} = \Pi_{\mathbf{x}+S}(\mathbf{x}^t + \epsilon \operatorname{sign}(\nabla_{\mathbf{x}} J(\boldsymbol{\theta}, \mathbf{x}, y))) \quad (5.2)$$

Operator $\Pi_{\mathbf{x}+S}$ predstavlja projekciju na $\mathbf{x} + S$. Skup $S \subseteq \mathbb{R}^d$ je skup dopuštenih perturbacija. $\mathbf{x} + S$ je ℓ_∞ kugla oko \mathbf{x} unutar koje se traže neprijateljski primjeri tako da minimiziraju funkciju $J(\boldsymbol{\theta}, \mathbf{x}, y)$.

Iako jednostavan, projicirani gradijentni spust je mnogo jači od FGSM i pokazuje svojstva koja ga čine boljim za neprijateljsko treniranje. Već nakon 10 iteracija uz $\epsilon = 1$ uspješno bude stvoreno 14/16 neprijateljskih primjera za skup 8.1. U tablici 5.1 su predviđanja mreže za tako stvorene primjere.

Slika	Izlaz mreže	Vjerojatnost
8.1a	African grey*	100.00%
8.1b	Soccer ball	100.00%
8.1c	Damselfly	99.94%
8.1d	Kite	99.99%
8.1e	Banana	99.99%
8.1f	Orangutan	99.99%
8.1g	Pembroke	98.08%
8.1h	Shield	99.14%
8.1i	Dungeness crab	63.76%
8.1j	Space bar	51.36%
8.1k	French bulldog	99.98%
8.1l	Meerkat*	98.92%
8.1m	Welsh springer spaniel	99.69%
8.1n	Neck brace	100.00%
8.1o	Egyptian cat	99.87%
8.1p	Shower curtain	99.99%

Tablica 5.1: Tablica top-1 izlaza mreže za neprijateljske primjere stvorene PGD algoritmom nakon 10 iteracija uz $\epsilon = 1$. Slike označene sa zvjezdicom (*) nisu uspješno pretvorene u neprijateljske primjere.

5.3. Napadi *Carlini and Wagner*

Obrambena destilacija naizgled se činila kao vrlo otporna obrana. Problem s gradijentima opisan u potpoglavlju 5.1 nije inicijalno bio uočen.

U svrhu poražavanja obrambene destilacije, autori Carlini i Wagner se vraćaju na početnu definiciju problema neprijateljskih primjera^[23]:

- minimizirati $\mathcal{D}(\mathbf{x}, \mathbf{x} + \mathbf{r})$ uz ograničenja
 - (a) $C(\mathbf{x} + \mathbf{r}) = t$
 - (b) $\mathbf{x} + \mathbf{r} \in [0, 1]^m$

S $C(\mathbf{x})$ je označena labela na izlazu mreže za ulaz \mathbf{x} . Cilj je dakle pronaći perturbaciju \mathbf{r} koja minimizira $\mathcal{D}(\mathbf{x}, \mathbf{x} + \mathbf{r})$. \mathcal{D} je ovdje neka funkcija udaljenosti, na primjer neka od normi $\ell_1, \ell_2, \ell_\infty$. Potrebno je formulirati problem tako da ga

je moguće riješiti nekim od postojećih optimizacijskih algoritama.

Definira se funkcija f tako da vrijedi $C(\mathbf{x} + \mathbf{r}) = t$ ako i samo ako $f(\mathbf{x} + \mathbf{r}) \leq 0$.

Slijedi nekoliko izbora za funkciju f :

$$\begin{aligned} f_1(\mathbf{x}') &= -\text{loss}_{F,t}(\mathbf{x}') + 1 \\ f_2(\mathbf{x}') &= (\max_{i \neq t}(F(\mathbf{x}')_i) - F(\mathbf{x}')_t)^+ \\ f_3(\mathbf{x}') &= \text{softplus}(\max_{i \neq t}(F(\mathbf{x}')_i) - F(\mathbf{x}')_t) - \log(2) \\ f_4(\mathbf{x}') &= (0.5 - F(\mathbf{x}')_t)^+ \\ f_5(\mathbf{x}') &= -\log(2F(\mathbf{x}')_t - 2) \\ f_6(\mathbf{x}') &= (\max_{i \neq t}(Z(\mathbf{x}')_i) - Z(\mathbf{x}')_t)^+ \\ f_7(\mathbf{x}') &= \text{softplus}(\max_{i \neq t}(Z(\mathbf{x}')_i) - Z(\mathbf{x}')_t) - \log(2) \end{aligned}$$

S $\text{loss}_{F,t}$ je označena funkcija gubitka unakrsne entropije. $(x)^+$ označava $\max(x, 0)$, a za softplus vrijedi $\text{softplus}(x) = \log(1 + \exp(x))$. $Z(\mathbf{x})$ označava izlaz mreže prije softmax sloja – ove vrijednosti se nazivaju *logiti*. Ovakvom zamjenom je moguće optimizacijski problem formulirati na drugačiji način:

$$\text{minimizirati } \mathcal{D}(\mathbf{x}, \mathbf{x} + \mathbf{r}) + c \cdot f(\mathbf{x} + \mathbf{r}) \text{ uz ograničenje } \mathbf{x} + \mathbf{r} \in [0, 1]^n$$

Još je potrebno razriješiti ograničenje $\mathbf{x} + \mathbf{r} \in [0, 1]^n$. U originalnom radu o neprijateljskim primjerima^[3] korišten je optimizacijski algoritam L-BFGS koji ovaj oblik ograničenja podržava. Međutim, moguće je ograničenje riješiti i na drugačiji način: zamjenom varijabli. Umjesto optimizacije varijable \mathbf{r} , na slijedeći način se uvodi nova varijabla \mathbf{w} koju je potrebno optimizirati:

$$r_i = \frac{1}{2}(\tanh(w_i) + 1) - x_i$$

Pošto $-1 \leq \tanh(w_i) \leq 1$, i dalje vrijedi $0 \leq x_i + r_i \leq 1$, pa je rješenje ovog optimizacijskog problema automatski valjano. Ovaj pristup omogućava korištenje optimizacijskih algoritama koji ne podržavaju ograničenja u prethodnom obliku. Autori su odlučili koristiti gradijentni spust uz optimizacijski algoritam Adam. Koeficijent c se traži binarnim pretraživanjem tako da se nađe najmanji c za koji vrijedi $f(\mathbf{x}^*) \leq 0$ – za premali c , optimizacija ne uspije, a za preveliki c optimizacija uspije, ali napad je manje kvalitete.

Autori provode evaluaciju navedenih 7 funkcija za različite vrijednosti c . Funkcija f_6 se iskazala kao najuspješnija, stoga je korištena pri formulaciji konačnih napada.

Carlini i Wagner L_2 napad. Kombinacijom svega navedenog, dolazi se do konačnog optimizacijskog problema. Za normu je u ovom slučaju uzeta ℓ_2 norma. Napad može biti ciljani, jer se za t može odabratи proizvoljni razred. Optimizacijski problem je onda:

$$\text{minimizirati } \|\frac{1}{2}(\tanh(\mathbf{w}) + 1) - x\|_2^2 + c \cdot f\left(\frac{1}{2}(\tanh(\mathbf{w}) + 1)\right)$$

gdje je funkcija f definirana na slijedeći način:

$$f(\mathbf{x}') = \max(\max\{Z(\mathbf{x}')_i : i \neq t\} - Z(\mathbf{x}')_t, -\kappa)$$

Parametar κ predstavlja “samouvjerenost” napada. Za veću vrijednost κ pro-nalazi se neprijateljski primjer za koji će mreža dati veći vjerojatnost. U nastavku se koristi $\kappa = 50$. Na slici 5.1 je prikazan primjer napada na *ResNet* mrežu.



Slika 5.1: Neciljani CW ℓ_2 napad. Slike su redom klasificirane kao: perzijska mačka, štit, razmak (na tipkovnici) i ovan. Prosječna promjena po pikselu je 0.40 na skali [0, 255].

