MNB-BME Együttműködés
2020/2021
Digitalizáció, mesterséges
intelligencia és adatkorszak Műhely

Szűcs Gábor

Gépi tanuló modellek elleni adversarial támadások és védekezési módszerek

A tanulmány a Magyar Nemzeti Bank és a Budapesti Műszaki és Gazdaságtudományi Egyetem között létrejött Együttműködés keretében és finanszírozásával készült a Digitalizáció, mesterséges intelligencia és adatkorszak műhelyben.

BUDAPEST, 2021

Tartalomjegyzék

Vezetői összefoglaló

Az elmúlt években a mesterséges intelligencia technológiáit széles körben alkalmazták a számítógépes látásban, természetes nyelvfeldolgozásnál, az iparban, a gazdasági és egyéb területeken. A mesterséges intelligenciát használó rendszerek azonban kiszolgáltatottak a különböző támadásoknak, így ezzel korlátozzák az intelligens technológiák alkalmazását a legfontosabb biztonsági területeken. Ezért ezeknek a rendszereknek a robusztusságának javítása a támadásokkal szemben egyre fontosabb szerepet játszik a mesterséges intelligencia további fejlődésében.

Ez a tanulmány kívánja összefoglalni a mesterséges intelligencia elleni támadásokkal kapcsolatos legújabb kutatási eredményeket (itt elsősorban a mély neurális hálózatok elleni támadásokról esik szó) és a védelmi technológiákat. Bemutatjuk, hogy hogyan kategorizálhatók a gépi tanuló modellek elleni támadások, majd részletesen tárgyaljuk az adversarial mintákon alapuló támadások módszereit és az ellenük való védekezési lehetőségeket. Tanulmányunkat a különböző alkalmazási területekből vett konkrét példákkal (arcfelismerés, támadás a fizikai világ területén) zárjuk.



1 Bevezetés

A mesterséges intelligencia bár nem új diszciplína, de a legperspektivikusabb része, a mélytanulás technológiái nemrégiben alakultak ki, és egyre szélesebb körben alkalmazzák ezeket a különböző területeken, mint például a képek osztályozásában, az objektumok felismerésében, a hangvezérlésben, a gépi fordításban, a pénzügyi alkalmazásokban és az orvosi területeken.

A mesterséges intelligenciánál a gépi tanulási modelleket (különösen a mélytanulást végző mély neurális hálózatokat) viszont meg lehet téveszteni egy szándékosan módosított (elrontott) bemenettel, ezt a szándékos megtévesztést *adversarial támadás*nak hívják [22]. Ezt a támadó irányított módon az *adversarial gépi tanulás*sal tudja elérni, amely egy speciális gépi tanulási technika; ennek célja olyan megtévesztő bemenetek keresése, amelyek nagyon hasonlítanak az igazi (eredeti) bemenetekhez, de egy támadó jellegű zaj hozzáadásával próbálják meg becsapni a modelleket. Ez a fajta támadás és a védekezés ellene egy nagyon friss és fontos kutatási terület a mesterséges intelligencia diszciplínában.

Azt a modellt, amit a támadó meg szeretne téveszteni, *célmodell*nek hívjuk; azokat a bemeneti adatokat (példányokat) pedig, amikkel a megtévesztést el szeretné érni, *adversarial példák*nak / *adversarial minták*nak nevezzük (bár lehetne használni magyar fordítást erre, mint például *támadó minta,* de a magyar változat elterjedtségének hiánya miatt a tanulmányunkban az eredeti angol kifejezésnél maradunk). A célmodell különböző szakaszai szerint a támadások felosztathatók 2 kategóriára: a tanítási szakaszban és a tesztelési szakaszban alkalmazott támadásokra.

A tanítás szakaszában zajló támadások arra utalnak, hogy a célmodell kialakításának szakaszában olyan támadásokat hajtanak végre, amely a tanuló adatállomány módosításával, a bemeneti jellemzők vagy az osztálycímkék manipulálásával jár. Ilyen beavatkozás a tanuló adatállomány eredeti eloszlásának módosítása, vagy néhány adat törlése vagy akár olyan új adatok hozzáadása, amely a megtévesztést elősegíti.

A tesztelési szakasz támadásai felosztathatók fehérdobozos (white-box) és feketedobozos (black-box) támadásokra. Fehérdobozos esetekben a támadó hozzáférhet a célmodell struktúrájához, paramétereihez, vagy akár a célmodellben megvalósított

algoritmusokhoz, míg a feketedobozos esetekben nem férhet hozzá. A támadók ezen ismeretek felhasználásával olyan bemeneti mintákat (példányokat) készíthetnek a támadások végrehajtására, melyek szándékosan megtévesztik a gépi tanuló modellt.

A támadó célja alapján szintén két csoportra oszthatók az osztályozó modellt érhető támadások: célzott és nem célzott támadásra. Célzott támadás esetén a támadó olyan minimális zajt keres, amivel egy kiválasztott osztályba juttathatja el a mintát (azaz a modell ezt fogja predikcióként adni). Nem célzott támadás esetén a támadó célja bármely olyan osztályba eljuttatni a mintát, ami nem egyezik az eredeti osztállyal.

Bár ennek a problémakörnek a szakirodalma még nagyon friss, de már néhány ilyen támadó jellegű zaj keresésére (illetve kivédésére) alkalmas algoritmust kidolgoztak, ezek közül mutatjuk be a legfontosabbakat a következő fejezetekben.

2 Gépi tanuló modellek elleni támadások taxonómiája

Napjainkban a képfelismerési feladatokra nagyon pontos megoldást tudnak nyújtani a mély neurális hálózatok mindaddig, amíg a bemeneti minták a tanítóhalmaz eloszlásából származnak. Azonban, ha a betanított modellnek olyan bemeneteket adnak, amik hasonlítanak ugyan egy-egy eredeti eloszlásból származó képre, de szándékosan egy irányított zajt adnak hozzá, akkor azokat a modell tévesen rossz osztályba sorolhatja. Ez az érzékenység nem kívánatos viselkedés, hiszen egy támadó fél ezt kihasználva (különösen ha a célmodellről mindent tud) kárt tud okozni az ilyen megoldásokat alkalmazó rendszerekben.

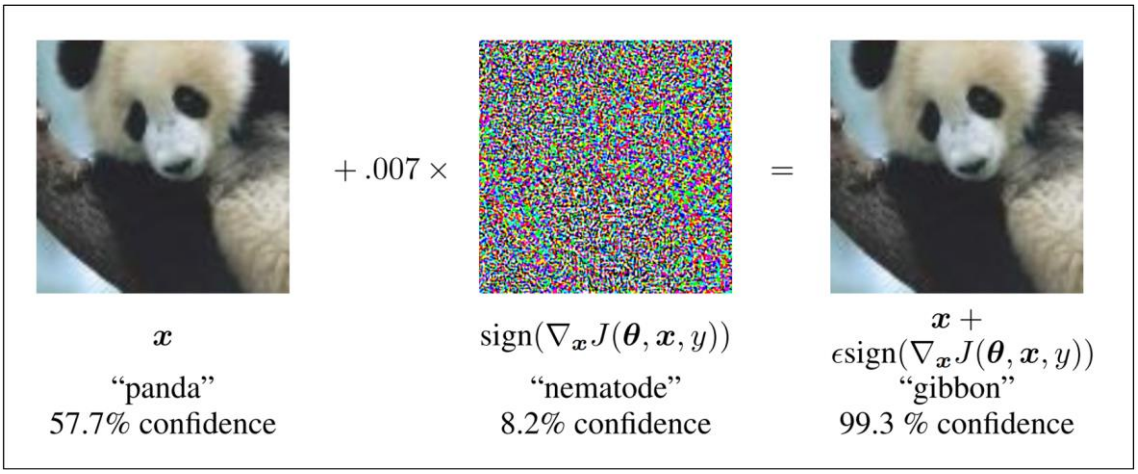
A black-box szcenáriókban a támadók nem szerezhetnek információkat a célmodellről, de kialakíthatnak egy helyi helyettesítő modellt (helyi alatt itt azt értjük, hogy a támadó a saját környezetében építi ezt fel) a célmodell lekérdezésével, a bemeneti-kimeneti párok felhasználásával vagy egy modell inverziós módszerrel. Papernot munkatársaival [27] szintetikus bemeneteket használt egy helyi helyettesítő modell kialakításához, majd hozott létre támadó mintákat. Fredrikson szerzőtársaival modell inverziós támadást [11] hajtott végre, amely a gépi tanulás alkalmazás-programozási interfészeit (API) használta az érzékeny jellemzők kikövetkeztetésére. Tramèr munkatársaival [38] sikeres modellkinyerési támadást mutatott be olyan online gépi tanulást biztosító szolgáltatók ellen, mint a BigML és az Amazon Machine Learning.

