Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Automatizálási és Alkalmazott Informatikai Tanszék

Gépi tanulási eljárásokkal generált adatok adatvédelmi vizsgálata

Diplomaterv

Készítette
Csarnó Tamás Péter

Konzulens
dr. Gulyás Gábor György

Budapest, 2021

Tartalomjegyzék

HALLGATÓI NYILATKOZAT

Alulírott *Csarnó Tamás Péter*, szigorló hallgató kijelentem, hogy ezt a diplomatervet meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy autentikált felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Budapest, 2021. december 9.

Csarnó Tamás Péter
hallgató

Kivonat

A gépi tanulás területén elért áttöréseknek, és a hardverek fejlődésének köszönhetően egyre szélesebb körben alkalmaznak arcfelismerő rendszereket. Bár az arcfelismerő rendszerek számos alkalmazási területen rendkívül hasznosak, adatvédelmi kockázatokat is hordoznak magukkal. A modern arcfelismerő rendszerek a mély metrika tanulásra építenek. Működésük során a kamerafelvételeken látható arcokból képesek arclenyomatokat készíteni, amelyek az emberi arc jellemzőit kódolják magukban. Az arclenyomat segítségével könnyen beazonosítható a felvételen látható személy.

Munkám során az arclenyomatokhoz kapcsolódó adatvédelmi kockázatok feltárásával, és a kockázatok lehetséges kezelési módszereivel foglalkoztam. Sikerült bemutatnom, hogy az arclenyomatok nem csak azonosításra alkalmasak, hanem személyes adatokat is kódolnak magukban, mint például a felvételen látható személy életkorát, nemét és rasszát. A kódolt adatok jó pontossággal (97-98%) kinyerhetőek gépi tanulási modellek segítségével, amelyek akár publikusan elérhető képadathalmazokon is betaníthatóak.

Dolgozatomban elemeztem hogyan vannak tárolva a demográfiai adatok az arclenyomatokban. Azt tapasztaltam, hogy az arclenyomatokon betanított gépi tanulási modellek rendkívül robusztusok, így a modellek számára fontosnak vélt jellemzők eltávolításával nem lehet leplezni az érzékeny információkat. Egy példán keresztül demonstráltam, hogy az arclenyomatok minimális módosításával lehetséges a gépi tanulási modelleket megtéveszteni, ezzel megakadályozni a személyes adatok kiszivárgását. Végezetül bemutattam két olyan kriptográfiai módszert, a locality sensitive hashing (LSH) technikát, és a homomorfikus titkosítást, amelyek megfelelőek arclenyomatok védelmére.

Abstract

Facial recognition systems are becoming more widely used due to breakthroughs in machine learning and hardware improvements. Although facial recognition systems have great potential in many fields, they carry certain privacy risks. Modern facial recognition systems are based on deep metric learning. These systems can generate face embeddings from the human faces seen on camera images, which are later used for identification.

In the course of my work, I have explored the data protection risks associated with face embeddings and the possible methods of managing the risks. I was able to show that face embeddings are not only useful for identification, but also encode personal information such as age, gender, and race of the person on the image. The encoded data can be extracted with good accuracy (97-98 %) using machine learning models, which can even be trained on publicly available image datasets.

In my thesis I analyzed how demographic data is stored in face embeddings. I found that machine learning models trained on face embeddings are extremely robust, so that sensitive information cannot be masked by removing features that are considered important to the models. Through an example, I have demonstrated that by making minimal changes to the face embeddings, it is possible to deceive machine learning models, thereby preventing the leakage of personal information. Finally, I have presented two cryptographic methods: locality sensitive hashing (LSH) technique, and homomorphic encryption, which are both suitable for protecting face embeddings.

1. Bevezetés

Az elmúlt években a nagyméretű adathalmazok elérhetőségének, a számítási kapacitás exponenciális növekedésének és a mély tanulás területén elért áttöréseknek köszönhetően jelentősen megnövekedett az érdeklődés a gépi tanulás iránt. Manapság a gépi tanulási algoritmusokat előszeretettel alkalmazzák nagy dimenziós bemeneti adatokhoz kapcsolódó osztályozási, regressziós, klaszterezési vagy dimenziócsökkentési feladatokra. A gépi tanulási algoritmusok számos területen embert meghaladó képességekkel rendelkeznek (például képosztályozásban). A mindennapi életünkben használt okos eszközökön a kép- és beszédfelismerést, az internetes keresést, az arcfelismeréses bejelentkezést stb. a gépi tanulási algoritmusok tesznek lehetővé.

A gépi tanulás nagy részben hozzájárult a tudomány és technológia fejlődéshez. Szinte minden iparág és vállalat felismerte a gépi tanulás előnyeit és lehetőségeit. A technológia fejlődése lehetővé tette a közelmúltbeli áttöréseket, amelyek elősegítik a gyorsabb és hatékonyabb üzleti intelligenciát, az arcfelismeréstől a természetes nyelvfeldolgozásig terjedő alkalmazások felhasználásával.

A gépi tanulás alkalmazásai között szerepelnek olyan eljárások, amelyek képesek adatot generálni a bemenetből, mint például a generatív modellek vagy a mély metrika tanulás, de ide sorolhatóak a napjainkban egyre elterjedtebben használt gépi tanulásra épülő arcfelismerési rendszerek is.

A generatív modellek egyik típusa: a GAN (Generative Adversarial Network) olyan neurális hálózatok, amelyek képesek új, szintetikus adatot előállítani. A GAN egy olyan neurális háló, amelyben két alhálózat verseng egymással: a generátor és a diszkriminátor. A generátor feladata a tanító mintákhoz hasonló új minták előállítása, míg a diszkriminátor felel a valódi és a generált minták megkülönböztetéséért. A GAN tanítása során a generátor igyekszik egyre hihetőbb mintákat előállítani, hogy be tudja csapni a diszkriminátort. A GAN-ok sikeres betanítása után, a bemenetre adott véletlenszerű zajból realisztikus képeket tudnak generálni. Például az NVIDIA által fejlesztett StyleGAN3 képes fotorealisztikus emberi arcok generálására [31].

A GAN-ok elsősorban olyan szituációkban lehetnek nagyon hasznosak, amikor több tanítóadatra van szükségünk, viszont az adat gyűjtése nehezen megoldható. Sok esetben az adatgyűjtés és címkézés hosszadalmas, drága folyamat. Amennyiben be tudunk tanítani egy hálót, ami a célnak megfelelő minőségű szintetikus adatok előállítására képes, az megoldást adhat erre a problémára. Az új adatoknak megfele-

lően realisztikusnak kell lenniük ahhoz, hogy a generált adatokból szerzett ismeretek továbbra is érvényesek legyenek a valós adatokra.

A gépi tanulási technikák fejlődésének köszönhetően, illetve az egyre olcsóbb okos eszközök elterjedésével, egyre jobban elterjedt az arcfelismerés használata. A generatív modellekhez hasonlóan, a modern arcfelismerő rendszerek is gépi tanulásra támaszkodnak. Működésük során az emberek arcáról készült digitális képekből képesek arcot jellemző metrikákat előállítani oly módon, hogy egy emberhez tartozó arcleíró vektorok (későbbiekben arclenyomatok) távolsága kicsi legyen, míg különböző emberek között minél nagyobb. Erre a célra létrehozott neurális hálók: a sziámi hálók képesek megtanulni azt a mély metrikát, ami legjobban leírja az emberi arc struktúráját. Miután sikerült betanítani egy hálót ami képes digitális arcképekből arclenyomatokat származtatni, az arclenyomatok összehasonlításával már képes a rendszer összevetni a keresett személyt a rendszer által ismert személyekkel.

Az ilyen módon kinyert arclenyomatokban tárolt információ ember számára nem értelmezhető, csupán egy lebegőpontos számsornak tűnik, de belőlük gépi tanulási módszerekkel személyes adat származtatható a képen látható személyekről. Magukba kódolva olyan információkat is hordoznak, mint például a képen látható illető rassza, a neme, életkora és az arc egyéb jellemzői. Ezek az információk együttesen lehetővé teszik egy személy azonosítását. Az arclenyomatokban hordozott információk alapján lehetséges az eredeti arcképek rekonstruálása [40]. Az így kapott arcképek kevésbé részletgazdagok, mint az eredeti.