Carlini i Wagner L_∞ napad. L_∞ metrika nije u potpunosti diferencijabilna pa gradijentni spust ne radi dobro na naivno postavljenom optimizacijskom problemu:

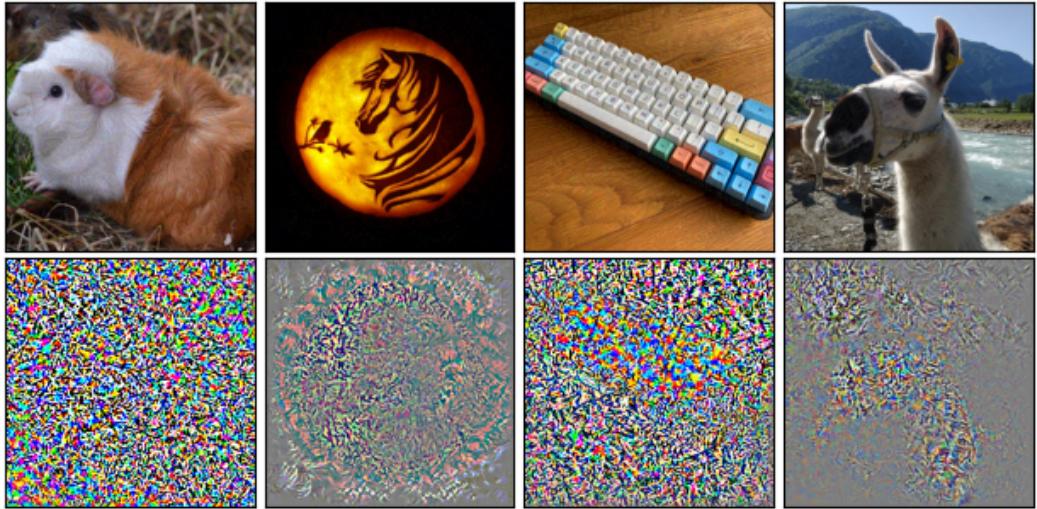
$$\text{minimizirati } c \cdot f(\mathbf{x} + \mathbf{r}) + \|\mathbf{r}\|_\infty$$

Dovoljno je normu zamijeniti s funkcijom koja kažnjava elemente r_i koji pre-laze vrijednost τ (inicijalno postavljena na 1, u svakoj iteraciji se smanjuje za faktor 0.9):

$$\text{minimizirati } c \cdot f(\mathbf{x} + \mathbf{r}) + \sum_i [(r_i - \tau)^+]$$

Koeficijent c se ponovno traži binarnim pretraživanjem. Postavi se na neku malu vrijednost i onda se provede optimizacija. Ako ne uspije, c se poveća za

faktor 2 i postupak se ponovi sve dok se ne pronađe dobro rješenje. Na slici 5.2 je prikazan primjer napada na *ResNet* mrežu.



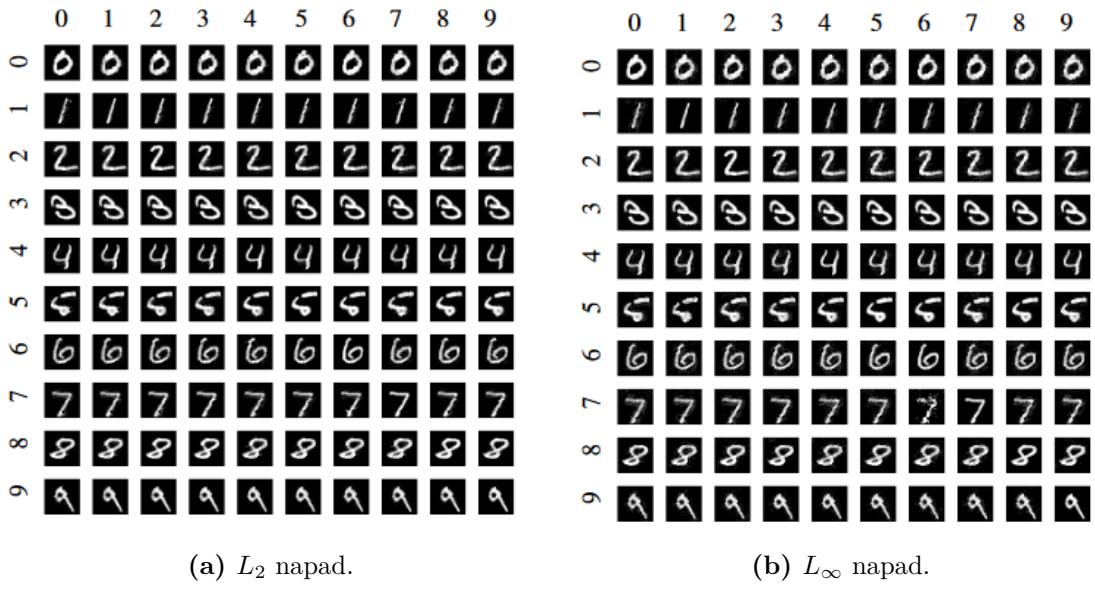
Slika 5.2: Ciljani CW ℓ_∞ napad. Sve četiri slike su klasificirane kao sombrero s vjerojatnostima od 92.04%, 96.55%, 99.73% i 99.85%. U donjem redu se nalazi normalizirana razlika između originalne i neprijateljske slike.

Problem s obrambenom destilacijom je opisan ranije. Metode napada na model bijele kutije često ovise o gradijentima modela pa su zato i bili neuspješni. Pod ovim napadom, obrambena destilacija pruža minimalnu zaštitu. Rezultati napada su prikazani u tablici 5.2.

Čisti podatci	CW ℓ_∞	CW ℓ_2
$\epsilon = 10$	78.93%	9.60%

Tablica 5.2: Rezultati CW napada na obrambenu destilaciju za $T = 100$. Napadi su provedeni na 500 i 200 primjera iz skupa podataka za testiranje.

Ovaj napad se pokazao iznimno uspješnim i teškim za poraziti. Zbog toga je postao standardni napad za evaluaciju obrana. Na slici 5.3 se nalaze napadi na *MNIST* skup podataka. Arhitektura i način treniranja nisu pretjerano različiti od opisanog u potpoglavlju 2.4.



Slika 5.3: Primjeri napada na *MNIST* skup podataka. Na dijagonalama se nalaze izvorne slike. Redovi su ispravne labele, a stupci su izlazne labele nakon ciljanog napada.

5.3.1. Napad na termometar kodiranje

Iako nije izravno vezano uz opisane CW napade, u nastavku je dana ideja napada na termometar kodiranje od strane istih autora^[21]. Ova i slične ideje se često primjenjuju pri opovrgavanju obrana koje upotrebljavaju neki nediferencijabilni element.

Mnogo nediferencijabilnih obrana se može opisati na slijedeći način: za neki klasifikator $f(\cdot)$ dan je postupak predobrade $g(\cdot)$. Obranjeni klasifikator je onda $\hat{f}(\mathbf{x}) = f(g(\mathbf{x}))$, a za g vrijedi $g(\mathbf{x}) \approx \mathbf{x}$. Ovakav $g(\cdot)$ je npr. JPEG kompresija, ali se JPEG kompresija ionako pokazala dovoljno slabom da ju i FGSM može zaobići.

S obzirom da vrijedi $g(\mathbf{x}) \approx \mathbf{x}$, derivacija funkcije se može aproksimirati kao derivacija funkcije identiteta, iako je sama transformacija nelinearna. Vrijedi $\nabla_{\mathbf{x}} g(\mathbf{x}) \approx \nabla_{\mathbf{x}} \mathbf{x} = 1$. Takve obrane se onda vrlo lako zaobilaze jer se dovoljno dobro mogu procijeniti gradijenti.

Iako je taj pristup efikasan kod obrana gdje stvarno vrijedi $g(\mathbf{x}) \approx \mathbf{x}$, nije dovoljno generalan za druge nediferencijabilne obrane. Takvo je na primjer termometar kodiranje.

Neka je neuronska mreža $f(\cdot) = f^{1 \dots j}(\cdot)$, a $f^i(\cdot)$ nediferencijabilni sloj. Da bi

se aproksimira $\nabla_x f(\mathbf{x})$ potrebna je prvo diferencijabilna aproksimacija $g(\mathbf{x}) \approx f^i(\mathbf{x})$. Pri prolazu unazad kroz mrežu se onda $f^i(\mathbf{x})$ zamjeni s $g(\mathbf{x})$. Iako funkcije nisu ekvivalentne, pokazalo se da je ovakva aproksimacija gradijenata dovoljno dobra za konstrukciju neprijateljskih primjera. Ovaj pristup se naziva diferencijalna aproksimacija prolaza unazad (engl. *Backward Pass Differentiable Approximation – BPDA*).