Szegedy Krisztián [36] elsőként (első kutatók között) javasolt egy olyan koncepciót egy kiválasztott kép (bemeneti minta) módosítására, hogy apró zajt adnak hozzá, amelyet az emberi szem nem vesz észre, és a célmodell mégis nagy magabiztossággal tévesen fogja osztályozni. Ezt a következőképpen írhatjuk fel formálisan: tegyük fel, hogy létezik egy gépi tanulási modell (M) és egy eredeti *x* minta, amelyet a modell helyesen osztályoz, azaz $M(x) = y_{true}$; enyhe zajt adva az *x*-hez, a támadó elkészítheti az *x*' adversarial mintát, amely hasonló az *x*-hez, de M-vel tévesen osztályozza, azaz

$$M(x') \neq y_{true}$$

Goodfellow szerzőtársaival megmutatta, hogy az adversarial minták oka a lineáris viselkedés a nagy dimenziós térben [14]. Nagy dimenziós lineáris osztályozóban minden bemeneti jellemző normalizálódik, így az egyes bemenetek egy dimenziójának kis

változtatása még nem változtatja meg az osztályozó előrejelzését, míg a bemenetek minden dimenziójára vonatkozó kis zavarok már hatékony változtatáshoz vezetnek. Az adversarial minta alapú támadásra egy példa az 1. ábrán látható. Az osztályozó modell az eredeti baloldali képet „pandának” tekinti (57,7% konfidenciával). Egy kicsi irányított zaj (a zajt csupán 0,007 súllyal figyelembe véve) hozzáadásával ugyanez a modell a jobb oldalon látható képet „gibbonnak” osztályozza (99,3% konfidenciával), míg az emberi szem nem képes különbséget tenni a két kép között.



1. ábra: Egy pandát ábrázoló képhez kis zajt hozzáadva az osztályozó döntése gibbon lesz [14]

Az adversarial mintán alapú támadások másik fontos jelensége a transzferabilitás [43]. A transzferabilitás alatt az értjük, hogyha valaki adversarial mintákat állít elő egy M_1 célmodell elleni támadáshoz, akkor nem szükséges megszereznie az M_1 modell architektúráját vagy paramétereit; mivel, ha van egy olyan M_2 modellje, amelyet az M_1 helyettesítőjeként tanítottak be, akkor ezt fel fogja tudni használni (és minél nagyobb a transzferabilitás, annál jobb a helyettesítés).

2.1 Támadási stratégiák tanítási és tesztelési fázisban

A gépi tanulás folyamatához köthető támadások egy része a tanítási fázisban, másik része a tesztelési fázisban érvényesül. A tanítási szakasz támadásai megpróbálják közvetlenül befolyásolni (elrontani) a célmodellt a tanulóállomány megváltoztatásával még a célmodell elkészülte előtt. Azonban ezek inkább csak elméleti lehetőségek, hiszen a tanítást zárt, azaz védett rendszerben végzik, és a gépi tanuló modell tanítása alatt ez jól védhető. A tanítás szakasz támadási stratégiái/típusai három kategóriába sorolhatók:

- Adatinjekció: a támadónak nincs hozzáférése a tanulóállományhoz és a tanuló algoritmusokhoz, de képes új adatokat hozzáadni az adathalmazhoz. A támadó ezt speciális minták hozzáadásával teszi meg.
- Adatmanipuláció: a támadó nem fér hozzá a tanuló algoritmusokhoz, de tanulóállományhoz igen. A támadó ilyenkor „megmérgezheti" [2] a tanulóállományt azáltal, hogy még azelőtt módosítja az adatokat, mielőtt azt a célmodell tanítására használnák.
- Modellmanipuláció (logikai korrupció): a támadó hozzáférhet és módosíthatja a tanuló algoritmust, amit a célmodell előállításánál használnak.

A tesztelési szakasz támadásai megpróbálják megtéveszteni a kész célmodellt egy megfelelően előkészített bemeneti adattal. A kész célmodell alatt értjük azt a gépi tanuló modellt, amit tulajdonosa a korábbi (tanítási) fázisban betanított; a célmodellt ezek után használják (a használatot hívjuk a gépi tanulás szakirodalmában tesztelési fázisnak / szakasznak), amelynek során bemeneti adatokat vár (ez a bemeneti lehetőség általában publikus, ami egyben támadási lehetőséget kínál). A tesztelési szakasz támadási stratégiái/típusai a következők:

- Adversarial minták: a támadó nem módosíthatja sem a tanuló algoritmust, sem a tanulóállományt, viszont a tesztelési fázisban olyan adatot adhat a gépi tanuló bemenetére, amely szándékosan megtéveszti a gépi tanuló modellt.
- Modellinverzió: a támadó nem módosíthatja sem a tanuló algoritmust, sem a tanulóállományt; a támadó célja nem a megtévesztés, hanem a bemeneti adatok visszanyerése a kimeneti adatokból.
- Modellkinyerés: a támadó nem módosíthatja sem a tanuló algoritmust, sem a tanulóállományt; a támadó célja a modell titkos működésének felfedése (általában egy későbbi megtévesztés céljából, azaz azért, hogy befolyásolni tudja, az immár felfedett belső működést).

1. táblázat: Gépi tanulási folyamatot érintő támadások típusai

Típus	Tanítási fázis	Teszt fázis	Adat általi támadás	Modell általi támadás	Információ feltárás
adatinjekció	X		X		
adatmanipuláció	X		X		
modellmanipuláció (logikai korrupció)	X			X	
adversarial minták (bemenetmanipuláció)		X	X		
modellinverzió		X			X
modellkinyerés		X			X

A fent említett támadási stratégiákat/típusokat összevetve, az 1. táblázatban láthatjuk, hogy 3 támadási stratégia a tanítási fázisra, 3 pedig a tesztelési fázisra vonatkozik. Egyes típusoknál a támadási stratégia az adat módosításával éri el a célját, míg mások a modell módosításával érik azt el, hogy a modell kimenete a támadó számára megfelelő legyen. Az utolsó oszlopban az információ feltárás látható, ahol a támadó célja nem a modell kimenetének megváltoztatása (megtévesztés); hanem a bemeneti adatok vagy a modell felfedése, azaz a titkosságot veszi célba. Ez tehát nem tartozik közvetlenül a modell megtévesztési témakörbe, mégis fontos szerepe lehet (közvetett módon) a támadásnál, mivel a támadó a felfedett bemeneti adatokból és/vagy modellből származó információkat felhasználhatja majd később adversarial minták előállításánál. A következőkben ezeket a felsorolt típusokat vesszük sorba és nézzük meg őket részletesen.

Az adatinjekciónál a támadó nem tud hozzáférni a tanulóállományhoz, hogy azokat módosítsa, de joga van új adatokat hozzáadni az adathalmazhoz. Például egy támadó hamis adatokat közölhet a pénzügyi idősoros előrejelzési modellek befolyásolása érdekében.