A dolgozatomban azt vizsgálom, hogy a gépi tanulás alapú arcfelismerés által kinyert arclenyomatok lehetővé teszik-e az arclenyomat forrását, az eredeti alanynak a felismerését. Megvizsgálom, hogy az arclenyomatok hogyan kódolják az arc struktúráját, illetve, hogy annak mely részei hordoznak érzékeny információt az adatalanyról. Célom arclenyomatok vizsgálatát olyan módszerrel végezni, amely általánosítható lehet akár generatív modelleknél használt belső reprezentációk analízisére.

A dolgozatom felépítése a következő. A 2. fejezetben a témámhoz kapcsolódó fontosabb témaköröket mutatom be, amit a 3. fejezetben a probléma bemutatása követ. A 4. fejezetben az arclenyomatokhoz kapcsolódó adatvédelmi kérdésekkel foglalkozom. Az 5. fejezetben azt vizsgálom, milyen információkat hordozhat egy arclenyomat, azok miként manipulálhatóak. A 6. fejezetben javaslatot teszek az adatvédelmi kockázatok kezelésére.

2. Irodalomkutatás

A gépi tanulási modellek csoportosíthatók a feladat típusa alapján: megkülönböztetünk felügyelt, felügyelet nélküli, illetve megerősítéses tanulást.

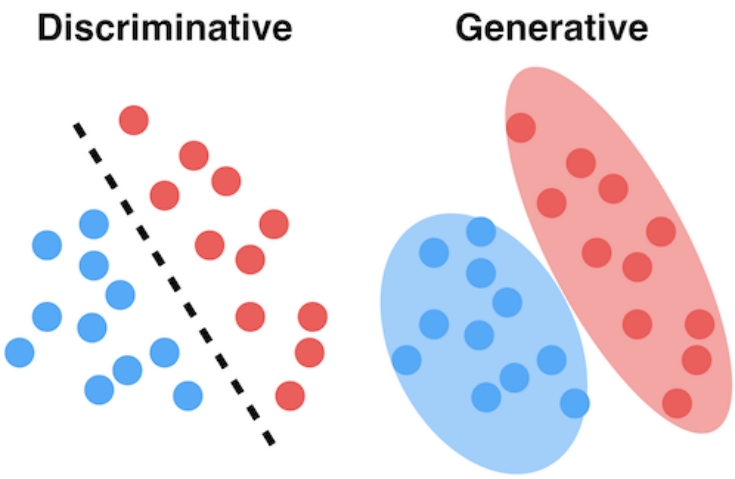
Felügyelt tanulásról akkor beszélünk, amikor a a gépi tanulási modellt címkézett mintakészleten tanítjuk be, azaz ismerjük az egyes tanítómintákhoz tartozó elvárt kimeneteket. A felügyelt tanulás célja egy olyan funkció elsajátítása, amely az adatok mintázata és a címkék alapján a legjobban közelíti a bemenet és kimenet közötti kapcsolatot. A betanítást követőn a modell képes általánosítani, azaz a modell által tanítás során nem látott, új mintákra is képes helyes becslést adni. A felügyelt tanulásnak két alkategóriája van, a regresszió és az osztályozás (vagy más néven klasszifikáció). Osztályozásról akkor beszélünk, ha a cél a bemeneti minták egy vagy több osztályba való besorolása. Ha a háló a bemenetet két különböző osztályba sorolja, akkor ezt bináris osztályozásnak nevezzük, míg a több mint két osztály közötti választást többosztályos osztályozásnak nevezzük. Ezzel szemben a regressziós hálókat folytonos értékek előrejelzésére használják. Ilyen folytonos adat például egy ingatlan árának becslése.

Felügyelet nélküli tanulás esetén a mintakészlethez nincsenek címkéink, nem ismert az egyes mintákhoz tartozó elvárt kimenet. Ekkor a modell célja a tanító adathalmaz belső mintázatainak, összefüggéseinek megtanulása, ami tipikus számos iteráció után alakul ki. Felügyelet nélküli tanulást alkalmaznak többek között klaszterezési feladatok megoldására, dimenziócsökkentésre, asszociációs feladatokra, anomália detektálásra, illetve generatív modellekhez is [16].

A gépi tanulási modelleket feladatuk szerint két további alkategóriába sorolhatjuk: generatív és diszkriminatív modellek (lásd: 1. ábra)

A diszkriminatív modellek a statisztikai osztályozásban használt modellek egy osztályába tartoznak, amelyeket főként felügyelt gépi tanulási feladatokra használnak. A betanított diszkriminatív modellek (nevükből adódóan) képesek megkülönböztetni az egyes osztályokhoz tartozó mintákat. Céljuk megtanulni az egyes osztályok közötti határokat. Például az 1. ábrán látható mintakészleten a diszkriminatív modell képes volt megtanulni a két osztály szeparáló egyenesét. Ezek a modellek azonban nem képesek új adatpontok előállítására.

A generatív modellek a felügyelet nélküli tanulásban használatosak. Ezek a modellek képesek megtanulni a tanító minták valószínűségi eloszlását, majd tanítást kö-



1. ábra. Diszkriminatív és generatív modellek közötti eltérés. (forrás: [8])

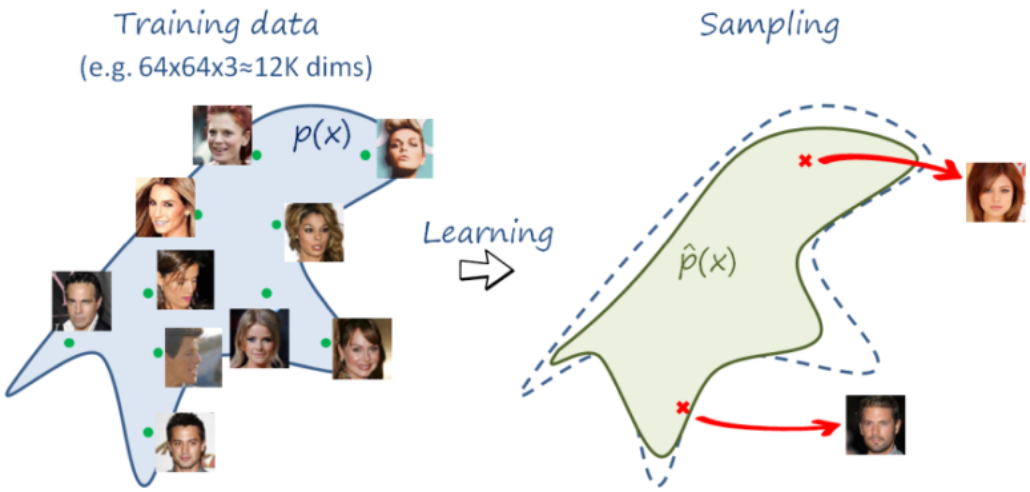
vetően új minták generálására képesek. Ez a tulajdonság a gyakorlatban rendkívül előnyös. A mély tanulásban használt modellek betanításához nagy mennyiségű, és megfelelő minőségű adatra van szükség, ezért az ezzel foglalkozó vállalatok rengeteg erőforrást áldoznak a tanítóminták begyűjtésére. Jellemzően manuális annotációra van szükség ami munkaigényes folyamat. Ez a megközelítés költséges, és nehezen skálázható, ezért a jó minőségű adathoz tipikusan csak a nagyobb vállalatok képesek hozzájutni. A generatív modellek használatával több vállalat képes minőségi tanítóadatot előállítani lényegesen kevesebb mintából.

A generatív modelleknek számos érdekes felhasználási területe van, ezek közül néhány példa:

- Fotorealisztikus emberi arcképek generálására képes [31].
- Egy képet leíró szöveg alapján képes képet készíteni [48].
- Az adathalmazban ritkán előforduló adatok generálását tudja, ami kiegyensúlyozhatja az adatot [51].
- Realisztikusan képes utánozni az emberi hangot [25].