Termometar kodiranje za sliku \mathbf{x} je opisano u potpoglavlju 4.3. Za piksel $x_{i,j,c}$, termometar enkodirani vektor je zadan s $\tau(x_{i,j,c})_k = 1$ ako $x_{i,j,c} > k/l$, a 0 inače (u ranije opisanom potpoglavlju je nejednadžba bila obrnuta, a posljedično je rezultatni vektor bio negiran, no ovo ne utječe na svojstva obrane). Ako se zada:

$$\hat{\tau}(x_{i,j,c})_k = \min(\max(x_{i,j,c} - k/l, 0), 1)$$

onda vrijedi:

$$\tau(x_{i,j,c})_k = \text{floor}(\hat{\tau}(x_{i,j,c})_k)$$

Odabere li se potom $g(\mathbf{x}) = \hat{\tau}(\mathbf{x})$ moguće je provesti diferencijabilni prolaz unazad kroz mrežu i izračunati približne gradijente. Ovo efektivno poništava obranu i ponovno se mogu koristiti napadi na model bijele kutije kako su pretvodno opisani.

5.4. Granični napad

Do sada su bili razmatrani samo napadi na model bijele kutije. Takvi napadi su u pravilu jači od napada na model crne kutije jer mogu iskoristiti dodatno znanje o mreži da poboljšaju napad. Napadi na model crne kutije ne znaju ništa o mreži osim vrijednosti izlaza. U praksi, modeli o kojima je znanje minimalno su česti, na primjer modeli kojima se pristupa preko nekakvog web sučelja (npr. *Microsoft Azure* i slični servisi). Usprkos manjku znanja o mreži, moguće je konstruirati vrlo uspješne napade. Jedan od napada na model crne kutije je granični napad (engl. *boundary attack*). Napad je temeljen na odluci (engl. *decision based attack*), odnosno zahtjeva samo labelu na izlazu mreže. Za razliku od napada temeljenog na odluci, neki napadi crne kutije očekuju i vjerojatnosti na izlazu.

Napad započinje iz neke točke koja već je neprijateljska (odnosno pogrešno klasificirana) i iterativno se približava zadanim ulazu \mathbf{x} pod uvjetom da je u svakom koraku i dalje neprijateljska. Najjednostavniji opis algoritma je dan u nastavku. Korištena je notacija kao i u radu, stoga je oznaka za ulaz \mathbf{o} .

Podatci: ulazna slika \mathbf{o}

Rezultat: neprijateljski primjer $\tilde{\mathbf{o}}$

inicijalizacija: $k = 0$, $\tilde{\mathbf{o}}^0 \sim \mathcal{U}(0, 1)$ takav da je neprijateljski klasificiran;

dok $k < \text{maksimalni broj koraka čini}$

odabere se nasumična perturbacija iz distribucije $\boldsymbol{\eta}_k \sim \mathcal{P}(\tilde{\mathbf{o}}^{k-1})$;

ako je $\tilde{\mathbf{o}}^{k-1} + \boldsymbol{\eta}_k$ neprijateljski onda

postavi $\tilde{\mathbf{o}}^k = \tilde{\mathbf{o}}^{k-1} + \boldsymbol{\eta}_k$;

inače

postavi $\tilde{\mathbf{o}}^k = \tilde{\mathbf{o}}^{k-1}$;

kraj

$k = k + 1$

kraj

Algoritam 1: Najjednostavniji oblik graničnog napada. Potreban je samo izlaz mreže.

Ako je napad neciljani, ulazna slika može biti neki šum (pod uvjetom da je šum pogrešno klasificiran). Za ciljani napad, ulazna slika treba biti slika iz ciljanog razreda. Ključni dio algoritma je distribucija \mathcal{P} . Optimalna distribucija bi se u pravilu trebala razlikovati između domena problema i različitih modela, no iznenađujuće je kako je moguće uspješno konstruirati neprijateljske primjere uz jednostavnu distribuciju. Perturbacije se generiraju iz distribucije s maksimalnom entropijom uz sljedeća ograničenja:

1. Perturbirani uzorak je i dalje u domeni, odnosno:

$$\tilde{o}_i^{k-1} + \eta_i^k \in [0, 255] \quad (5.3)$$

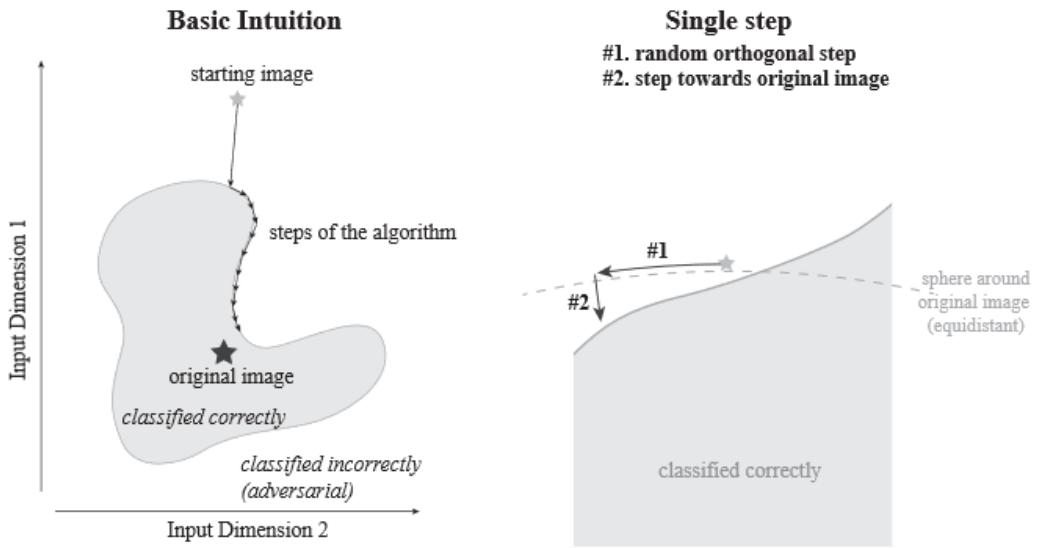
2. Perturbacija ima relativnu veličinu iznosa δ , odnosno:

$$\|\boldsymbol{\eta}^k\|_2 = \delta \cdot d(\mathbf{o}, \tilde{\mathbf{o}}^{k-1}), \text{ gdje je } d(\mathbf{o}, \tilde{\mathbf{o}}) = \|\mathbf{o} - \tilde{\mathbf{o}}\|_2^2 - \text{udaljenost između } \mathbf{o} \text{ i } \tilde{\mathbf{o}} \quad (5.4)$$

3. Perturbacija smanjuje udaljenost između ciljane slike i perturbirane slike za neki relativni iznos, odnosno:

$$d(\mathbf{o}, \tilde{\mathbf{o}}^{k-1}) - d(\mathbf{o}, \tilde{\mathbf{o}}^{k-1} + \boldsymbol{\eta}^k) = \epsilon \cdot d(\mathbf{o}, \tilde{\mathbf{o}}^{k-1}) \quad (5.5)$$

Autori su priložili ilustraciju koja slikovito prikazuje rad algoritma. Ilustracija je prikazana na slici 5.4.



Slika 5.4: Na lijevoj slici je prikazan rad algoritma kroz nekoliko koraka. U svakom koraku se neprijateljski primjer približava slici, ali tako da uvijek bude izvan granice ispravne klasifikacije. Na desnoj slici je prikazan jedan korak algoritma. Prvo se napravi nasumičan korak na sferi oko originalne slike, a nakon toga se napravi korak prema originalnoj slici.

Nije lako uzorkovati opisanu distribuciju \mathcal{P} , ali je moguće konstruirati jednostavniju heuristiku koja je dovoljno dobra. Prvo se uzorkuje normalna distribucija $\boldsymbol{\eta}_i^k \sim \mathcal{N}(0, 1)$ nakon čega se uzorak skalira i ograniči tako da zadovoljava ograničenja 5.3 i 5.4. U idućem koraku se radi pomak po sferi oko \mathbf{o} u nasumičnom smjeru, odnosno mora vrijediti $d(\mathbf{o}, \tilde{\mathbf{o}}^{k-1} + \boldsymbol{\eta}^k) = d(\mathbf{o}, \tilde{\mathbf{o}}^{k-1})$ i ograničenje 5.3. U zadnjem koraku se pomakne prema originalnoj slici tako da ograničenja 5.3 i 5.5 budu zadovoljena. Za visoko-dimenzionalne ulaze i dovoljno male δ i ϵ , ograničenje 5.4 će biti praktički zadovoljeno.