Az adatmanipuláció, más néven „mérgezés” [3][20] vagy „causative” támadás [2] a gépi tanuló modell elleni manipulációs támadás a tanítási fázis során, pontosabban még a tanítás megkezdése előtt. A támadó módosítja a gépi tanuló modell által használandó tanulóállományt (például az osztálycímkéket és/vagy az adatokat, tartalmakat) annak érdekében, hogy befolyásolja annak viselkedését.

Modellmanipuláció (illetve hívják még logikai korrupciónak is) esetében a támadó hozzáférhet és módosíthatja a tanuló algoritmust. Bár elképzelhető olyan eset, amikor a támadó egy fehérdobozos modellt tesz közzé, amelyet harmadik felek

akaratlanul is elfogadnak, majd később a támadó ezt „modell” szintjén kihasználja; a gyakorlatban ilyen modellmanipulációt azonban nem említ a szakirodalom. A mélytanulási közösségben általában nyílt forráskódú licenc alapján adnak ki modelleket; a kód újrafelhasználása és az átviteli tanulás elterjedtsége miatt ez a fajta eset a későbbiekben mégis potenciális támadási felület lehet, így a modellmanipulációból fontos vizsgálati terület alakulhat ki a szakirodalomban.

A bemenetmanipuláció, vagy ahogy a nemzetközi szakirodalom hívja adversarial példák (adversarial minták) [14], egy manipulációs támadás egy gépi tanuló modell ellen tesztelési fázis, azaz működés közben. Ebben az esetben a támadó olyan adatot ad a gépi tanuló bemenetére, amely nagy valószínűséggel más kimenetet eredményez, mint amelyet eredetileg adott volna. Ilyen például egy olyan módosított stop jelet ábrázoló kép, amelyet sebességkorlátozó jelnek észlel a modell [10].

A modellinverzió (más néven bemenetkinyerés) azokra az esetekre vonatkozik, amikor a modell kimenete nyilvános, de a benenet titkos, és a támadó megpróbálja visszaszerezni a bemeneteket a kimenetekből [11]. Például az orvosi dokumentumok jellemzőinek kinyerése a gépi tanuló modell által ajánlott gyógyszeradagolásból, vagy egy arc felismerhető képének elkészítése, amely csak az azonosító számot (arcfelismerési modellben az osztálycímkét) és az osztálycímkéhez tartozó konfidenciát / valószínűségi pontszámot adja meg. A bemenetkinyerésnél két alesetet is megkülönböztethetünk: az egyiknél egy-egy konkrét bemenet, addig a másiknál egy nagyobb adathalmaz (tipikusan a teljes tanulóállomány) megszerzése a cél. Ehhez a modellinverzió típusú támadásokhoz kapcsolódik a privacy-preserving [37] gépi tanulók szakirodalma is, hiszen ezek foglalkoznak a bemeneti információk elfedésével.

A modellkinyerésnél a támadó egy teljesen feketedobozos gépi tanuló modellt vesz célba, megkísérelve „kinyitni”, belelátni a belső működésbe és lemásolni annak viselkedését vagy paramétereit. A modellkinyerés egy másnak a tulajdonában levő modell lopásaként is felfogható [38], és a modellkinyerés teszi lehetővé a fehérdobozos támadásokat egy feketedobozos modell ellen.

2.2 Fehérdobozos és feketedobozos támadások

A gépi tanuló modell használata során, azaz a tesztelési szakaszban zajló adversarial támadások nem befolyásolják a célmodellt, hanem helytelen kimenetek létrehozására kényszerítik. Az ilyen támadások hatékonysága elsősorban attól függ, hogy

a támadó rendelkezésére állnak-e információk a modellre vonatkozóan. A tesztelés során a támadások feloszthatók fehérdobozos és feketedobozos támadásokra.

Az (*x*,*y*) bemeneti párokból (ahol *x* a bemeneti minta és *y* az elvárt kimenet) álló tanulóállományon betanított célmodellt jelöljük *f*-el! A betanított f modell nem csak a tanuló algoritmussal, hanem konkrét paraméter értékekkel is rendelkezik (ezek halmazát θ-val jelölik), amelyek a tanulóállományon kívül – a legtöbb gépi tanulónál – véletlenszerű kezdeti értékektől is függnek. A betanított f modell, mint célmodell ismeretétől és a teljes tanulóállomány ismeretétől függően a fehérdobozos és feketedobozos támadásokat további altípusokra bonthatjuk, ahogy ez a következő táblázatban látható.

2. táblázat: Fehérdobozos és feketedobozos támadások típusai

Támadás típusa	**Információ a célmodellről**	**Ismeri-e a tanuló-állományt?**	**Létrehozhat-e új bemeneti példát?**
Fehérdobozos	van	igen/nem	igen
Nem adaptív feketedobozos	nincs	igen	igen/nem
Adaptív feketedobozos	nincs	nem	igen
Korlátozottan adaptív feketedobozos	nincs	nem	nem

Fehérdobozos (white-box) támadások: ha gépi tanuló modellje publikus, akkor ezt fehérdobozos támadásnak nevezzük. Ezekben az esetekben a támadók teljes ismeretekkel rendelkeznek az f célmodellről, beleértve az algoritmust, az adateloszlást és a θ paramétereket. A potenciális támadók a rendelkezésre álló információk felhasználásával feltérképezik a célmodell paraméterterét, és megvizsgálják, hogy hol lehet a legkisebb változtatással a legnagyobb rossz irányú kimeneti változást elérni (ezek lesznek az adversarial minták). A fehérdobozos támadások a modellparaméterek ismerete miatt erős támadásnak számítanak.

Feketedobozos (black-box) támadások: ha gépi tanuló modellje nem publikus, akkor ezt feketedobozos támadásnak nevezzük. Ezekben az esetekben a támadóknak nincsenek ismereteik az *f* célmodellről a feketedobozos támadásoknál. Ehelyett elemzik a modell sebezhetőségét a korábbi bemeneti / kimeneti párokról szóló információk felhasználásával például úgy, hogy megfigyelik egy adott bemenetre milyen kimenetet ad ki a modell. A fehérdobozos támadásokhoz képest a feketedobozos támadásoknak nem

szükséges megtanulniuk a célmodell paramétereit (vagy a véletlenszerűségét). A black-box támadások tovább bonthatók „nem adaptív”, „adaptív” és „korlátozottan adaptív” feketedobozos támadási típusokra [29].

Nem adaptív feketedobozos támadás: A támadó csak ahhoz a tanulóállományhoz férhet hozzá, amelyen a célmodellt tanították, de nincs információja arról, hogy a tanítást milyen módszerrel és hogyan végezték. Ezért a támadó a saját helyi környezetében egy helyettesítő modellt (gépi tanuló modellt) épít fel, ami jelen esetben nem a célmodellt fogja tökéletesen helyettesíteni (hiszen nincs arról információja, hogy a célmodell mennyire jól tanulta meg a tanulóállományban található összefüggéseket), de mégis jó közelítéssel a támadott modellhez valamennyire hasonló eredményre fog jutni. A helyettesítő f_h modellen fehérdobozos támadási stratégiák alkalmazásával kikísérletezheti, hogy mely adversarial minták tévesztik meg az f_h modellt, majd ezeket támadásként alkalmazza az f modell ellen, hogy téves osztályozási eredményekre vezessen.

Adaptív feketedobozos támadás: A támadó nem férhet hozzá a célmodell semmilyen információjához, és egyszerre a teljes tanulóállományhoz sem, viszont az f modellt működés közben fel tudja használni. Ez azt jelenti, hogy bármilyen x adatot be tud adni a célmodellnek (f modell bemenetére juttatva x-et), és le tudja olvasni az f modell kimenetén a választ (y-t). Így (x,y) párok begyűjtésével fel tudja térképezni a célmodell viselkedését, és egy valódi helyettesítő f_h modellt tud felépíteni a célmodell szimulálására. Ezek után – ugyanúgy, ahogy az előző esetben – fehérdobozos támadási stratégiák alkalmazásával az eredeti célmodell ellen is tud megtévesztő mintákat létrehozni.