A generatív modellek megértéséhez vegyünk egy példát. Tegyük fel, hogy van egy adathalmazunk, amely emberi arcokról készült fényképeket tartalmaz. Egy olyan gépi tanulási modellt szeretnénk betanítani az adathalmazunk alapján, amely képes egy nem létező emberekről valóságosnak tűnő arcképet generálni. Egy ilyen feladat

megoldható generatív modellezéssel. A 2. ábra egy tipikus generatív modellezési folyamatot mutat be.



2. ábra. A betanított generatív modell által megtanul eloszlás $\hat{p}(x)$ jól közelíti a valódi adateloszlást $p(x)$. (forrás: [21])

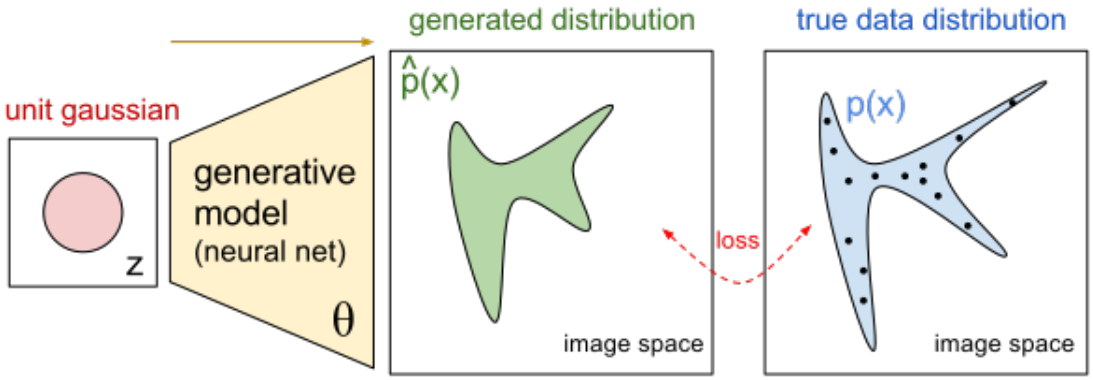
Először is szükségünk van egy adathalmazra, amely számos mintát tartalmaz az általunk generálni kívánt dologról. Az adathalmaz egyes elemeit megfigyeléseknek (observation) nevezzük. Minden megfigyelés sok jellemzőből (feature) áll amelyek képek esetén az egyes pixelek értékeit jelentik. Célunk egy olyan modell létrehozása, amely képes olyan új megfigyeléseket generálni, amelyek jellemzői jól közelítik az eredeti adathalmazban lévő jellemzőket. Képgenerálás esetén ez egy rendkívül nehéz feladat, mivel az egyes pixelek rengeteg lehetséges értéket vehetnek fel, és ezek közül csak relatíve kevés konfiguráció felel meg annak a képnek amit mi generálni szeretnénk.

A generatív modellek sztochasztikus jellegűnek kell lennie, nem pedig determinisztikusnak. Ha a modellünk csupán egy fixált számítást végezne, például az adathalmaz minden egyes pixelének átlagát számolná, akkor a modell minden alkalommal ugyanazt a kimenetet eredményezi. Így a generatív modellek szükségszerűen tartalmaznia kell egy véletlenszerű elemet, amely befolyásolja a modell által generált mintákat.

Más szavakkal, létezik egy ismeretlen valószínűségi eloszlás, amely leírja, hogy az egyes képek miért találhatóak a mintakészletünkben, míg más képek nem. A gyakorlatban nem ismerjük a tényleges valószínűségi eloszlást, ezért azt csak közelí-

teni tudjuk a megfigyelésekből. A generatív modellek célja az, hogy ezt az ismeretlen eloszlást minél jobban közelítse. A modell tanítása után a megtanult eloszlást mintavételezve olyan megfigyeléseket kaphatunk, amelyek az eredeti adathalmaz elemeihez nagyon hasonlóak.

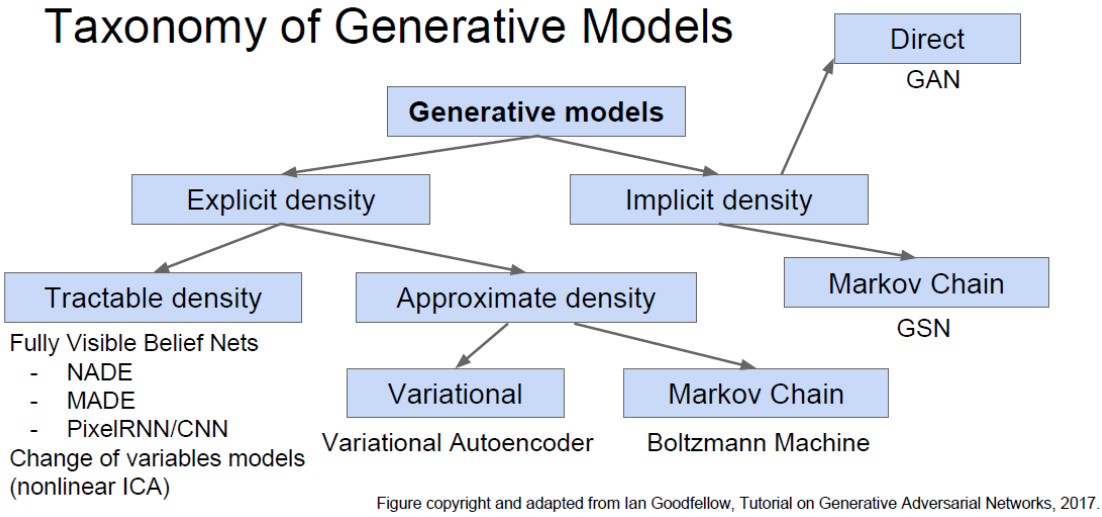
Matematikailag, az adathalmaz elemei $x_1, \ldots, x_n$ a valódi háttéreloszlásból $p(x)$ lettek mintavételezve. A 3. ábrán a kék háttérrel jelölt terület azon részét mutatja, amely nagy valószínűséggel (bizonyos küszöbérték felett) valódi képeket tartalmaz, és fekete pontok jelzik adatpontjainkat (mindegyik egy kép az adatkészletünkben). Modellünk egy becsült elosztást ír le $\hat{p}_\theta(x)$ (zölddel jelölt az ábrán) amelyet implicit módon úgy határozunk meg, hogy pontokat veszünk egy standard normális eloszlásból (pirossal jelölt), és feltérképezzük őket egy determinisztikus neurális hálón - ami a generatív modellünk (sárga). A Neurális hálózat θ paraméterekkel rendelkezik, amelyeket módosítva megváltozik a generált képek eloszlását. Célunk olyan θ paraméterek meghatározása, amelyek egy olyan eloszlást hoznak létre, amely jól közelíti a valódi adateloszlásunkat. A θ paraméterek meghatározása egy iteraív folyamat, véletlenszerű értékekkel inicializálva, majd a tanítás során a paraméterek megváltoztatásával egyre jobban közelíti a becsült eloszlás a valódi adateloszlást.



3. ábra. Általános generatív modell tanítása. (forrás: [43])

A generatív modellek általában kétféle sűrűségbecslést használnak. Explicit Sűrűségbecslés (EDE), és Implicit Sűrűségbecslés (IDE) [17]. Az EDE-ben előre meghatározott sűrűségfüggvényeket használnak a megfigyelések és valószínűségeik közötti kapcsolat közelítésére. Ezek a függvények paramétereik változtatásával illeszthetőek a megfigyelésekre. Például normális eloszlás két paramétere: várható érték és szórás állításával illeszthető az adatra. Az IDE-ben nem előre meghatározott sűrűségfüggvényeket használnak, hanem egy algoritmust használnak a valószínűségi eloszlás

közelítésére. Ennek egy példája kernel sűrűségbecslés. Bár az IDE módszerek is paramétereket használnak a közelítéshez, ezeket nem lehet közvetlenül manipulálni mint az EDE esetén. A 4. ábra a különböző generatív modellek rendszerezését mutatja be az alkalmazott sűrűségbecslés típusa szerint.



4. ábra. A generatív modellek fajtái (forrás: [17]).