Postavlja se pitanje što je s lokalnim minimumima. Lokalni minimumi predstavljaju problem utoliko što se na višestruko pokretanje algoritma dogodi konvergencija prema različitim minimumima, ali ti minimumi su sličnog reda veličine. Ne dogada se slučaj da algoritam zapne u nekom lokalnom minimumu do te mjere da se ne postigne vizualno prihvatljiv neprijateljski primjer. Ta činjenica ukazuje na ne-robustnost modela. U idealnom slučaju napad se ne bi uspio približiti dovoljno ciljanoj slici da postane kvalitetan neprijateljski primjer. Na slici 5.5 su tri primjera napada.



Slika 5.5: Tri primjera graničnog napada na *ResNet* mrežu. U prvom redu su originalne slike, u drugom redu je rezultat nakon 400 koraka, a u trećem redu je rezultat nakon 8000 koraka. Algoritam jako brzo nađe šumovito rješenje, a onda većinu vremena optimizira neprijateljski primjer, kao što je pokazano na slici 5.4. Za svaki pojedini stupac, izlaz modela je isti za svaki red (po stupcima: *llama*, *jack-o'-lantern*, *African grey*).

Bitno je naglasiti da ni ovako jaki napad ne dolazi besplatno. U usporedbi s napadima bijele kutije, broj evaluacija mreže je veći za nekoliko redova veličine (uz isti cilj, na primjer isti iznos norme). Za DeepFool napad, autorima je u prosjeku trebalo 7 prolaza kroz mrežu (broj predikcija ulaza) i 37 prolaza unazad (za računanje gradijenata), dok je za granični napad sličnog iznosa norme potrebno 1, 200, 000 prolaza kroz mrežu i 0 prolaza unazad. Ovo je u slučaju da se žele iznimno dobri rezultati. Na slici 5.5 je napad već nakon 8000 iteracija

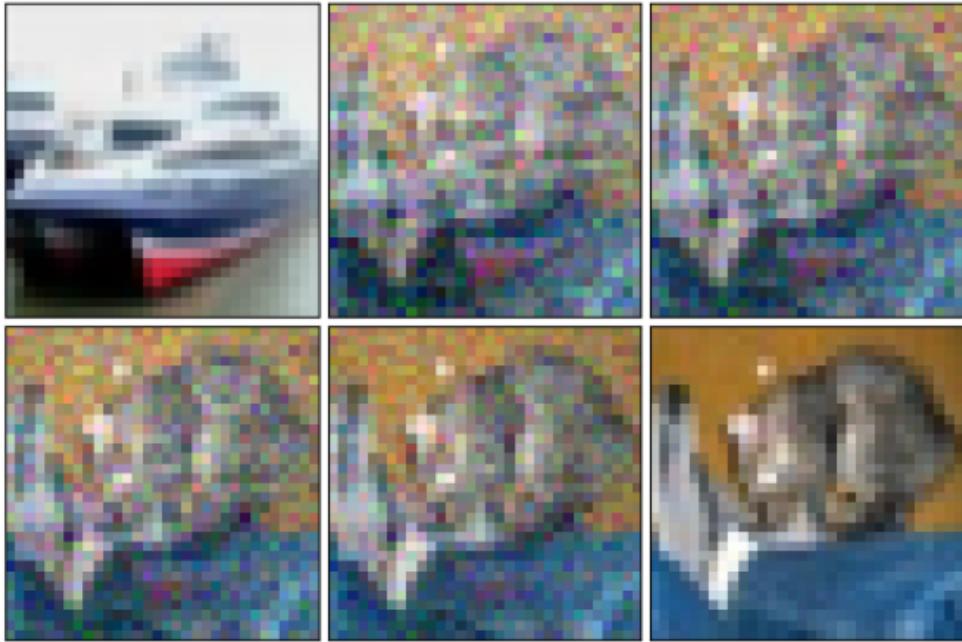
postigao vrlo dobar rezultat, a još su k tome i slike birane tako da problem bude teži. Ciljana slika ima vrlo jednostavnu jednoboju pozadinu i napad je svejedno iznimno uspješan. Jedino se u slučaju ljudi vidi obris, no i to bi nestalo nakon mnogo iteracija. Za 8000 iteracija potrebno vrijeme izvršavanja je preko 8 sati.

Što se tiče svih prethodno opisanih obrana, za sve postoji napad na model bijele kutije koji ih može poraziti. Slijedi primjer ovog napada na obrane neprijateljskog treniranja (uz FGSM) i obrambenu destilaciju. Napad na obrambenu destilaciju uz $T = 100$ je na slici 5.6. Napad na neprijateljski treniranu mrežu uz FGSM je na slici 5.7.

Dodatna zanimljivost je to što su autori graničnog napada također autori *FoolBox* biblioteke koja je bila razmatrana za korištenje u sklopu rada.



Slika 5.6: Napad na obrambenu destilaciju. Slike su nakon svake desete iteracije. Zadnja slika je nakon 2500 iteracija uz $\ell_2 = 107.63$.



Slika 5.7: Napad na neprijateljsko treniranje uz FGSM. Slike su nakon svake desete iteracije. Zadnja slika je nakon 2500 iteracija uz $\ell_2 = 101.51$.

5.5. Prenosivost napada

Neprijateljski primjeri iskazuju još jedno posebno svojstvo koje je jednako iznenajuće koliko i zabrinjavajuće. Moguće je konstruirati neprijateljske primjere za neki model, a da ti isti primjeri budu neprijateljski za drugi model. Ovo svojstvo se zove prenosivost (engl. *transferability*) ili generalizacija te je uočeno u isto vrijeme kada je nastao i FGSM napad^[13]. U nastavku slijede dva primjera:

- Uspješnost prenosivosti napada između dva različita modela
- Napad na obrambenu destilaciju putem prenosivosti

5.5.1. Prenosivost neprijateljskih primjera između modela

U svrhu provjere svojstva prenosivosti neprijateljskih napada je osmišljen dodatni model. Model se temelji na onom opisanom u 2.4, ali je kompleksniji.

Mreža se sastoji od 11 slojeva, redom:

- konvolucijski sloj oblika $32 \times 32 \times 64$ s filtrom veličine 3×3
- konvolucijski sloj oblika $32 \times 32 \times 64$ s filtrom veličine 3×3
- konvolucijski sloj oblika $32 \times 32 \times 64$ s filtrom veličine 3×3

- sloj sažimanja oblika 2×2
- sloj ispadanja s vjerojatnošću 0.2
- konvolucijski sloj oblika $16 \times 162 \times 128$ s filtrom veličine 3×3
- konvolucijski sloj oblika $16 \times 162 \times 128$ s filtrom veličine 3×3
- sloj sažimanja oblika 2×2
- sloj ispadanja s vjerojatnošću 0.2
- potpuno povezani sloj veličine 1024
- sloj ispadanja s vjerojatnošću 0.50
- potpuno povezani sloj veličine 10

Broj parametara koje mreža može naučiti je 8,696,970, a obje mreže su treningane 50 epoha uz optimizacijski algoritam Adam. Nakon 50 epoha, navedena mreža postigne točnost od 81.02%.

Mreže jesu slične, ali se i u mnogo toga razlikuju. Ponajviše po broju parametara koji se mogu naučiti te nešto drugčijoj arhitekturi. Radi kratkoće i čitkosti, mreža iz 2.4 je u nastavku označena s A , a nova mreža je označena s B . Prenosivost je pokazana u oba smjera: koliko su napadi konstruirani pomoću mreže A uspješni na mreži B i koliko su napadi konstruirani pomoću mreže B uspješni protiv mreže A . Isprobano je nekoliko verzija FGSM napada te Carlini i Wagner ℓ_2 i ℓ_∞ napadi. FGSM napad je proveden nad cijelim skupom za testiranje. Carlini i Wagner ℓ_2 napad je proveden samo na prvih 200 slika, a ℓ_∞ je proveden na prvih 500 slika skupa za testiranje. Model A na zadnjih 200 i 500 slika postiže točnost od 80.6% i 83.0%, a model B postiže točnost od 82.4% i 82.0%. Razlog provođenja napada samo na podskupu skupa za testiranje je vrijeme izvođenja. Hiperparametar je $\kappa = 50$ za oba CW napada.