Korlátozottan adaptív feketedobozos támadás: A támadó nem férhet hozzá a célmodell semmilyen információjához, és egyszerre a teljes tanulóállományhoz sem, viszont hasonlóan az adaptív feketedobozos támadáshoz itt is feltérképezheti a célmodell viselkedését (x,y) párok begyűjtésével. Viszont abban különbözik a sima adaptív változattól, hogy nem változtathatja meg az x bemeneteket a kimenetek változásainak megfigyeléséhez. A két adaptív típus közös sajátossága, hogy a célmodell lekérdezésével gondosan kiválasztott adatkészletet alkalmaznak és használnak fel a későbbi támadásokhoz.

2.3 Félreosztályozási típusok

A támadó osztályozási modellre gyakorolt hatásának szándéka szerint háromféle célt különböztethetünk meg: konfidencia csökkentése, félreosztályozás (untargeted misclassification), célzott félreosztályozás (targeted misclassification).

Konfidencia csökkentés: a támadó megpróbálja csökkenteni a célmodell előrejelzésének konfidenciáját, például különböző képeken látható számok osztályozásánál az 5-ös számjegynek ne 95%-os legyen a valószínűsége, hanem alacsonyabb.

Félreosztályozás (pontosabban: nem célzott félreosztályozás): a támadó megpróbálja megváltoztatni a bemenetre adott minta kimeneti döntését az eredeti osztálycímkéről egy tetszőleges másik osztályra, például az 5-ös számjegy helyett bármelyik másik számjegyre.

Célzott félreosztályozás: a támadó megpróbálja megváltoztatni a bemenetre adott minta kimeneti döntését az eredeti osztálycímkéről egy előre rögzített másik osztályra, például az 5-ös számjegy helyett a 9-es számjegyre. Ennek kétféle alesete is létezik: az egyiknél a bemeneti minta eredeti osztálya befolyásolja a célosztályt (az 5-ösből 9-es legyen, de például már a 3-asból 6-os), a másiknál viszont a bemenettől függetlenül mindig ugyanaz a cél (mindig 9-es számjegyet adjon a kimeneten).

Ebben a fejezetben bemutatott támadási stratégiák és típusok tárgyalása után a következő fejezetben azokat a konkrét módszereket ismertetjük, melyek az adversarial mintákon alapulnak.

3 Adversarial mintákon alapuló támadási módszerek

Ahogy azt az előző fejezetben bemutattuk, a potenciális támadások a gépi tanulási folyamat tesztelési fázisában a legvalószínűbbek. Adversarial támadáskor (azaz adversarial mintákon alapuló támadáskor) a támadó olyan (minimális) perturbációt / zajt keres, amit a bemenetre keverve a modell kimenetét tetszőlegesen befolyásolhatja. Általában a perturbáció keresésekor a támadó megelégszik egy szuboptimális megoldással, amivel a kívánt osztályba juttatható el a minta. Az adversarial minták létrehozásának folyamata felbontható 2 lépcsőre [29]: irányérzékenység becslésére és perturbáció kiválasztására.

- Irányérzékenység becslésénél a támadó az egyes bemeneti jellemzők változásának érzékenységét vizsgálja, hogy meghatározza a vizsgált *x* minta körül azokat az irányokat a jellemzők terében, amelyekben az *f* modell a legérzékenyebb és valószínűleg a kimenetet is megváltoztathatja. A végső cél megtalálni az *x* azon dimenzióit, amelyek minimális változtatás mellett biztosítják a megtévesztés eredményét.
- Perturbáció kiválasztásánál ezután a támadó az információk ismeretében kiválaszt egy olyan *dx* perturbációt, amely a lehető leghatékonyabb változást (adversarial támadást) eredményezi.

Mindkét lépés minden új iteráció elején az *x*-et *dx*+*x*-el helyettesíti, amíg a perturbált (irányított zajjal ellátott) minta ki nem elégíti az adversarial támadó célját. A teljes perturbációnak (melyet az eredeti mintához adnak hozzá az adversarial minta létrehozása érdekében) a lehető legkisebbnek kell lennie, hogy az adversarial mintát ne lehessen emberi észleléssel felismerni (pl. képek esetén ne lehessen látni a beavatkozást).

A feketedoboz szcenárióban a támadó nem férhet hozzá a célmodell belső struktúrájához és paramétereihez, de ki tud alakítani egy helyettesítő modellt, amely a célmodellhez hasonló „viselkedésű” lesz. Ezután bármilyen fehérdobozos támadási stratégia alkalmazható a helyettesítő modellre, hogy adversarial mintákat állítson elő, majd megtévessze a célmodellt ezen minták transzferálásával. A helyettesítő modell kialakítására a következő lehetőségek adódnak:

- vagy a teljes tanulóállományhoz (amin a célmodell tanult) férhet hozzá a támadó, ld. a korábban tárgyalt nem adaptív feketedobozos támadás;
- vagy a célmodell külső működését (be és kimeneti párok) tudja megfigyelni, ebben az esetben modellkinyeréssel megoldható működésének feltárása.

Modellkinyerés: ilyet például Tramèr [38] mutatott be szerzőtársaival az online, azaz felhő alapú ML (machine learning – gépi tanuló) szolgáltatók ellen. Az „ML-as-Service" szolgáltatók által biztosított ML API-k a legtöbb esetben pontos megbízhatósági értékeket és osztálycímkéket adnak vissza. Ezt használták ki az említett publikációban és a BigML [4] és az Amazon Machine Learning [1] elleni sikeres modellkinyerést hajtottak végre, azaz le tudták másolni a gépi tanuló modell belső működését. A helyettesítő modell elkészítése után következik a transzferabilitás kihasználása a feketedoboz támadásoknál; azaz azokat az adversarial példákat, melyek a helyettesítő modellt megtévesztették, a célmodell ellen is használni tudták (a transzferabilitás tulajdonságból adódóan nagy valószínűséggel szintén sikeresen).

A következőkben a konkrét támadási módszereket (algoritmusokat) mutatjuk be egyenként.

L-BFGS

Szegedy és mtsai. [36] bevezették be az adversarial minta kifejezést és formalizálták a minimalizálási problémát, ahogy azt az alábbi képlet mutatja (a BFGS módszer elnevezése a szerzők: Bruna, Fergus, Goodfellow, Szegedy/Sutskever kezdőbetűiből állt össze).

$$X_* = X + \arg\min_{\delta X} \{ \|\delta X\| : f(X + \delta X) \neq f(X) \}$$

Mivel a probléma összetett, ezért egy egyszerűsített változatot vizsgáltak, ahol a minimális veszteségfüggvény módosítást keresték a félreosztályozáshoz. Bár ennek a módszernek jó a megtévesztési teljesítménye, de futási időben költséges az adversarial minták kiszámítása.