2.1. Boltzman gépek

A következő részben a generatív modellek egyik fajtáját: a Boltzman gép (BM) működését, illetve annak egyik változatát, a Restricted Boltzman gépet (RBM) mutatom be, de előtte néhány alapvető fogalmat vezetek fel, amik szükségesek a Boltzman gépek megértéséhez.

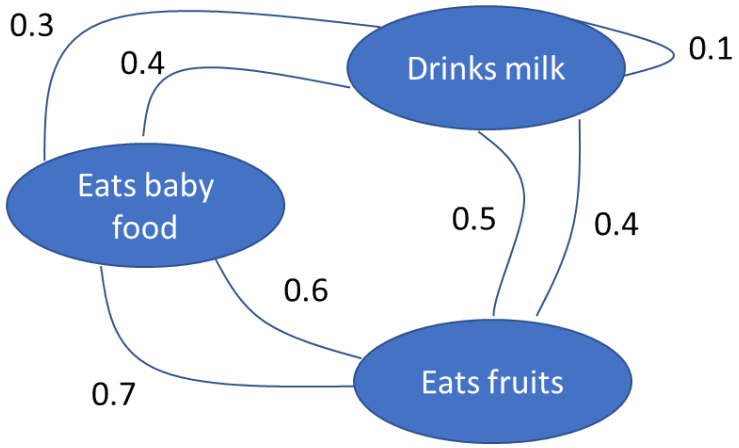
2.1.1. Markov-lánc

A matematikában a Markov-lánc egy olyan diszkrét sztochasztikus folyamatot jelent, amely Markov-tulajdonságú. Nevét egy orosz matematikusról, Andrej Markovról kapta, aki hírnevét a tudomány ezen ágában végzett kutatásaival szerezte. A Markov-tulajdonság röviden annyit jelent, hogy egy adott rendszer jövőbeli állapota csak a jelenlegi állapottól függ, a múltbeli állapotoktól nem. Másképpen megfogalmazva ez azt jelenti, hogy a jelen leírása teljesen magába foglalja az összes olyan információt, ami befolyásolhatja a folyamat jövőbeli helyzetét. Példaképpen: a vélet-

lenszerűen sétáló személy helyzete a $t+1$ pillanatban a t aktuális állapottól függ, és nem a korábbi állapotoktól $(t-1, t-2, \dots)$. Ezt a viselkedést Markov tulajdonságnak nevezik.

2.1.2. Grafikus modell

A grafikus valószínűségi modell egy grafikus ábrázolás, amivel véletlen változók közötti feltételes valószínűséget lehet kifejezni. A grafikus modellnek két fő komponense van: csúcsok és élek. A csúcsok a véletlen változó állapotát jelzik, míg az élek az átalakulás irányát. Az ilyen gráfoknak két fő típusa van: irányított és irányítatlan. A 5. ábrán látható példán egy csecsemő táplálkozási szokásainak Markov folyamatának irányítatlan grafikus modelljét láthatjuk. A baba következő étkezésének kiválasztása kizárólag attól függ, hogy mit eszik most, és nem attól, amit korábban evett.



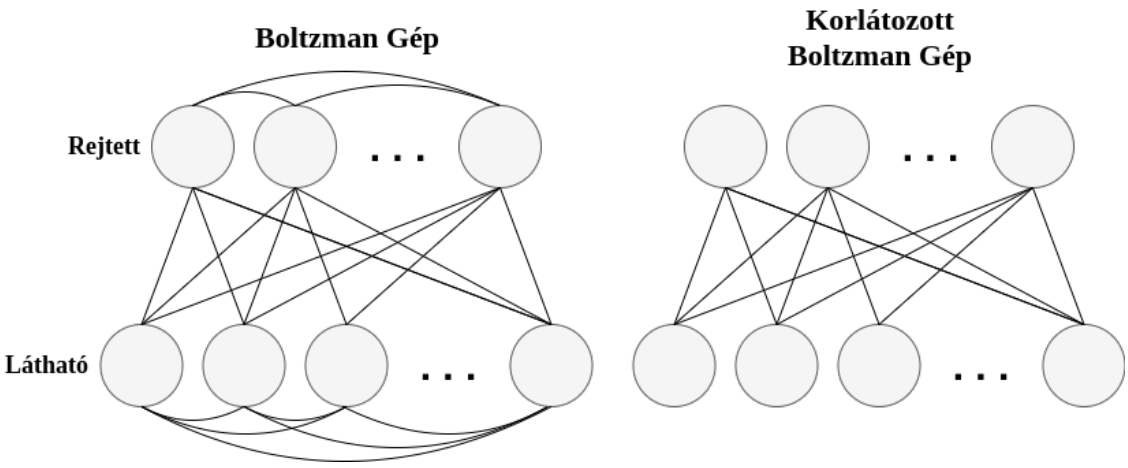
5. ábra. Példa grafikus ábrára.

Egy Markov tulajdonsággal rendelkező irányítatlan gráf által leírt valószínűségi változók halmazát Markov hálózatnak hívjuk (Markov Random Field az angol szakirodalomban). A Boltzman gépek is a MRF-ek közé tartoznak.

A Boltzman gép egy probabilisztikus generatív irányítatlan gráfmodell, amely kielégíti a Markov tulajdonságot. A BM-ek megtanulják a tanító minták eloszlását, majd abból új mintákat tudnak generálni. A BM rendelkezik bemeneti réteggel (látható réteggel) és egy vagy több rejtett réteggel. Kimeneti rétege nincs. A 6. ábrán láthatunk egy általános, több rejtett réteget tartalmazó Boltzman gépet.

A hálózat neuronjai megtanulnak sztochasztikus döntéseket hozni akkor, hogy be vagy kikapcsoljanak a tanítás során látott mintáknak megfelelően. Így a BM-ek

fel tudják fedni a tanító adatok összetett mögöttes mintázatait. Lényeges különbség a BM és az egyéb neurális hálózat architektúrák között az, hogy a BM neuronjai nemcsak más rétegek neuronjaihoz kapcsolódnak, hanem ugyanazon a rétegen belüli neuronokhoz is, minden neuron kapcsolódik a hálózat összes többi neuronhoz. Ezt azt architektúrát Korlátlan Boltzman Gépnek nevezik, aminek a tanítása rendkívül nehézkes és e miatt kevés gyakorlati haszna van. Az azonos rétegbeli neuronok közötti kapcsolatok megszüntetésével egy más architektúrát kapunk, amit Korlátozott Boltzman Gépnek (RMB) hív a szakirodalom. A gyakorlatban az RBM-et használják, mert ezt a struktúra könnyebben tanítható. A BM és RBM közötti különbséget a 6. ábra szemlélteti.



6. ábra. A Boltzman gép egy kétirányú gráfon alapuló irányítatlan grafikus modell, aminek egyik részén a látható elemek vannak, a másik részén a rejtett elemek. (forrás: [41] alapján saját)

2.1.3. RBM alkalmazásai

A mély tanulás első napjaiban az RBM-eknek különféle feladatoknál alkalmazták, mint például dimenziócsökkentéshez, vagy ajánlórendszerekhez vagy természetes nyelv feldolgozásra.

Néhány alkalmazási példa RBM-re:

- Mintázat felismerés: Az RBM jellemzők kinyerésére (feature extraction) alkalmas mintázatfelismerési problémák esetén, például részvényárfolyam becslésnél. [5].

- Ajánlórendszerek: Az RBM-eket széles közben használják olyan feladatokra, ahol megjósolják, hogy mit érdemes ajánlani a felhasználónak ami számára hasznos, érdekes lenne. Példa: filmajánlás, zeneajánló [37].

Azonban az utóbbi időkben a gyakorlatban az RBM-eket szinte teljesen felváltották a Generatív Adversarial Network-ök (GAN), vagy a Variational Autoencoderek (VAE).