Model	FGSM	FGSM	FGSM	CW ℓ_∞	CW ℓ_2
	$\epsilon = 2$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 10$	
A	51.36%	32.83%	22.19%	38.80%	10.00%
B	79.03%	72.18%	54.89%	49.20%	46.50%

Tablica 5.3: Rezultati neprijateljskih napada generiranih na modelu A . Model A je “jednostavniji”. Napadi s A djelomično prelaze na B , a ℓ_2 napad je najučinkovitiji za oba modela.

Model	FGSM	FGSM	FGSM	CW ℓ_∞	CW ℓ_2
	$\epsilon = 2$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 10$	
<i>A</i>	76.84%	67.39%	47.82%	36.66%	40.50%
<i>B</i>	51.08%	33.69%	25.17%	22.88%	11.00%

Tablica 5.4: Rezultati neprijateljskih napada generiranih na modelu *B*. Model *B* je “kompleksniji”. Napadi konstruirani na *B* relativno uspješno prelaze na *A*.

Rezultati su fascinantni. Napadi konstruirani na *A* modelu prelaze na model *B* u puno manjoj mjeri nego obrnuto. Napad CW uz ℓ_∞ normu je zapravo uspješniji na modelu *B*, a još k tome je i uspješniji nego kada se napad izravno provede na *A*. Bitno je ponovno uočiti dvije stvari:

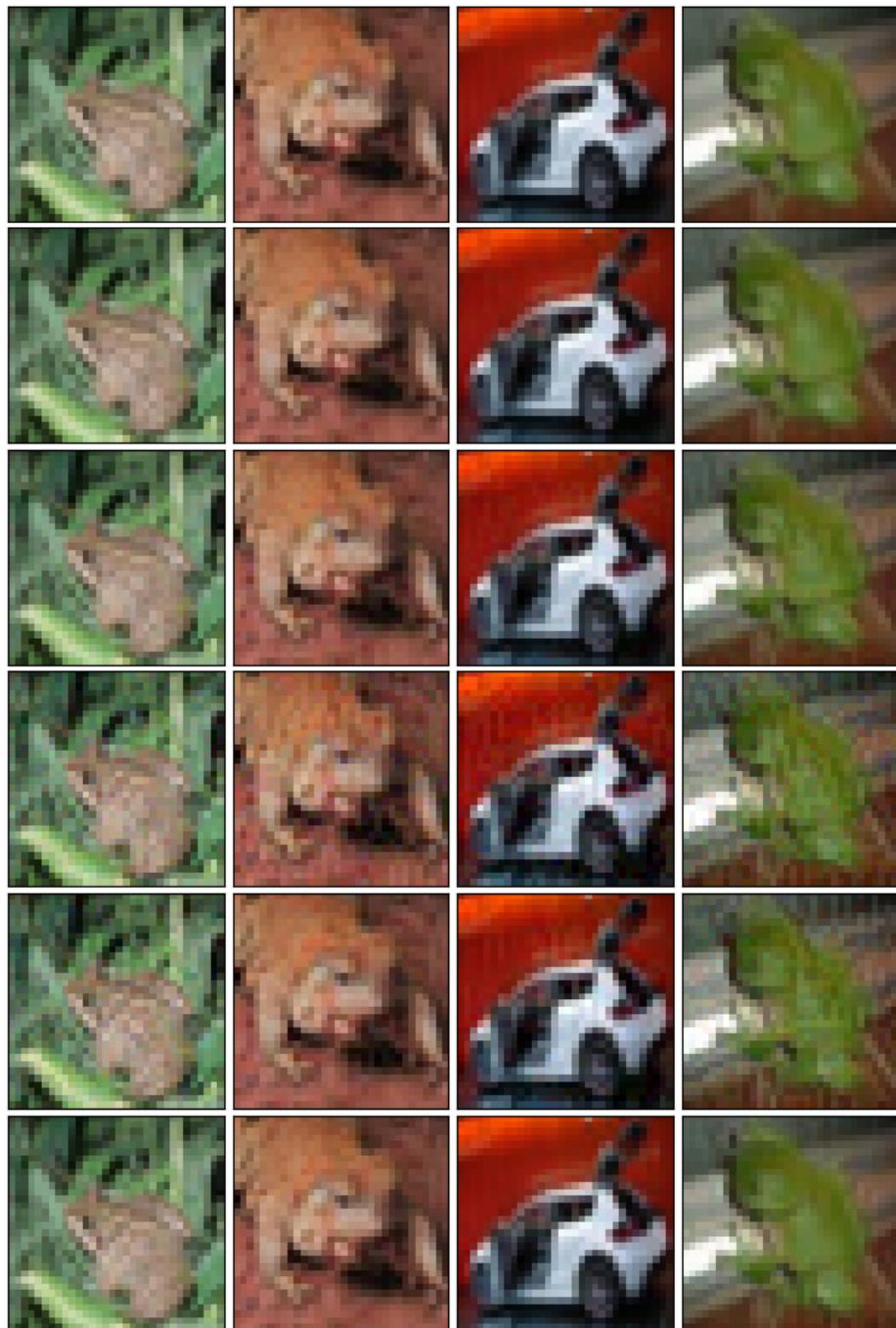
- Model *B* ima otprilike 4 puta više parametara od modela *A* i drugčiju arhitekturu
- Ovo nipošto ne može biti slučajnost jer nasumični šum ne prouzročuje ni približno toliki pad u točnosti. Ovo je pokazano u tablici 5.5.

Model	Čisti podatci	Šum od $\epsilon \in [-10, 10]$	Gaussov šum s $\mu = 0$ i $\sigma = 5$	Šum od $\epsilon \in \{-10, 10\}$
<i>A</i>	79.80%	79.53%	78.87%	72.81%
<i>B</i>	81.02%	79.54%	78.67%	72.22%

Tablica 5.5: Točnost modela na čistom *CIFAR-10* skupu i na istom skupu uz nasumični šum u rasponu $\epsilon \in [-10, 10]$, Gaussov šum uz $\sigma = 5$ i šum $\epsilon \in \{-10, 10\}$. U svim slučajevima je šum skaliran i slika omeđena ponovo na $[0, 1]$. Ovo je pretpostavka lokalne generalizacije.

Teorije koje pokušavaju objasniti prenosivost postoje, ali su izvan dosega ovog rada. Dodatno je zanimljivo kako prenosivost ne ovisi čak ni o skupu podataka na kojem se trenira te je moguće napraviti napade koji prelaze s modela na model čak i kada su trenirani na različitim skupovima podataka^[13]. Prenosivost je bitna pojava koje se treba uzeti u obzir pri evaluaciji obrana. Zato u idućem potpoglavlju slijedi kratka evaluacija obrambene destilacije korištenjem prenosivosti. Naime, robusna mreža ne bi trebala biti podložna ovakovom obliku napada. Na slici 5.8 su prikazani uspješni neprijateljski primjeri koji su preneseni s *B* na *A*. U

prvom redu su originalne slike, a nadalje su u svakom redu neprijateljski primjeri za napade kako su navedeni ranije.



Slika 5.8: Primjeri uspješno prenesenih neprijateljskih primjera s modela B na model A . U prvom redu su originalne slike, iduća tri reda su FGSM napadi te su u zadnja dva reda CW napadi.

5.5.2. Prenosivost neprijateljskih primjera na obrambenu destilaciju

U 5.3 je pokazana uspješnost Carlini i Wagner napada na obrambenu destilaciju. U ovom dijelu je testirana prenosivost FGSM i CW napada na destilirani model. Rezultati se nalaze u tablici 5.6. Napad je proveden na standardno treniranom modelu iz 2.4 nakon 50 epoha. Hiperparametar je $\kappa = 50$ za oba CW napada, a napad je proveden na 500 i 200 slika kao i u prethodnom poglavlju.

FGSM $\epsilon = 2$	FGSM $\epsilon = 5$	FGSM $\epsilon = 10$	CW ℓ_∞ $\epsilon = 10$	CW ℓ_2
75.39%	65.47%	45.65%	35.20%	36.00%

Tablica 5.6: Prenosivost sa standardno treniranog modela na obrambeno destilirani model.