FGSM

A Fast Gradient Sign Method (FGSM) [14] egy gyors gradiens jel számítási módszer, amely használható white- és black-box támadási modellek esetén is (utóbbinál szükséges egy helyettesítő modell betanítása, a megtévesztendő modell

transzferabilitásától függően lesz hatékony az FGSM ilyen esetben). Az osztályozó modell – melynek paramétereit θ jelöli – feladata a bemenet (x) a megfelelő osztályba (y) sorolása N osztály közül. Első lépésben egy betanított modellt (ami lehet a megtévesztendő, ú.n. célmodell vagy lehet egy helyettesítő modell) felhasználva kiszámoljuk az N osztály feletti eloszlást (azaz a modell szerint melyik osztályba, milyen valószínűséggel tartozik a bemenet), majd definiálunk egy veszteségfüggvényt (J), ami a helyes osztálytól való eltérést bünteti (általában keresztentrópiát használunk), végül kiszámítjuk ennek gradiensét a bemenet (x) szerint. Ebből kapjuk meg azt az irányt, ami felé módosítva a bemenetet a legnagyobb növekedést érhetünk el a hibafüggvényen (azaz a bemenetet így lehet legkisebb változtatással hibás osztályba juttatni). Az FGSM a bemenet egy ℓ_∞ határolt környezetében keresi a támadó perturbációt úgy, hogy a bemenet szerint számított gradiensre alkalmazott előjelfüggvénnyel kapott irányba ε-nyit lép (az ε egy hiperparaméter az irányított zaj amplitúdójának kontrollálásához):

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} + \varepsilon \cdot \text{sign}\big(\nabla_{\boldsymbol{x}} J(\boldsymbol{x}, y, \boldsymbol{\theta})\big)$$

Iterative FGSM

Az Iterative Fast Gradient Sign Method (Iterative FGSM, vagy röviden I-FGSM) módszer az FGSM kiterjesztett változata [22]. Az FGSM módszer iterált alkalmazásával keres egy optimális zajt a bemenet ℓ_∞ környezetében. Ez lényegében azt a gradiens módszeres eljárást futtatja, amit az FGSM-nél bemutattunk; az iterációs lépések a következő két egyenleten alapulnak (ahol *clip* egy olyan levágási módot jelöl, melyet a legalsó egyenlet mutat):

$$\tilde{\boldsymbol{x}}_{k+1} = \text{clip}_{x,\varepsilon}\left(\tilde{\boldsymbol{x}}_k + \varepsilon \cdot \text{sign}\left(\nabla_{\tilde{\boldsymbol{x}}_k} J(\tilde{\boldsymbol{x}}_k, y, \boldsymbol{\theta})\right)\right)$$

$$\tilde{\boldsymbol{x}}_0 = \boldsymbol{x}$$

$$\text{clip}_{x,\varepsilon}(A_{i,j}) = \begin{cases} X_{i,j} + \varepsilon \; ha \;\; A_{i,j} > X_{i,j} + \varepsilon \\ X_{i,j} - \varepsilon \; ha \;\; A_{i,j} < X_{i,j} - \varepsilon \\ A_{i,j} \qquad\qquad egyébként \end{cases}$$

PGD

A Projected Gradient Descent (PGD) [22] módszer ugyanúgy működik, mint az iteratív FGSM azzal a különbséggel, hogy $\tilde{x}_0$-t nem az eredeti bemenetre (x-re) inicializálja, hanem véletlenszerűen választja annak ℓ_∞ környezetéből.

JSMA

Jacobian Based Saliency Map (JSMA): Papernot szerzőtársaival [28] egy olyan módszert javasolt, mely az érzékenység irányának megkeresését a modell Jakobi mátrixának felhasználásával végzi. Ez a módszer közvetlenül biztosítja a kimeneti komponens gradiensét az egyes bemeneti komponensekhez viszonyítva, és a megszerzett ismereteket az adversarial minták előállítására használják. Ez a módszer limitálja a perturbáció ℓ_0 normáját, ami azt jelenti, hogy a képnek csak néhány pixelét módosíthatja. Ez a módszer különösen a célzott félreosztályozásos támadásoknál használatos.

One Pixel Attack

Su cikkében [35] egy olyan támadási módszert javasoltak, amelynél a képen csak egy pixel értéket változtatnak meg. A differenciális evolúciós algoritmus segítségével minden képpontot iteratívan módosítottak egy új kép létrehozásához, és összehasonlították az eredeti képpel, hogy megtartsák a legjobb támadási hatással rendelkező új képet a kiválasztási kritériumok szerint az adversarial támadások megvalósításához.

DeepFool

Egy konferencia publikációban [24] a szerzők azt javasolták, hogy adversarial mintákhoz az *x* mintához legközelebb eső döntési határt kell megkeresni. Az FGSM epszilon paraméterének kiválasztásával oldották meg ezt a problémát, és a nemlineáris döntési függvények elleni támadást több lineáris függvény approximációjával valósították meg. Megmutatták, hogy az általuk generált zajok kisebbek, mint az FGSM támadásnál, és mégis hasonló megtévesztési arányt tudnak elérni.

HOUDINI

Cisse szerzőtársaival [7] kidolgozott egy HOUDINI nevű módszert, amely megtéveszti a gradiens alapú gépi tanulási algoritmusokat. A módszer a differenciálható veszteségfüggvényének gradiensinformációit használta az adversarial minták generálásához. Megmutatták, hogy a HOUDINI nemcsak képosztályozásra alkalmas neurális hálózat, hanem beszédfelismerő neurális hálózat megtévesztésére is használható.

CW támadási módszer

Carlini és Wagner egy olyan módszert publikáltak [5], amelyet a nevük kezdőbetűiből neveztek el CW támadásnak. Ennek lényege, hogy egy optimalizálási

feladatként értelmezték a megváltoztatott (x-ről $x + \delta$-ra változtatott) bemenet előre meghatározott t osztályba való juttatását (ahol a t osztálycímke különbözik az x eredeti osztálycímkéjétől). Az optimalizálásnál egy távolság függvény szerint minimális távolság értéket kerestek azon kényszerfeltétel mellett, hogy az $x + \delta$ bemenet egy n dimenziós, 0 és 1 közé normalizált vektor, ahogy az az alábbi képletekben látható.

$$minimalizálni\colon D(x, x + \delta) \quad \text{távolságot úgy, hogy}$$

$$C(x + \delta) = t \qquad \text{és} \qquad x + \delta \in [0, 1]^n$$

Ez átalakítható a következő formára, ahol f egy jól megválasztott célfüggvény, c egy konstans, a p-normára pedig a szerzők L_0, L_2 és L_∞ normák vizsgálatait, azaz támadási alternatíváit is elvégezték:

$$minimalizálni\colon \|\delta\|_p + c \cdot f(x + \delta) \quad \text{kifejezést úgy, hogy}$$

$$x + \delta \in [0, 1]^n$$

4 Védekezési módszerek az adversarial támadások ellen

4.1 Védekezés az adatok módosításával

Az adversarial támadások elleni védekezési módszerek egyrészt az adatok módosításával, másrészt a modellek módosításával tudják elérni, hogy a megtévesztés ne legyen annyira sikeres, mint amit a támadó szeretne. Az adatok módosításának módszerei egyrészt magában foglalják a tanításhoz használt adathalmaz megváltoztatásának lehetőségét még a tanítás előtt, másrészt a bemeneti adatok módosítását a tesztelési szakaszban. Az adatok módosítására a következő módszerek léteznek [29]:

- adatbővítés és adversarial tanítás
- adattömörítés
- adatrandomizálás

Adatbővítés és adversarial tanítás

Az első módszer az adatbővítés, amikor „legális" adversarial példákkal bővítik a tanulóállományt a célmodell robusztusságának javítása érdekében [36]. Huang [19] a célmodellt például azzal javította, hogy a rosszul osztályozott adversarial mintákat jobban büntette (azaz nagyobb súllyal vette be őket az átlagos hiba számításába); irreális elvárás azonban az összes ismeretlen támadási mintát belevenni az adversarial tanításba. Mivel a támadó által előállított adversarial mintákkal a modellt nem tanítottuk, így a tanult osztályozó függvényre ezeken a helyeken direkten nincsenek korrekciók. Ennek következtében lehet az, hogy ilyen mintákra a modell válasza teljesen eltérő a legközelebbi valós mintákhoz képest. Ezt a jelenséget publikálták egy képosztályozás problémán demonstrálva [14] és a cikkben egy adversarial tanítási módszert javasoltak a robusztus modell tanítására a következőképpen: a tanító mintahalmazt virtuálisan augmentálják (kibővítik) olyan mintákkal, amik tartalmaznak támadó perturbációt. Az augmentálásnál osztálycímkéket úgy kell beállítani, hogy azok megegyezzenek az eredeti minták osztálycímkéivel (azaz a helyes, elvárt kimenettel). Majd az így kibővített tanulóhalmazon történik a tanítás, melyet a veszteségfüggvény kiegészítésével érnek el, ahogy az az alábbi képletekben látszik:

$$J'(\boldsymbol{x}, y, \boldsymbol{\theta}) = \alpha J(\boldsymbol{x}, y, \boldsymbol{\theta}) + (1 - \alpha) J(\tilde{\boldsymbol{x}}, y, \boldsymbol{\theta})$$