2.2. Generatív Adversarial Network

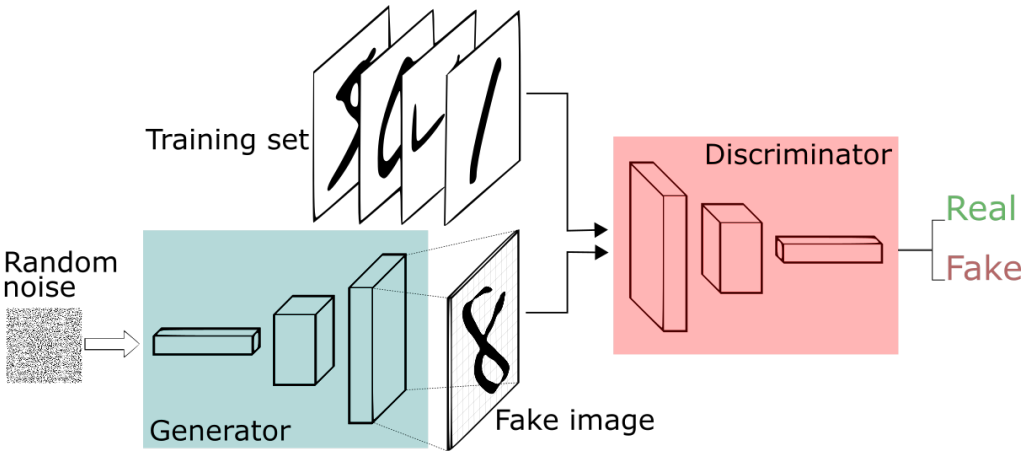
A Generatív Adversarial Network (más néven GAN) az utóbbi évek egyik innovációja a mély tanulás területén. A GAN-okat eredetileg Ian Goodfellow és Yoshua Gengio vezették be a Montreali Egyetemen, 2014-ben [18]. A GAN-ok célja egy eloszlás megtanulása, és abból új példányok generálása.

A GAN egy generatív modell, amelyben két neurális háló verseng egymással. Az első hálót generátornak nevezik. A generátor felelős új szintetikus minták előállításáért, amelyek hasonlítanak a tanító halmaz mintáihoz. A másik neurális hálót diszkriminátornak nevezik, aminek a feladata megkülönböztetni a valódi minták (tanító minta) illetve a generátor által létrehozott szintetikus minták között. A generátor célja becsapnia a diszkriminátort, míg a diszkriminátor megpróbál ennek ellenállni. A két háló versengéséből ered az „adversarial" elnevezés.

Az alábbi 7. ábrán látható, hogy a generátor nem találkozik a tanító mintákkal, csupán véletlen zajt kap a bemenetére. A diszkriminátor hozzáfér a tanító adatokhoz osztályozás céljából. A generátor képes javítani a kimenetén kizárólag a diszkriminátortól kapott visszajelzés alapján (pozitív, ha a tanító mintával egyezik a kimenet, negatív ha nincs egyezés).

2.2.1. GAN tanítása

A generátor be kell tanítani, mielőtt realisztikus mintákat képes létrehozni. Tegyük fel, hogy egy újonnan inicializált hálót használunk 200 kép előállítására, minden mintánál új véletlenszerű zaj bemenettel. Mivel a generatív modell tanítása felügyelet nélküli, nincsenek címkéink amivel össze lehetne vetni a háló kimenetét. Honnan tudjuk mégis, hogy milyen irányba kell módosítani a háló paramétereit, hogy javuljon a generált képek minősége? Egy generált kép „jóságát" nem külsőleg mondjuk

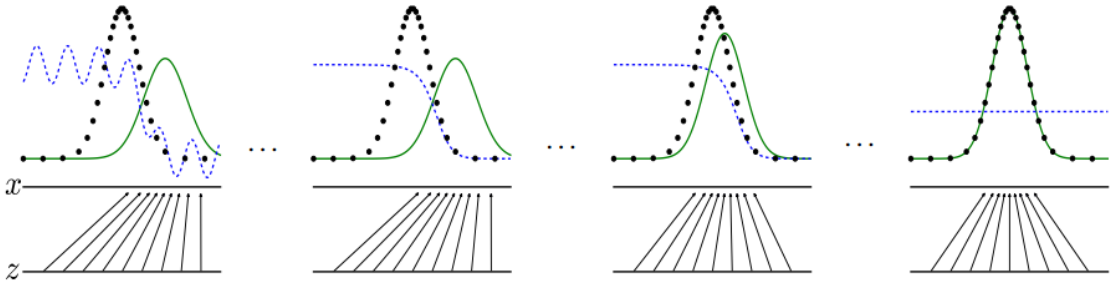


7. ábra. A GAN általános struktúrája. (forrás: [59])

meg (címke segítségével), hanem az dönti el, hogy a generátornak sikerült-e megtéveszteni a diszkriminátort.

A diszkriminátor háló ad visszajelzést a generátornak tanulás során, hogy meg tudja tanulni az adott minták eloszlását. A diszkriminátor tipikusan egy osztályozó konvolúciós neurális háló ami képes megkülönböztetni a generált mintákat a valós mintáktól. A tanítás során a diszkriminátornak az esetek felében valódi mintákat, a másik felében szintetikus mintákat lát. A valós mintákhoz 1-hez közeli valószínűséget, hamis mintákhoz pedig 0-hoz közelit rendel. A generátor e közben megtanul olyan mintákat generálni, amelyekre a diszkriminátor 1-hez közeli kimenetet produkál, és valódi mintának minősítené azt. Idővel a generátor kénytelen egyre realisztikusabb kimeneteket előállítani, hogy meg tudja téveszteni a diszkriminátort. Ezzel a módszerrel egy felügyelet nélküli tanulási feladatot sikerül visszavezetni egy felügyelt tanulási feladatra.

A 8. ábrán láthatjuk a GAN tanulásának folyamatát. A generátor z véletlen zaj értéket kap bemenetre, és azt leképezi az x kimeneti értékre. Az x értékeinek eloszlása sűrűbbé válik, ahova több z értéket képezünk le. A diszkriminátor magas értéket ad azokon a helyeken, ahol a valódi adatok sűrűsége nagyobb, mint a generált adatoké. A generátor a háló paramétereinek változtatásával tud módosítani a leképezésen, és így eléri, hogy az eloszlása a diszkriminátor gradiens irányába változzon. Végül a generátor eloszlása megegyezik a valós minták eloszlásával. Ekkor a diszkriminátor minden mintára 0,5-ös valószínűséget ad, mivel nem tudja megkülönböztetni a valós és generált mintákat.



8. ábra. Tanulás folyamata. Kék pontozott vonal jelöli a diszkriminátor eloszlását, zöld folytonos vonal a generált minták eloszlását, a fekete görbe pedig a tanító adat eloszlását [18].

A GAN-oknak annak ellenére, hogy nemrég jelentek csak meg, gyakorlati hasznuk is van. Erre pár példa:

- **StyleGan3**: Ez egy GAN-on alapuló neurális háló, amely képes az fotorealisztikus emberi arcképeket generálni. [31]
- **TecoGAN**: Videó felvételek felbontásának növelésére képes GAN használatával. Az egyes képkockákat részletgazdagabbá teszi [9].
- Egy 2018-as aukción egy GAN által „festett” kép több mint $400 ezer dollárért kelt el [10].

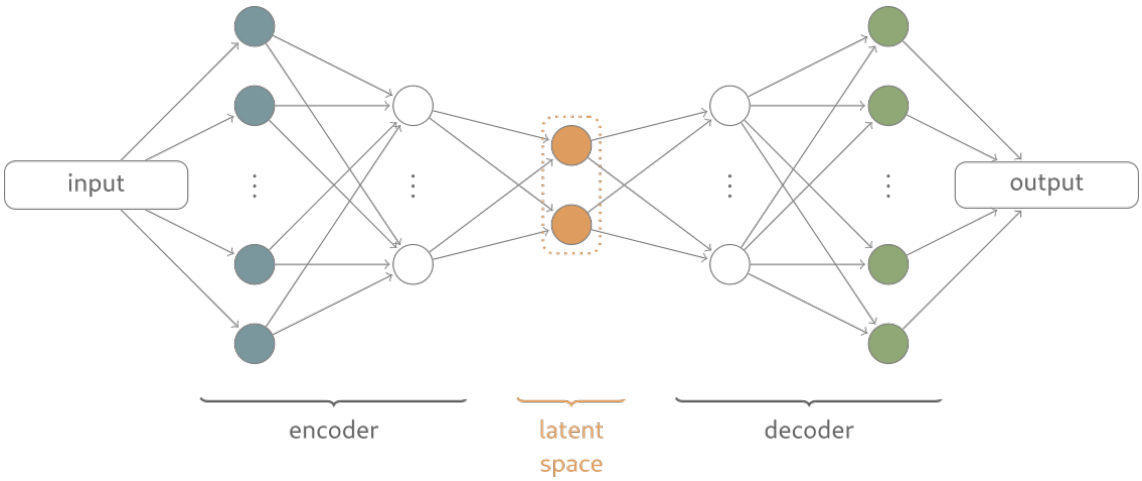
2.3. Generatív Autoenkóderek

Az autoenkóderek olyan neurális hálók, ahol a kimeneti réteg mérete megegyezik a bemeneti réteg méretével. Az autoenkóder felügyelet nélküli tanulásra képes, feladata minél kisebb veszteséggel visszaállítani a bemenetre érkező adatot, ezért replikátor neurális hálónak is nevezik.