Zaključak je ponovno da obrambena destilacija ne pruža jaku zaštitu, pogotovo u slučaju CW napada. U potpoglavlju 5.5.1 je provedena evaluacija prenosivosti između dva modela s različitim arhitekturama i brojem parametra. U ovom je slučaju provedena evaluacija prenosivosti između iste arhitekture i jednakog broja parametara, ali s različitim režimom treniranja.

6. Obrana dubokih konvolucijskih modela II

6.1. Preduvjeti uspješnih obrana

Zaključak do sada je da je evaluacija obrana vrlo teška. Usprkos velikom uloženom radu, vrlo malo obrana se uistinu pokazalo uspješnima. Čak i obrane koje su se pokazale uspješnima nisu bile idealno rješenje koje uvijek radi. Jedna od takvih obrana je opisana u potpoglavlju 6.2 i temelji se na ranije opisanom projiciranom gradijentnom spustu. Svakako niti jedna od naizgled idealnih obrana nije dugo opstala, takve su na primjer termometar kodiranje i defenzivna destilacija.

Potaknuti neuspjehom dosadašnjih obrana, skupina autora prethodno opisanih napada i obrana objavljaju popis principa za uspješnu evaluaciju obrana te česte "zamke" s kojima su se prethodne obrane susrele^[24]. U nastavku su dani neki od bitnih principa i zamki.

Bitni principi pri evaluaciji obrana

1. Precizno navesti model prijetnje za koji bi obrana trebala biti uspješna.
2. Provesti adaptivne napade te dati gornju granicu robusnosti.
 - (a) Adaptivni napadi su napadi koji su izravno prilagođeni obrani. Nije dovoljno isprobati nasumične napade, nego je nužno osmisliti i iskoristiti napad koji se "prilagodio" toj specifičnoj obrani.
 - (b) Napadi moraju imati pristup mreži od početka do kraja.
 - (c) Fokus treba biti na najjačim takvim napadima.
 - (d) Potvrditi da adaptivni napadi rade bolje od bilo kojeg drugog oblika napada.
3. Podijeliti izvorni kôd obrane i trenirane modele.
4. Navesti i opisati korištene napade i njihove hiperparametre.
5. Provjeriti da vrijede najosnovnije i nužne pretpostavke:

- (a) Provjeriti da iterativni napadi rade bolje od napada od jednog koraka.
 - (b) Provjeriti da pojačanje perturbacije nužno povećava uspješnost napada.
 - (c) Provjeriti da pri velikim perturbacijama, model se dovodi u stanje “nasumičnog pogadanja” izlazne vrijednosti.
6. Dati točnost modela bez ikakvog napada.
 7. Evaluirati obranu korištenjem prenosivosti neprijateljskih primjera.

Česte zamke

1. Potrebno je obranu evaluirati na širokom skupu napada, posebno u slučaju kada se obranu neprijateljski trenira na samo jednom napadu.
2. Isprobati barem jedan napad koji ne koristi gradijente te jedan napad koji koristi samo labele.
 - (a) Provjeriti da su napadi koji ne koriste gradijente neuspješniji nego napadi koji koriste gradijente (inače je velika vjerojatnost da je došlo do maskiranja gradijenata).
 - (b) Pažljivo istražiti hiperparametre napada koji utječu na uspješnost.
3. Za ne-diferencijabilne komponente, primjeniti diferencijabilne tehnikе.
4. Provjeriti da je napad konvergirao za određene hiperparametre.
5. Pažljivo odabratи hiperparametre korištenih napada, navesti korištene parametre i opisati njihov utjecaj.

Ovo su samo neki od principa koje bi trebalo slijediti i zamki koje bi trebalo izbjegavati. U stvarnosti je lista mnogo dulja, te svaki pojedini pristup ima svoje posebne napomene, principe i zamke. Ovakve liste se ne bi trebale slijepo pratiti te je uvijek potrebno pristupiti evaluaciji bilo kakvog napada ili obrane na vrlo (samo)kritičan način. Ako se neki bitni principi zanemaruju, bitno je naglasiti i objasniti zašto.

Postavlja se pitanje zašto je toliki fokus na napade na model bijele kutije i zašto se neka obrana smatra slabom samo zato što ju se može zaobići s vrlo specifičnim napadom. Zapravo se koncept neprijateljskih primjera lako može usporediti s napadom na kriptografski sustav. Kriptografski sustav se ne može smatrati sigurnim samo zato što je crna kutija, a ako postoji bilo kakav oblik napada na kriptografski sustav, onda definitivno nije siguran. Uz taj pristup se veže princip sigurnosti kroz zamračivanje (engl. *security through obscurity*). Skrivanje detalja kriptografskog sustava može povećati sigurnost u kratkom roku, ali u dugom roku je moguće vjerovati samo sustavima koji su javno objavljeni i

analizirani. Uz kriptografske sustave se veže i Kerckhoffsov princip, koji je se u originalnom obliku sastojao od šest aksioma i odnosio se na vojne šifre. Izdvojene su najrelevantnije značajke principa.

Kerckhoffsov princip

1. Dizajn sustava ne treba zahtijevati tajnost.
2. Može postojati tajni dio, ali se mora moći lako zamijeniti ako se otkrije.

U kontekstu kriptografskih sustava, tajni dio je najčešće tajni ključ koji se lako ponovno generira ako ga napadač sazna. Sličan pristup se može primijeniti kod obrane dubokih modela. Obrana bi trebala biti uspješna čak i ako ju napadač zna te je prihvatljivo da postoji tajna koja je vezana uz obranu (npr. ključ). Gradjeni mreže ne spadaju u ovu tajnu jer se težine mreže ne mogu lako zamijeniti.

Ova i slične liste imale su pozitivan učinak na tok istraživanja područja neprijateljskih primjera. U usporedbi s prijašnjim radovima, današnji radovi su svakako “bolji”. Puno je truda uloženo da bi se pokazao neuspjeh ranijih obrana kao što su termometar kodiranje ili defenzivna destilacija, dok je danas teorijska podloga mnogo jača i zamke se puno lakše uočiti i izbjjeći.

6.2. Neprijateljsko treniranje - PGD

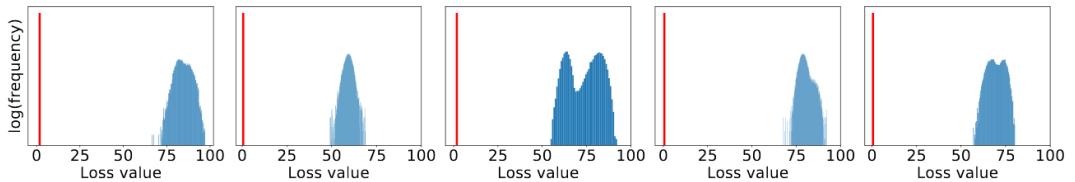
Ideja neprijateljskog treniranja pojavila se već uz FGSM napad^[13] te je treniranje tog oblika evaluirano u potpoglavlju 4.2. Tada se pokazalo da neprijateljsko treniranje povećava robustnost modela, ali model postaje pretreniran na hiperparametar napada i ne uspije generalizirati na druge napade. Međutim, postoji i generalizirana verzija FGSM napada: projicirani gradijentni spust – PGD (opisan u potpoglavlju 5.2).

U kontekstu standardnog treniranja dubokog modela, cilj je pronaći parametre mreže $\boldsymbol{\theta} \in \mathbb{R}^p$ takve da se minimizira očekivana vrijednost funkcije gubitka: $\mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{D}}[J(\mathbf{x}, y, \boldsymbol{\theta})]$. Funkcija $J(\cdot)$ je najčešće gubitak unakrsne entropije, a \mathcal{D} predstavlja distribuciju podataka. U ovu definiciju je moguće direktno ugraditi optimizacijski problem neprijateljskih primjera. Za svaki uzorak \mathbf{x} definira se skup dopuštenih perturbacija $\mathcal{S} \subseteq \mathbb{R}^d$, na primjer ℓ_∞ kugla oko uzorka. Umjesto da se izravno minimizira gubitak na podatcima iz \mathcal{D} , prvo se dopusti “neprijatelju” da maksimizira gubitak tako da se konstruiraju neprijateljski primjeri. Optimizacijski problem onda poprima drugi oblik:

$$\min_{\boldsymbol{\theta}} \rho(\boldsymbol{\theta}) \text{ gdje je } \rho(\boldsymbol{\theta}) = \mathbb{E}_{(\boldsymbol{x}, y) \sim \mathcal{D}} [\max_{\boldsymbol{r} \in \mathcal{S}} J(\boldsymbol{x} + \boldsymbol{r}, y, \boldsymbol{\theta})] \quad (6.1)$$