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} + \varepsilon \cdot \text{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{x}, y, \boldsymbol{\theta})\right) \quad \text{(FGSM)}$$

$$\alpha \in [0; 1]$$

Adattömörítés

A következő módszer az adattömörítés, amely a bemeneti adatok tömörítésével próbálja meg javítani a robusztusságot. Dziugaite [9] megállapította, hogy a JPG tömörítési módszer a képeknél képes javítani az FGSM támadás által csökkent felismerési pontosságot. Das szerzőtársaival hasonló JPEG tömörítési módszert használt az FGSM és a DeepFool támadások elleni védekezésnél [8]. Ezeknek a módszereknek az a korlátja, hogy egy erős tömörítés az eredeti képosztályozás pontosságának csökkenéséhez vezet, míg egy kis tömörítés gyakran nem elegendő a támadó zaj hatásának megszüntetésére. Továbbá ezek a képtömörítési technológiák azonban még mindig nem elég hatékony védekezésnek számítanak egy erősebb támadás, mint például a Carlini és Wagner támadás ellen [5].

Adatrandomizálás (adatok véletlenszerűsítése)

Az adatmódosítás módszerei közül az utolsó az adatok randomizálása [41]; ehhez Wang a szerzőtársaival [40] egy olyan – a neurális háló modelltől elkülönített – adatkonverziós modult használt a képosztályozó elleni támadás kiküszöbölésére, amely a tanítás folyamata során adatbővítési műveleteket hajtott végre. Ilyen adatbővítési művelet például néhány Gauss-féle véletlenszerű zaj hozzáadása. De ezen kívül néhány véletlenszerű textúrát is hozzá lehet adni a képhez, illetve az adversarial minták véletlenszerű átméretezése szintén csökkentheti a támadás hatékonyságát.

4.2 Védekezés a modell módosításával

A fentebb említett módszerek után bemutatjuk azokat a lehetőségeket, amikkel előzetesen védekezhetünk egy megfelelő modellmódosítás segítségével. A modellmódosítás a gépi tanuló célmodellek (legtöbbször neurális hálózatok) módosítására utal, és – ahogy az alábbi felsorolásból látszik – több típusra osztható:

- gradiens elrejtés
- transzferabilitás blokkolás
- regularizáció
- defenzív desztilláció

- jellemzők dimenziószámának csökkentése
- támadó zaj csökkentése
- maszkréteg használata

Gradiens elrejtés

Az első módszert gradiens elrejtésnek nevezzük [39]; mely elrejti a célmodell gradiens információit a külső megfigyelők (támadók) elől; így ez egy természetes védekezés a gradiens alapú támadásokkal (mint például az FGSM) szemben. Ha a célmodell nem differenciálható (például egy döntési fa, egy legközelebbi szomszéd osztályozó vagy random forest), akkor a gradiens alapú támadás sikertelen lesz. Ha azonban a támadó készít egy olyan helyettesítő feketedoboz modellt, amelynek belső szerkezetében differenciálható függvényeket használ (ilyen tulajdonsággal bír az összes neurális hálózat), majd e modell által létrehoz adversarial mintákat [27], akkor ebben az esetben a módszer könnyen megtéveszthető, azaz a célmodell ellen az így létrehozott adversarial minták felhasználhatók.

Transzferabilitás blokkolás

Mivel a modellek közti transzferabilitási tulajdonság akkor is fennáll, ha a neurális hálózatok eltérő architektúrával rendelkeznek, vagy diszjunkt tanulóállományon lettek tanítva a modellek, a következő módszer a transzferabilitás nagyságát próbálja csökkenteni a feketedobozos támadás megakadályozása érdekében. Hosseini és szerzőtársai háromlépcsős Null címkézési módszert [18] javasoltak az adversarial minták átvihetőségének megakadályozása érdekében. Ennek a módszernek az az előnye, hogy a támadó bemenetet (azaz az adversarial mintát) üres (Null) címkeként jelöli meg, ahelyett, hogy az eredeti címkének minősítené.

Regularizáció

A regularizációs módszernek a célja a célmodell általánosítási képességének javítása azáltal, hogy a veszteségfüggvényhez olyan plusz tago(ka)t adunk – ezt nevezik regularizációs vagy más néven büntető tagnak – amelyek büntetik a speciális képességek javulását, és így a modellt jó alkalmazkodóképességgel ruházzák fel ahhoz, hogy ellenálljon a támadásoknak adversarial mintákból álló, ismeretlen adathalmaz esetén. Az egyik publikációban [30] ilyen regularizációs módszereket alkalmaztak az algoritmus robusztusságának javítására, és jó eredményeket értek el a támadásokkal szemben. Ennek

egy speciálisabb esete, amikor a hálózat (ld. Parseval hálózat [6]) hierarchikus regularizációt alkalmaz, hogy kontrollálja a hálózat egy fontos paraméterét, az úgy nevezett globális Lipschitz-állandót.

Defenzív Desztilláció

A következő módszer a Defenzív Desztilláció [25][26], amelynél simább kimeneti felülettel és kisebb érzékenységgel rendelkező modellt hoznak létre annak érdekében, hogy javítsa a modell robusztusságát, és csökkentse az adversarial támadás sikerességét. Ezt a módszert javasolt Papernot munkatársaival a támadások ellen a desztillációs technológia alapján [17], amelyet eredetileg hálózatok tudásának tömörítésére alkalmaztak. A Defenzív Desztilláció védekezési módszer is hasonló elven működik, annyi eltéréssel, hogy a modell architektúráján nem változtatnak (azaz nem tömörítenek), amikor a soft-címkékkel (ez a 0 és 1-eseket tartalmazó diszkrét címkevektor helyett 0 és 1 közötti értékeket tartalmaz) tanítják. A Defenzív Desztilláció lépései a következők:

1. *M* modell inicializálása egy adott *A* architektúra szerint.
2. *M* modell tanítása a tanítóminták és diszkrét osztálycímkéken vett keresztentrópia alapján.
3. Soft-címkék képzése *M* modellel minden tanítómintához.
4. *M'* modell inicializálása *A* architektúra szerint.
5. *M'* modell tanítása a tanítóminták és soft-címkék szerint számított keresztentrópia szerint.

Az *M* és *M'* modell utolsó rétege után egy *softmax* aktiváció található:

$$\text{softmax}(X)_i = \frac{e^{\frac{x_i}{T}}}{\sum_{k=0}^{N-1} e^{\frac{x_k}{T}}}$$

ahol *M* tanításakor a *T* paramétert 1-nek választjuk meg, míg *M'* tanításakor *T* értékét minél nagyobbra célszerű megválasztani, ugyanis belátható [25], hogy a *softmax* függvény *T* paraméterének növelésével a modell Jacobi mátrixának abszolút értékét csökkenthetjük, ami azt jelenti, hogy a modell kimenete kevésbé lesz érzékeny a bemeneten való kis változásokra, így a támadó perturbációra is. Következtetéskor *M'* modell *T* paraméterét visszaállítjuk 1-re (habár a címkék sorrendezését ez nem változtatja).

Jellemzők dimenziószámának csökkentése

A következő módszer a jellemzők dimenziószámának csökkentése [42], amelynek célja az adatábrázolás komplexitásának csökkentése. Kép típusú adatokra például ezt a

következő módokon lehet megtenni: a pixeleknél (képpontoknál) kisebb színmélységet használunk, vagy lekicsinyítjük a képet. Bár ez a módszer hatékonyan megelőzheti az adversarial támadásokat, de csökkenti a valós minták osztályozásának pontosságát is.