A háló tartalmaz egy $\mathbf{h}$ rejtett réteget, ami a bemenet belső leképezését tárolja. A háló két részből tevődik össze: egy kódoló függvényből $\mathbf{h} = f(\mathbf{x})$, és egy dekódolóból, amely rekonstruálja a bemenetet $\mathbf{r} = g(\mathbf{h})$. Ha az autoenkóder egyszerűen megtanulná a $g(f(\mathbf{x})) = \mathbf{x}$ függvényt, az nem lenne túl hasznos, ezért a kódolót úgy tervezték, hogy képtelen legyen a bemenetet tökéletesen másolni. A kódolót úgy korlátozzák, hogy a középső rejtett réteg kevesebb neuront tartalmaz mint a be-

meneti réteg. Ennek hatására a kódoló kénytelen priorizálni, hogy a bemenet mely jellemzőit érdemes másolni, így képes megtanulni az adatok hasznos tulajdonságait.

Az autoenkóderek tipikus felépítése: (lásd: 9. ábra).



9. ábra. Az autoenkóder részei: enkóder, kód (látens tér) és dekódoló (forrás: [56])

- **Kódoló**: A kódoló egy előrecsatolt, teljesen kapcsolt neurális háló, amely a bemenetet látens térbeli reprezentációba tömöríti. A bemeneti képet tömörített ábrázolásként kódolja csökkentett méretben.
- **Kód**: A hálózat ezen része tartalmazza a dekódolóba táplált bemenet csökkentett ábrázolását (más néven látens változók).
- **Dekódoló**: A dekódoló a kódolóhoz hasonló struktúrájú előrecsatolt háló. Feladata kód alapján a bemenet visszaállítása.

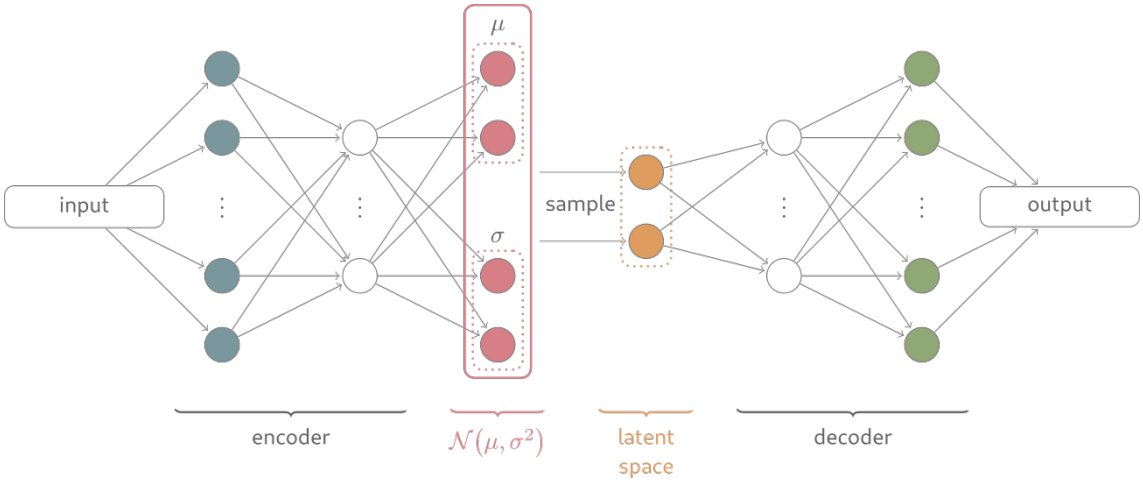
Az autoenkóderek évtizedek óta ismertek a neurális hálózatok világában [22]. Míg eleinte az autoenkódereket dimenziócsökkentéshez, vagy jellemző tanuláshoz használták, a közelmúltban a generatív modellezésben is előtérbe került. Az első komoly generatív tulajdonsággal rendelkező autoenkódert a variációs autoenkóder megjelenése hozta.

2.3.1. Variációs autoenkóderek

A variációs autoenkóder (VAE) megjelenése nem túl régi, 2013-ban publikálták először [35]. Bár mély neurális háló alapú autoenkóderek korábban is léteztek, mint

generatív modell rosszul teljesítettek. Később azonban több alkalmazási területen is előremutató eredmények születtek, például a képgenerálás területén [47], vagy felbontásnövelő alkalmazásokban [36].

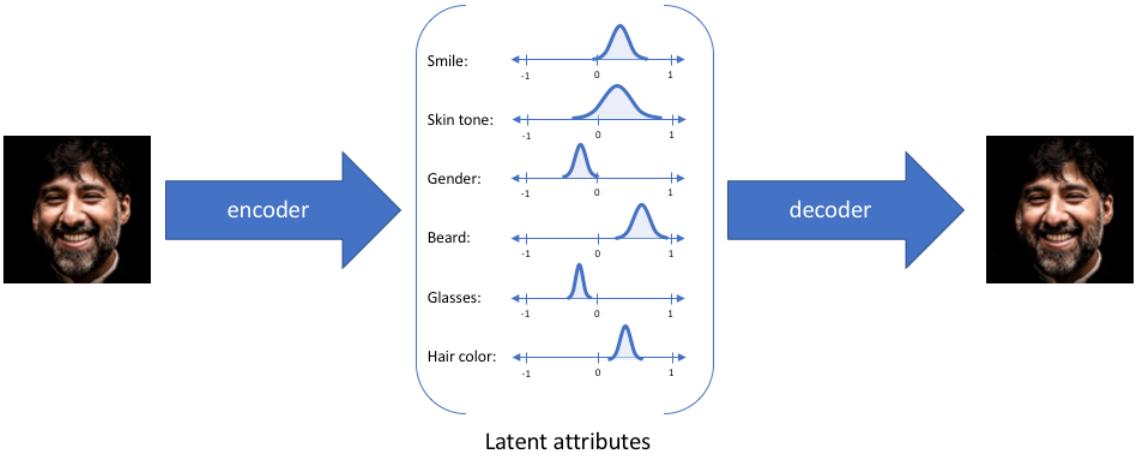
A variációs autoenkóder annyiban tér el a standard autoenkódertől, hogy adunk egy megkötést a kódolónak, aminek hatására a generált látens változók vektora ($\mathbf{h}$) közel követik a normális eloszlást. Ebben az esetben a normális eloszlás $\mathcal{N}(\mu, \sigma^2)$ várható értéke (μ) és szórásnégyzete (σ^2) leírja a látens változók eloszlását. Ekkor az enkóder és dekóder nem determenisztikus hanem sztochasztikus jellegű. Ez a megkötés azért hasznos számunkra, mert normális eloszlásból könnyen tudunk mintavételezni, ami megkönnyíti az új képek generálását (lásd: 10. ábra)



10. ábra. A variációs autoenkóder részei. (forrás: [56])

A VAE működését könnyebb egy példán keresztül bemutatni. Tegyük fel, hogy van egy képünk egy emberi arcról. A betanított háló ez a kép alapján képes látens változó vektort kinyerni a képből, ami az arcot leíró fontos jellemzőkből áll. Erre mutat egy példát a 11.ábra.