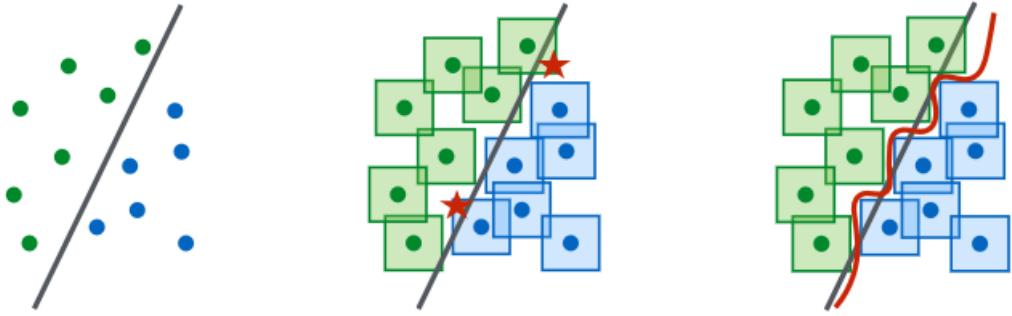
Optimizacijski problemi ovog oblika su poznati i rješenje problema se zove sedlasta točka (engl. *saddle point*) ili *minimax* točka. U kontekstu ranije opisanog neprijateljskog treniranja gdje su se neprijateljski primjeri stvarali s određenim omjerom (npr. 50%), u ovom obliku treniranja se svi primjeri pretvaraju u neprijateljske pri treniranju.

Iako je optimizacija sedlaste točke poznat problem, ovdje je slučaj da je vanjski minimizacijski problem vrlo ne-konveksan i unutarnji maksimizacijski problem vrlo ne-konkavan što znatno otežava optimizaciju do te mjere da se ona na prvi pogled može smatrati nerješivom. Međutim, najveći doprinos ovog rada je demonstracija toga da je ovaj problem ipak rješiv u praksi. Usprkos tome što maksimuma funkcije gubitka ima mnogo u prostoru $\boldsymbol{x} + \mathcal{S}$, njihovi iznosi su iznimno slični te autori nisu pronašli ekstreme nakon 10^5 napada. Dodatno je pokazano da PGD treniranje drastično smanjuje taj maksimalni iznos što ide u prilog tome da je PGD oblik univerzalnog neprijateljskog napada. Na slici 6.1 su prikazani histogrami vrijednosti funkcije gubitka za 10^5 različitih početnih točaka napada.



Slika 6.1: Frekvencija vrijednosti funkcije gubitka nakon PGD napada. Napad je proveden na 5 slika iz 10^5 različitih početnih točaka. Crvenom bojom su označene vrijednosti nakon neprijateljskog treniranja uz PGD.

Dakle, korištenjem PGD napada pokazano je da za neki \boldsymbol{x} može biti generirano mnogo različitih neprijateljskih primjera, ali će svi ti neprijateljski primjeri povećati funkciju gubitka do približno iste vrijednosti. Autori su trenirali dva modela: jedan za *MNIST* te jedan za *CIFAR-10*. Pokazalo se da je kapacitet mreže od presudne važnosti za uspješnost neprijateljskog treniranja. Razlog tome je što je potrebno jako povećati nelinearost granice između razreda. Ilustracija toga je na slici 6.2.



Slika 6.2: Na lijevoj slici je pokazana linearna razdvojivost između dva razreda te je granica vrlo jednostavna. Na srednjoj slici su prikazani primjeri uz njihovo ϵ -okruženje. Neprijateljski primjer može biti u okruženju, a pripasti drugom razredu. To je označeno zvjezdicom. Na desnoj slici je nacrtana nelinearna granica koja razdvaja i ϵ -okruženja.

Autori su zato koristili široku verziju *ResNet* mreže koja ima puno veći kapacitet od “obične” mreže. Rezidualne jedinice u ovoj širokoj mreži imaju redom broj filtera: 16, 160, 320, 640. Na ovoj mreži se može postići točnost od 95.2% uz standardno treniranje. U sklopu rada je testirano neprijateljsko treniranje na standardnoj *ResNet-18* mreži. Nakon 30 sati treniranja, postignuta točnost na čistim podatcima je 74.20%. Uspješnost neprijateljskih primjera je praktički jednaka, te nije postignuta veća robusnost modela na neprijateljske primjere što ukazuje da je veći kapacitet presudan. Za treniranje *CIFAR-10* modela opisanog u radu je autorima trebalo 80 sati. U svrhu evaluacije obrane, autori su organizirali dva “natjecanja” gdje je cilj bio poraziti obranu na *MNIST*¹ i *CIFAR-10*² skupu podataka.

Za *CIFAR-10*, najbolji rezultat napada na model crne kutije smanjio je točnost modela na 92.76%. Najbolji napad na model bijele kutije smanjio je točnost na 43.99%. Iako brojevi možda izgledaju poražavajuće, ovo je trenutno najuspješnija obrana protiv neprijateljskih napada. Bitno je podsjetiti se da neprijateljski napadi na *CIFAR-10* modele mogu lako spustiti točnost do približno 0%.

Za *MNIST*, najbolji rezultat napada na model crne kutije smanjio je točnost modela na 63.39%. Najbolji napad na model bijele kutije smanjio je točnost na 88.06%. Ovdje je obrana iznimno uspješna.

Iako je obrana iznimno uspješna za *MNIST* i relativno uspješna za *CIFAR-10*

¹https://github.com/MadryLab/mnist_challenge

²https://github.com/MadryLab/cifar10_challenge

6.3. Dokazivost obrane od neprijateljskih napada

6.4. Budući rad

7. Zaključak

Konvolucijske neuronske mreže su se pokazale iznimno uspješnima pri rješavanju problema klasifikacije slika. Međutim, pojava neprijateljskih primjera dovela je u pitanje tvrdnju da današnji modeli dobro generaliziraju. Također se ispostavilo da je vrlo lako osmisliti uspješne algoritme koji mogu generirati neprijateljske primjere, a za neke napade čak nije potrebno nikakvo dodatno znanje o mreži koju se napada. Ubrzo nakon pojave neprijateljskih primjera su se pojavile i prve potencijalne obrane. Iako su sve obrane na prvi pogled izgledale obećavajuće, pokazano je da većina tih obrana također ne mogu generalizirati na jače napade. Od svih obrana do sada, gotovo niti jedna obrana nije se pokazala korisnom. Najviše obećavajuća obrana do sada temelji se na posebnom obliku treniranja mreže korištenjem neprijateljskih primjera, no to je ono što tu obranu čini skupom i neuporabljivom na većim skupovima podataka kao što je *ImageNet*. Budućnost obrana od neprijateljskih primjera još uvijek nije očita. Praktički svaki dan se pojavljuju sve efikasnije metode temeljene na suparničkom treniranju, formalni postupci koji mogu dokazati robusnost mreže te potpuno nove metode obrana koje će tek biti detaljno testirane. Međutim, danas obrana dubokih konvolucijskih modela još uvijek stoji kao vrlo izazovan i neriješen problem.

LITERATURA

- [1] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, i Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3): 211–252, 2015. doi: 10.1007/s11263-015-0816-y.
- [2] 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*, stranice 1097–1105. Curran Associates, Inc., 2012. URL <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>.
- [3] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, i Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2014.
- [4] Nicholas Carlini. A complete list of all (arxiv) adversarial example papers. <https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html>. Zadnji pristup: 2020-06-27.
- [5] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. U *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, stranice 265–283, 2016.
- [6] François Chollet et al. Keras. <https://keras.io>, 2015.

- [7] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhi-shuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, i Rujun Long. Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv preprint arXiv:1610.00768*, 2018.
- [8] Jonas Rauber, Wieland Brendel, i Matthias Bethge. Foolbox: A python toolbox to benchmark the robustness of machine learning models. In *Reliable Machine Learning in the Wild Workshop, 34th International Conference on Machine Learning*, 2017. URL <http://arxiv.org/abs/1707.04131>.
- [9] Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, Beat Buesser, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, Ian Molloy, i Ben Edwards. Adversarial robustness toolbox v1.2.0. *CoRR*, 1807.01069, 2018. URL <https://arxiv.org/pdf/1807.01069>.
- [10] Alex Krizhevsky, Vinod Nair, i Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL <http://www.cs.toronto.edu/~kriz/cifar.html>.
- [11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, i Jian Sun. Identity mappings in deep residual networks. *ArXiv*, abs/1603.05027, 2016.
- [12] Diederik P. Kingma i Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.
- [13] Ian J. Goodfellow, Jonathon Shlens, i Christian Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2015.
- [14] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, i Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, stranice 2574–2582, 2016.
- [15] Gintare Karolina Dziugaite, Zoubin Ghahramani, i Daniel M. Roy. A study of the effect of jpg compression on adversarial images. *ArXiv*, abs/1608.00853, 2016.