Támadó zaj csökkentése

A következő módszer a támadó zaj csökkentése, amikor például egy speciális mély neurális hálózatot (mély tömörítési hálózatot), a DCN (Deep Contractive Network) hálózatot [16] használják zajszűrő autoencoderrel (denoising autoencoders - DAE), és ezzel érnek el védekező hatást.

Maszkréteg használata

Az utolsó módszer egy maszkréteg beillesztése a neurális hálózatba az osztályozási rész előtt [12]. A maszk réteg az eredeti képek és a maszkréteget megelőző rétegek alkotta hálózati modell kimeneti jellemzői közötti különbségek kódolására szolgál. Ez a maszkréteg megtanulja az eredeti képeket és a hozzájuk tartozó adversarial képeket, és kódolja a köztük levő különbségek, valamint az előző hálózati modell réteg kimeneti jellemzői között levő összefüggéseket. Ennek a maszkrétegnek a legfontosabb súlya a hálózat legérzékenyebb tulajdonságának felel meg; ezért az osztályozásnál ezeket a jellemzőket elfedik a súlyok kinullázásával. Ily módon kiküszöbölhető az adversarial minták által okozott eltérés az osztályozásnál.

4.3 A védekezés kétféle módja

Az eddig bemutatott módszereken kívül is vannak még védelmi megoldások, melyek nem sorolhatók be a fenti típusokba. Ilyen egyéb kategóriába tartozik Samangouei a munkatársaival közös cikkében [31] tett javaslata a Defence Generative Adversarial Nets (Defense-GAN) elnevezésű mechanizmusra, amely mind a fehér, mind a feketedoboz támadásokra alkalmazható, hogy csökkentse a támadás hatékonyságát. Ez a módszer a GAN generáló képességét használja ki [13]; a fő gondolata az, hogy a bemeneti képeket áttranszformálja a generátor tartományába a rekonstrukciós hiba minimalizálásával még a kép osztályozása előtt. Így az adversarial minták és az eredeti minták közelebb kerülnek egymáshoz a generátor tartományában, ezáltal nagymértékben csökkentik a potenciális adversarial támadások hatékonyságát. Szintén az egyéb kategóriába sorolható a MagNet nevű keretrendszer [23], amely az osztályozó utolsó rétegének kimenetét feketedobozként használja. A MagNet egy detektort használ, hogy

megkülönböztesse az adversarial mintákat a nem módosított mintáktól. A detektor megméri a vizsgált minta és az n-dimenziós sokaság közötti távolságot, és elutasítja a mintát, ha a távolság meghaladja a küszöbértéket.

A védekezésnek kétféle módja van: az egyik, amikor az osztályozó modellnek mindenképpen dönteni kell, hogy a bemenetére adott minta az eredeti osztályok közül melyikbe tartozik; ez a modell robusztusabbá tételével oldható meg. Másik mód, amikor az osztályozó modellnek lehetősége van a bemeneti minta elkülönítésére (külön válogatására) úgy, hogy nem kell besorolnia azt az az eredeti osztályok valamelyikébe, hanem ezzel az elkülönítéssel tudja jelezni, hogy ez valószínűleg egy adversarial minta (ilyen volt például a korábban bemutatott Null címkézési módszer [18] és a MagNet nevű keretrendszer [23]). Ezt egy szűrési feladatnak is felfoghatjuk, ahol az adversarial mintákat ki kell szűrni (ez bináris osztályozással megoldható); illetve egy n+1 osztályozási feladatnak is tekinthető, ahol n az eredeti osztályok száma (ez többosztályos osztályozással megoldható).

5 Alkalmazási területek

5.1 Arcfelismerés

Sharif munkatársaival a beléptető rendszereknél használt arc biometrikus rendszerekhez egy olyan módszert [33] fejlesztett ki az adversarial támadások végrehajtására, amelynél egy speciális mintázattal ellátott kinyomtatott szemüvegkeretet használtak fel. A szemüvegkeretes arcképet az arcfelismerő algoritmushoz juttatva, a szemüveg lehetővé teszi, hogy elkerülje a felismerést, vagy más személynek adja ki magát, ahogy az a következő ábrán látszik (Reese Witherspoont középen tévesen Russel Crowe-nak ismeri fel a rendszer). Vizsgálatuk középpontjában a fehérdobozos arcfelismerő rendszerek álltak, de azt is bemutatták, hogy hasonló technikák miként alkalmazhatók a feketedobozos forgatókönyvekben. Az előzőektől eltérően a megtévesztés itt nem észrevétlen módon történik, hiszen emberi szemmel jól látszik, hogy egy idegen tárgy jelenik meg az arcon, amely nem az alakja miatt furcsa, hiszen a szemüveget sokan hordanak, hanem a mintázata miatt.

2. ábra: Arcfelismerés elleni támadás speciális szemüveggel [29]

5.2 Támadás a fizikai világban

Kurakin és munkatársai [21] először mutatták be, hogy az adversarial támadás fenyegetései a fizikai világban is fennállnak. Ennek bizonyítására adversarial képeket készítettek, majd kinyomtatták őket és ezekről pillanatfelvételeket készítettek a telefon kamerájával. Az eredmények azt mutatták, hogy a képosztályozó modell a kameraképeket nagyrészt tévesen osztályozta. Ahogy az a következő ábrán látható, egy mosógépet ábrázoló képből készítettek 2 féle erősségű zajjal adversarial képeket, amelyeket egy printerrel kinyomtattak és a nyomtatott képeket mutatták a képosztályozó modellnek. Azt a képet, amit nem módosítottak irányított zajjal, helyesen mosógépnek, az adversarial

zajjal módosított képeket pedig (rosszul osztályozva) páncélszekrénynek ismerte fel a modell.

3. ábra: Támadás a fizikai világban [21]

Egy másik példa a fizikai világban való támadásra az a cikk, amelyben a szerzők bemutatták, hogy fizikai adversarial mintákat állítottak elő a kód alapú írisz-felismerő rendszerek számára [34]. A hagyományos írisz-kódgeneráló algoritmusok azonban nem differenciálhatók a bemeneti kép szerint, ezért egy helyettesítő mély neurális hálózattal oldották meg, hogy a hagyományos algoritmus által létrehozott kódokhoz nagyon hasonló írisz-kódokat állítson elő. Majd helyettesítő modellen kikísérletezve elő tudták állítani az adversarial mintákat.

Számjegyek automatikus felismerése banki környezetben is előfordulhat, amikor egy számlaszámot kell egy képről beolvasni [15]. Ha valaki hozzáfér a képhez és van lehetősége módosítani is azt, akkor adversarial támadás során a számjegyek megváltoztatásával el tudja érni, hogy saját bankszámlájára menjen egy adott összeg.

A Computer Assisted Audit Techniques (más néven Computer-Assisted Audit Tools - CAATs) növekvő terület az IT audit szakmán belül, amikor is számítógépes programokat használnak az informatikai audit folyamatok automatizálására. Ez magában foglalja az egyszerű irodai szoftvereket, és az olyan fejlettebb szoftvercsomagokat is, mint az üzleti intelligencia eszközök, illetve statisztikai elemzésre alkalmas mesterséges intelligenciát tartalmazó programok. Az „Adversarial learning of deepfakes in accounting” című cikkben azt mutatták meg, hogy a vállalatirányítási információs rendszerekben használt CAAT programok is megtéveszthetők adversarial támadással [32].

Irodalomjegyzék

[1] Amazon Web Servives. https://aws.amazon.com/machine-learning. Utolsó megtekintés: 2021.04.01

[2] Barreno, M.; Nelson, B.; Sears, R.; Joseph, A.D.; Tygar, J.D. (2006). Can machine learning be secure? In Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security, Taipei, Taiwan, 21–24 March 2006; pp. 16–25.

[3] Biggio, B., Nelson, B., & Laskov, P. (2012). Poisoning attacks against support vector machines. In Proceedings of the 29th International Coference on International Conference on Machine Learning (pp. 1467-1474).