Mint az alábbi ábrán látjuk, minden jellemzőhöz tartozik egy valószínűségi eloszlás. Miben különböznek az arcok? Megfogalmazhatunk pár ilyen jellemzőt, mint a bőrszín, haj színe, szemek távolsága stb. Ezen jellemzőkből álló vektor leírhat egy emberi arcot. Eltérő arcoknál a jellemzők értékei mások lesznek, de jellemzők listája ugyanaz marad. Az autoenkóderek nem feltétlenül ilyen, ember által könnyen értelmezhető jellemzőket nyernek ki az arcokból, hanem azok ennél a példánál absztraktabbak.



11. ábra. Látens változók kinyerése a bemeneti arcképből. (forrás: [28])

Célunk, hogy új képeket állítsunk elő az arcképekből álló tanító mintahalmaz alapján. Feltéve, hogy a bemeneti adataink normális eloszlást követnek, új képek létrehozásához mindössze annyit kell tennünk, hogy normális eloszlásból mintavételezünk egy látens változó vektort. Ezt megadjuk a dekódolónak, ami a vektor alapján generálni tud egy új képet.

A VEA számos területen alkalmazott, ezek közül egy példa az úgynevezett „deepfake” előállítása, ami lényegében abból áll, hogy egy személy arcát alakítják át úgy, hogy egy másik személy vonásait utánozza [60].

2.4. Arcfelismerés neurális hálókkal

A generatív modellek után áttérnék a modern arcfelismerő rendszerek bemutatására. Mivel a legkorszerűbb arcfelismerő rendszerek mély tanulásra építenek, az ehhez kapcsolódó fogalmakat, neurális háló architektúrákat mutatom be a következő részekben.

2.4.1. Mély metrika tanulás

A mély metrika tanulás (angol szakirodalomban: deep metric learning) a felügyelet gépi tanulás azon területe, amelyben a cél egy hasonlósági függvény megtanulása [32]. Ez a függvény képes meghatározni két objektumról, hogy mennyire hasonlítanak egymáshoz és hasonlóság mértékét számszerűsíteni tudja. Magasabb hasonlósági

értéket ad vissza a függvény, ha az objektumok hasonlóak, és alacsonyabb értéket ad vissza, ha az objektumok eltérőek. Ez a metrika majd használható különböző feladatokra, mint osztályozásra, vagy klaszterezésre. A mély metrika tanulásnak sok felhasználási esete van, ezek közül mutatok be pár példát.

Tekintsünk egy olyan példát, amelyben egy iskolában elhelyezett kamerák felvételei alapján, egy gépi tanulási modell segítségével szeretnénk felismerni az iskola tanulóit. Szeretnénk tudni, hogy mely tanulók vettek részt a tanórákon. Egyik lehetséges megoldás az lenne, ha kép klasszifikációt alkalmazunk, ahol a klasszifikáció minden osztálya egy-egy tanulónak felel meg. Ehhez szükséges minden tanulóról begyűjteni valamennyi képet, majd a képek alapján be tudunk tanítani egy klasszifikációs modellt. A modell betanítása után már minden tanulót felismerhetünk az osztályteremben. Ez az elképzelés működhet, viszont probléma adódik akkor, ha új tanuló csatlakozik az osztályhoz. Ekkor a modellünk nem fogja felismerni az új tanulót addig, amíg új tanító minták alapján azt újra nem tanítjuk. A modell újratanítása költséges idő és számításigény szempontjából, ezért tipikus képklasszifikációs modell helyett a feladat érdemes mély metrika tanulással megoldani.

Ha mély metrika tanulást használunk, akkor a modell kimenete a hasonlóság mértéke lesz. Ebben az esetben összehasonlíthatjuk a kamera képet a tanulókról készült képekkel, és ha a hasonlósági függvény által megadott érték nagyobb egy küszöbértéknél, akkor sikerült azonosítani a tanulót. Ennek a módszernek az az előnye, hogy új tanuló csatlakozása esetén sem szükséges újratanítani a modellt. Csupán az új képekre van szükségünk amelyeket fel tudunk használni az összehasonlításhoz.

A mély metrika tanulásának másik példája lehet a csekkeken lévő aláírások összehasonlítása [55]. Ekkor a hasonlósági függvény összehasonlítja a csekken lévő aláírást a számlatulajdonos aláírásával. Ha a hasonlóság értéke megfelelően nagy, akkor a csekket elfogadják, ha az érték alacsony akkor az aláírás valószínűleg hamisított.

Mély metrika tanulást alkalmazzák természetes nyelvfeldolgozás (NLP) területén is. Az egyik gyakori felhasználása, a gyakran ismétlődő kérdések felismerése olyan fórumokon, mint a Quora vagy a StackOverflow, amelyekre óránként több ezer kérdés érkezik be. Erre a problémára a Kaggle weboldalon rendeztek is egy versenyt, amelyen az első helyezettet 12500 amerikai dollárral jutalmazták [29]. Ez látszólag nem tűnik olyan nehéz feladatnak, hiszen mindössze annyit kell tennünk, hogy összehasonlítjuk a szavakat a kérdésben. Viszont két azonos jelentésű kérdés sokszor

eltérő szavakat használ, mint például a „Milyen idős vagy?” és „Hány éves vagy?” kérdések. Ezért a szavak közvetlen összehasonlításával nem tudnánk megmondani, hogy a két kérdés tartalma hasonló-e. Ez a feladat is megoldható egy olyan hálóval, amely magas értéket ad akkor ha a kérdések hasonló jelentésűek, és alacsony értéket akkor, ha a kérdések különbözőek.

2.4.2. Sziámi háló struktúra

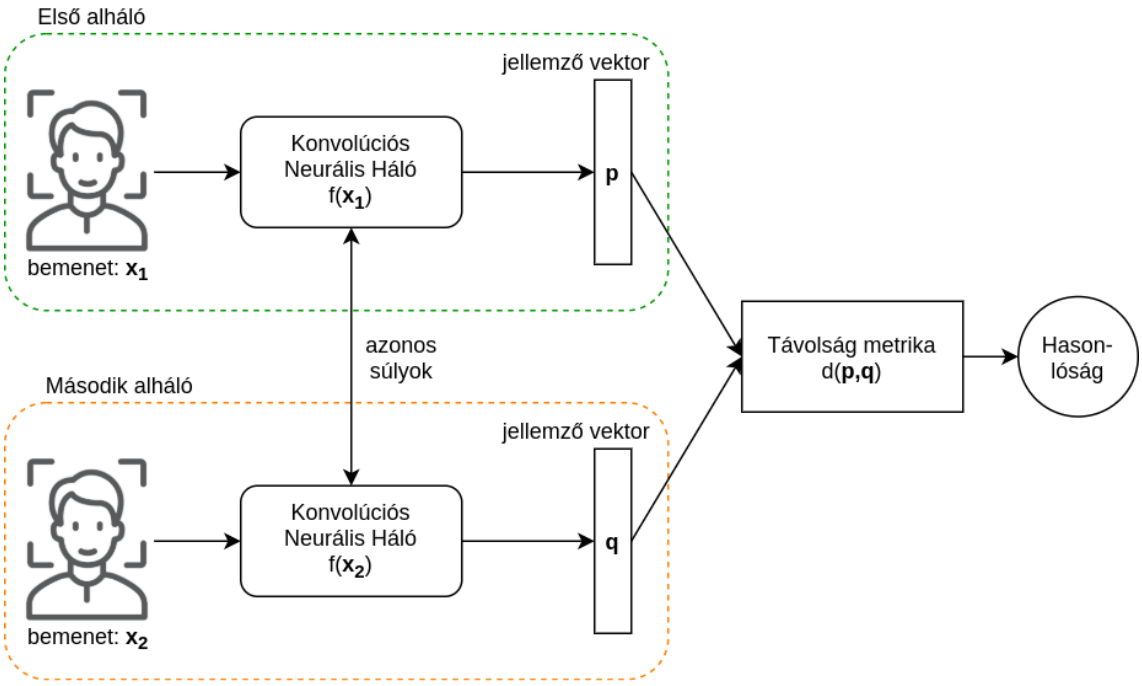
Az előző részben kifejtett mély metrikák tanulásához tipikusan sziámi hálókat alkalmaznak. A sziámi háló (siamese network) egy mély neurális háló architektúra, amely két vagy több azonos alhálózatot tartalmaz. Az alhálózatok teljes mértékben megegyeznek egymással, azaz azonos a paramétereik értéke. Ha tanítás során az egyik alháló súlyát megváltoztatjuk, akkor azzal egyidejűleg a többi alhálót is ugyanúgy módosítjuk, hogy azok mindig azonos súlyokkal rendelkezzenek. Ezeket a hálókat a bemenetek hasonlóságának megállapítására használják a jellemzővektoraik összehasonlításával [4].