- [16] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Li Chen, Michael E. Kounavis, i Duen Horng Chau. Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression. *ArXiv*, abs/1705.02900, 2017.
- [17] Chuan Guo, Mayank Rana, Moustapha Cissé, i Laurens van der Maaten. Countering adversarial images using input transformations. *ArXiv*, abs/1711.00117, 2018.
- [18] Weilin Xu, David Evans, i Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *ArXiv*, abs/1704.01155, 2018.
- [19] Yash Sharma i Pin-Yu Chen. Bypassing feature squeezing by increasing adversary strength. *ArXiv*, abs/1803.09868, 2018.
- [20] Jacob Buckman, Aurko Roy, Colin Raffel, i Ian Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. 2018. URL <https://openreview.net/pdf?id=S18Su--CW>.
- [21] Anish Athalye, Nicholas Carlini, i David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *ArXiv*, abs/1802.00420, 2018.
- [22] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, i Patrick D. McDaniel. Ensemble adversarial training: Attacks and defenses. *ArXiv*, abs/1705.07204, 2018.
- [23] Nicholas Carlini i David A. Wagner. Towards evaluating the robustness of neural networks. *2017 IEEE Symposium on Security and Privacy (SP)*, stranice 39–57, 2017.
- [24] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian J. Goodfellow, Aleksander Madry, i Alexey Kurakin. On evaluating adversarial robustness. *ArXiv*, abs/1902.06705, 2019.
- [25] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, i Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *ArXiv*, abs/1706.06083, 2017.

- [26] Eric Wong, Leslie Rice, i J. Zico Kolter. Fast is better than free: Revisiting adversarial training. *ArXiv*, abs/2001.03994, 2020.

8. Dodatak

8.1. Osobni skup slika

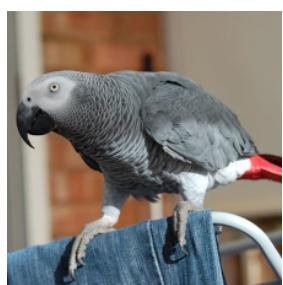
U skupu slika 8.1 je 16 slika i labela koje su u radu korištene za generiranje suparničkih primjera za modele trenirane na *ImageNet* skupu podataka. Sve slike su odabранe tako da spadaju unutar 1000 razreda za koje su modeli predviđeni, dakle slike bi trebale biti ispravno klasificirane. Također, sve slike su unaprijed izrezane u oblik kvadrata kako ne bi došlo do prevelike degradacije kvalitete pri smanjenju rezolucije na 224×224 . Ovim putem se zahvaljujem Sarah James na dopuštenju za korištenje slika u radu.

8.2. Izlazi modela na nepromijenjenim slikama iz osobnog skupa

Slike iz 8.1 su birane tako da budu reprezentativne za pojedinu klasu, odnosno da ne postoji mogućnost da su mreže “zbunjene” oko toga što je na slici. Ideja iza toga je pokazati kako je neprijateljske primjere lako konstruirati čak i kada je mreža iznimno sigurna u to što vidi na slici. U tablici 8.1 nalaze se top-1 izlazi mreža *ResNet50 V2* (primarna mreža korištena kroz rad) i *Xception*.

Slika	ResNet50 V2	Xception
8.1a	African grey 100.00%	African grey 100.00%
8.1b	Backpack 99.71%	Backpack 99.99%
8.1c	Dragonfly 99.93%	Dragonfly 99.79%
8.1d	Bald eagle 87.47%	Bald eagle 99.62%
8.1e	Grocery store 90.81%	Grocery store 97.68%
8.1f	Gorilla 99.83%	Gorilla 99.96%
8.1g	Guinea pig 100.00%	Guinea pig 99.99%
8.1h	Jack-o'-lantern 100.00%	Jack-o'-lantern 99.89%
8.1i	Jigsaw puzzle 100.00%	Jigsaw puzzle 100.00%
8.1j	Computer keyboard 65.34%	Computer keyboard 99.68%
8.1k	Llama 100.00%	Llama 100.00%
8.1l	Meerkat 99.98%	Meerkat 99.89%
8.1m	English Springer 88.93%	English Springer 99.30%
8.1n	Running shoe 80.76%	Running shoe 99.48%
8.1o	Tabby 64.28%	Tabby 89.72%
8.1p	Wardrobe 87.16%	Wardrobe 79.05%

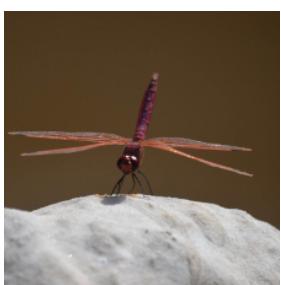
Tablica 8.1: Tablica top-1 izlaza mreža ResNet50 V2 i Xception za slike iz 8.1.



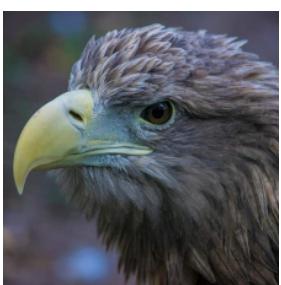
(a) *African grey* (87)



(b) *Backpack* (414)



(c) *Dragonfly* (319)



(d) *Bald eagle* (22)



(e) *Grocery store* (582)



(f) *Gorilla* (366)



(g) *Guinea pig* (338)



(h) *Jack-o'-lantern* (607)



(i) *Jigsaw puzzle* (611)



(j) *Keypad* (508)



(k) *Llama* (355)



(l) *Meerkat* (299)



(m) *English springer*
(217)



(n) *Running shoe* (770)



(o) *Tabby* (281)



(p) *Wardrobe* (894)

Slika 8.1: Slike korištene za generiranje suparničkih primjera te njihove ispravne labele

Obrana dubokih konvolucijskih modela od neprijateljskih primjera

Sažetak

Današnji konvolucijski modeli postižu visoku točnost u području raspoznavanja objekata. Način rada dubokih modela je još uvijek vrlo teško ili nemoguće interpretirati, a dodatan razlog za brigu predstavljaju i takozvani neprijateljski primjeri. Neprijateljski primjeri su slike s dodanim teško uočljivim perturbacijama koje potiču model na pogrešnu klasifikaciju. Pokazalo se da je vrlo lagano konstruirati brze i efikasne napade na postojeće modele, nakon čega su se ubrzo pojavile i obrane protiv takvih napada. Nedugo nakon toga pojavljuju se sve jači napadi, dok su obrane stagnirale te danas još uvijek ne postoji zadovoljavajuća obrana protiv neprijateljskih napada. U radu je dan pregled spomenutih jednostavnih napada i jednostavnih obrana, istaknuta je neuspješnost jednostavnih obrana protiv jačih napada te je opisana obećavajuća obrana koja se temelji na treniranju mreže korištenjem neprijateljskih primjera.

Ključne riječi: klasifikacija objekata, konvolucijske neuronske mreže, računalni vid, suparnički primjeri, neprijateljski primjeri, obrana

Defending Deep Convolutional Models from Adversarial Examples

Abstract

Today's convolutional models can achieve high accuracy in the field of object recognition. The way deep models work is still very difficult or impossible to interpret, and an additional reason for concern are the so-called adversarial examples. Adversarial examples are images with added imperceptible perturbations that encourage the model to misclassify the image. It turned out to be very easy to construct fast and effective attacks on existing models, after which new defenses against these attacks also appeared. Soon afterwards even stronger attacks appeared, while new defenses stagnated and today there isn't a satisfactory defense against adversarial attacks. The thesis reviews the aforementioned simple attacks and simple defenses, pointing out the failure of simple defenses against strong attackers. Additionally, a promising defense is described. The defense is based on the concept of adversarial training and has shown good results.

Keywords: object classification, convolutional neural networks, computer vision, adversarial attacks, defense