[4] BigML. https://www.bigml.com. Utolsó megtekintés: 2021.04.01

[5] Carlini, N.;Wagner, D. Towards evaluating the robustness of neural networks. (2017). In Proceedings of the IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2017; pp. 39–57.

[6] Cisse, M., Bojanowski, P., Grave, E., Dauphin, Y., & Usunier, N. (2017). Parseval networks: Improving robustness to adversarial examples. In International Conference on Machine Learning (pp. 854-863). PMLR.

[7] Cisse, M.; Adi, Y.; Neverova, N.; Keshet, J. (2017). Houdini: Fooling deep structured prediction models. arXiv:1707.05373.

[8] Das, N.; Shanbhogue, M.; Chen, S.T.; Hohman, F.; Chen, L.; Kounavis, M.E.; Chau, D.H. (2017). Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression., arXiv:1705.02900.

[9] Dziugaite, G.K.; Ghahramani, Z.; Roy, D.M. (2016). A study of the effect of jpg compression on adversarial images. arXiv:1608.00853.

[10] Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., ... & Song, D. (2018). Robust physical-world attacks on deep learning visual classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1625-1634).

[11] Fredrikson, M., Jha, S., & Ristenpart, T. (2015). Model inversion attacks that exploit confidence information and basic countermeasures. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (pp. 1322-1333)

[12] Gao, J.; Wang, B.; Lin, Z.; Xu, W.; Qi, Y. (2017). Deepcloak: Masking deep neural network models for robustness against adversarial samples. arXiv:1702.06763

[13] Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. (2013). Generative adversarial nets. In Advances in

Neural Information Processing Systems, Proceedings of the Annual Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013; Curran Associates, Inc.: Red Hook, NY, USA, pp. 2672–2680.

[14] Goodfellow, I.J.; Shlens, J.; Szegedy, C. (2014). Explaining and harnessing adversarial examples. arXiv:1412.6572.

[15] Graese, A., Rozsa, A., & Boult, T. E. (2016). Assessing threat of adversarial examples on deep neural networks. In 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA) (pp. 69-74). IEEE.

[16] Gu, S., & Rigazio, L. (2014). Towards deep neural network architectures robust to adversarial examples. arXiv preprint arXiv:1412.5068.

[17] Hinton, G.; Vinyals, O.; Dean, J. (2015). Distilling the knowledge in a neural network. arXiv:1503.02531.

[18] Hosseini, H.; Chen, Y.; Kannan, S.; Zhang, B.; Poovendran, R. (2017). Blocking transferability of adversarial examples in black-box learning systems. arXiv:1703.04318.

[19] Huang, R.; Xu, B.; Schuurmans, D.; Szepesvári, C. (2015). Learning with a strong adversary. arXiv:1511.03034.

[20] Kloft, M., & Laskov, P. (2007). A poisoning attack against online anomaly detection. In NIPS Workshop on Machine Learning in Adversarial Environments for Computer Security (Vol. 19).

[21] Kurakin, A., Goodfellow, I., & Bengio, S. (2016). Adversarial examples in the physical world., arXiv preprint arXiv:1607.02533.

[22] Kurakin, A., Goodfellow, I., & Bengio, S. (2017). Adversarial machine learning at scale. ICLR 2017, arXiv:1611.01236.

[23] Meng, D.; Chen, H. (2017). Magnet: A two-pronged defense against adversarial examples. In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October – 3 November 2017; pp. 135–147.

[24] Moosavi-Dezfooli, S.M.; Fawzi, A.; Frossard, P. (2016). Deepfool: A simple and accurate method to fool deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2574–2582.

[25] Papernot, N., McDaniel, P., Wu, X., Jha, S., & Swami, A. (2016). Distillation as a defense to adversarial perturbations against deep neural networks. In 2016 IEEE symposium on security and privacy (SP) (pp. 582-597). IEEE.

[26] Papernot, N.; McDaniel, P. Extending defensive distillation. (2017). arXiv:1705.05264.

[27] Papernot, N.; McDaniel, P.; Goodfellow, I.; Jha, S.; Celik, Z.B.; Swami, A. Practical black-box attacks against machine learning. (2017). In Proceedings of the ACM on Asia Conference on Computer and Communications Security, Abu Dhabi, UAE, 2–6 April 2017; pp. 506–519.

[28] Papernot, N.; McDaniel, P.; Jha, S.; Fredrikson, M.; Celik, Z.B.; Swami, A. (2016). The limitations of deep learning in adversarial settings. In Proceedings of the 2016 IEEE European Symposium on Security and Privacy (EuroS&P), Saarbrucken, Germany, 21–24 March 2016; pp. 372–387.

[29] Qiu, S., Liu, Q., Zhou, S., & Wu, C. (2019). Review of artificial intelligence adversarial attack and defense technologies. Applied Sciences, 9(5), 909.

[30] Rozsa, A.; Gunther, M.; Boult, T.E. (2016). Towards robust deep neural networks with BANG. arXiv:1612.00138.

[31] Samangouei, P.; Kabkab, M.; Chellappa, R. (2018). Defense-GAN: Protecting classifiers against adversarial attacks using generative models. arXiv:1805.06605.

[32] Schreyer, M., Sattarov, T., Reimer, B., & Borth, D. (2019). Adversarial learning of deepfakes in accounting. arXiv:1910.03810.

[33] Sharif, M.; Bhagavatula, S.; Bauer, L.; Reiter, M.K. (2016). Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016.

[34] Soleymani, S., Dabouei, A., Dawson, J., & Nasrabadi, N. M. (2019). Adversarial examples to fool iris recognition systems. In 2019 International Conference on Biometrics (ICB) (pp. 1-8). IEEE.

[35] Su, J.; Vargas, D.V.; Kouichi, S. (2017). One pixel attack for fooling deep neural networks. arXiv:1710.08864.

[36] Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; Fergus, (2013). R. Intriguing properties of neural networks. arXiv:1312.6199

[37] Szűcs, G. (2013). Decision trees and random forest for privacy-preserving data mining. In Research and Development in E-Business through Service-Oriented Solutions (pp. 71-90). IGI Global.

[38] Tramèr, F., Zhang, F., Juels, A., Reiter, M. K., & Ristenpart, T. (2016). Stealing Machine Learning Models via Prediction APIs. In Proceedings Of The 25th Usenix Security Symposium, pp. 601-618. Usenix Assoc.

[39] Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I.; Boneh, D.; McDaniel, P. (2017). Ensemble adversarial training: Attacks and defenses. arXiv:1705.07204.

[40] Wang, Q.; Guo, W.; Zhang, K.; Ororbia, I.; Alexander, G.; Xing, X.; Liu, X.; Giles, C.L. (2016). Learning adversary-resistant deep neural networks. arXiv:1612.01401.

[41] Xie, C.; Wang, J.; Zhang, Z.; Zhou, Y.; Xie, L.; Yuille, A. (2017). Adversarial examples for semantic segmentation and object detection. arXiv:1703.08603.

[42] Xu, W.; Evans, D.; Qi, Y. (2017). Feature squeezing: Detecting adversarial examples in deep neural networks. arXiv:1704.0115

[43] Zhang, J.; Jiang, X. (2018). Adversarial Examples: Opportunities and Challenges. arXiv:1809.04790

A szerzőről

Dr. Szűcs Gábor, PhD

egyetemi docens, Távközlési és Médiainformatikai Tanszék

Szűcs Gábor 1994-ben végzett villamosmérnökként a BME VIK Karán, és ugyanitt 2002-ben informatikai PhD fokozatot szerzett. Kutatási területei az adattudomány, a mesterséges intelligencia, a mélytanulás, a tartalom alapú képkeresés és a gépi látás. Publikációinak száma meghaladja a 100-at. A HTE Mesterséges Intelligencia Szakosztályának elnöke, a DCLAB (Data Science and Content Technologies) kutatócsoport vezetője.