A sziámi hálóknak elegendő kevesebb tanító minta, mint a tipikus mély neurális hálóknak. Ez a tulajdonsága kedvező olyan feladatoknál, ahol a tanító minták száma korlátozott, vagy nehezen begyűjthető nagy mennyiségben. A sziámi hálókat előszeretettel alkalmazzák arcfelismeréshez, illetve aláírás verifikáláshoz.

A sziámi hálók előnyei:

- Ellenállóbb a tanító adat kiegyensúlyozatlanságára: A One-shot learning segítségével, osztályonként egy, vagy néhány kép is elegendő ahhoz, hogy a sziámi hálózatok megfelelően rátanuljanak az adatokra, majd felismerjék ezeket a képeket a jövőben.
- A sziámi hálók embeddingeket tanulnak meg, a hasonló osztályokat/fogalmakat a látens térben közelebb helyezik egymáshoz, így szemantikai hasonlóságot képesek megtanulni.

A 12. ábrán láthatjuk a sziámi háló működését. A sziámi hálónak két bemenete van, az egyes bemenetekhez külön külön alháló tartozik. Az első alháló megkapja a bemenő képet, és miután az áthaladt a konvolúciós rétegeken létrehoz egy jellemzővektort (angolul feature vector vagy encoding), ami csökkentett dimenziójú a bemeneti képhez képest. A második alháló ami ugyanolyan súlyokkal dolgozik és egy



12. ábra. A sziámi háló működése két alháló esetén.

különböző képet kap a bemenetén. Ezt követően a két jellemzővektorunk van, rendre $\mathbf{p} = f(\mathbf{x_1})$ és $\mathbf{q} = f(\mathbf{x_2})$. A két jellemzővektort ezután valamilyen távolság metrika szerint összevethető, ami alapján meghatározzuk a hasonlóságot. A sziámi háló célja egy olyan metrika megtanulása, amely közel azonosnak tekinthető bemeneti képekre nagy hasonlóságot ad, míg az eltérő képekre kis hasonlóságot kapunk.

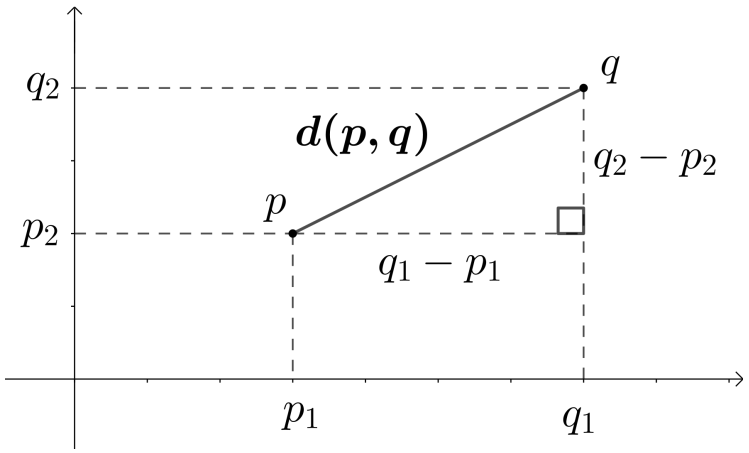
A jellemzővektorok összehasonlításához többféle távolság metrikát is használhatunk. Ezek közül leggyakrabban a használt metrika a két vektor euklideszi távolsága (ami megegyezik jellemzővektorok különbségének $L2$ normájával), ami egyszerűen kiszámolható az alábbi módon n dimenziós esetben [57]:

$$d(\mathbf{p}, \mathbf{q}) = \|\mathbf{p} - \mathbf{q}\|_2 = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2}$$

Ahol $\mathbf{p} = (p_1, p_2, \ldots, p_n)$ és $\mathbf{q} = (q_1, q_2, \ldots, q_n)$

2.4.3. Sziámi hálók tanítása

Mivel a sziámi hálózatok tanítása általában páros tanulással (angolul pairwise learning) megy végbe, a keresztentropia veszteség ebben az esetben nem használható.



13. ábra. Két pont euklideszi távolsága 2D esetben (forrás: [61])

Főként két veszteségfüggvényt használnak ezeknek a sziámi hálózatoknak a betanítására: a hármas veszteségfüggvényt (triplet loss) és a konsztratív veszteségfüggvényt (constrative loss).

A hármas veszteség lényege, tanítás során három sziámi háló bemenetére három különböző képet adunk. A három képből az elsőt referenciának nevezik (anchor), a másodikat pozitív példának, amely a referenciához nagyon hasonló kép, a harmadikat pedig negatív példának, amely a referenciától jelentősen eltérő kép. A tanítás során a neurális háló paramétereit úgy módosítjuk, hogy a referencia bemenetből és a pozitív bemenetből kinyert jellemzővektorok távolsága minimalizálva legyen, míg a referencia bemenetből és a negatív bemenetből kinyert vektorok távolsága maximalizálva legyen [23].

$$\mathcal{L}(A, P, N) = max\left(\|f(A) - f(P))\|_2 - \|f(A) - f(N)\|_2 + \alpha, 0\right)$$

A fenti egyenletben szereplő $f(A), f(P), f(N)$ a jellemzővektorokat jelöli (sorra referencia, pozitív és negatív bemenetekre), míg az α paraméterrel a hasonló és eltérő párok elvárt távolsága növelhető.

Kellő mennyiségű tanító kép és megfelelően választott kép hármasok használata esetén elérhető, hogy a neurális háló megtanuljon általánosítani, vagyis olyan képekből is megfelelő jellemzővektorokat hozzon létre, amelyeket nem használtunk a tanítás során.

A sziámi hálók tanítása során gyakran alkalmazzák még a konsztratív veszteségfüggvényt is. Ez egy távolság alapú veszteségfüggvény, nem a becsült kimenet és az elvárt kimenet különbségéből származtatott. A konsztratív veszteségfüggvényt főleg embeddingek tanulására használják, azzal a céllal, hogy két hasonló adatpontból származtatott embedding távolsága kicsi legyen, eltérő adatokból származtatott embeddingek távolsága pedig nagy legyen.

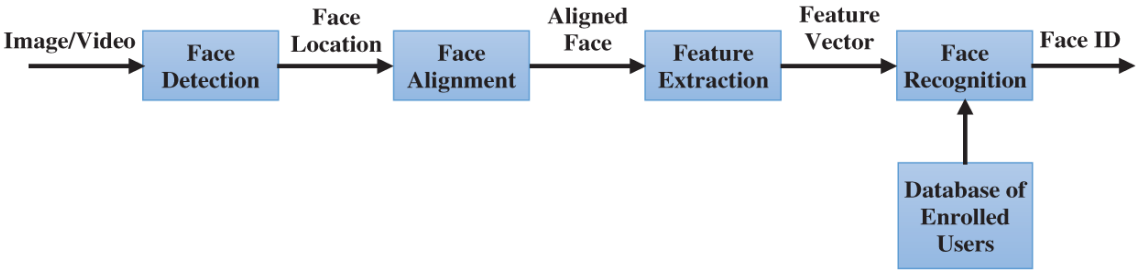
$$\mathcal{L}(A, B) = (1 - y) \ \|f(A) - f(B)\|_2^2 + (y) \ max\left(0, m - \|f(A) - f(B)\|_2\right)^2$$

Abban az esetben, ha A és B ugyanaz a kép, akkor a kimenet 0 lesz, míg mikor A és B eltér, akkor a kimenet 1 lesz.

3. A probléma bemutatása

Az utóbbi időkben széles körben elterjedt az arcfelismerő rendszerek alkalmazása, mivel az arcfelismerés egy hatékony eszköze a biometrikus azonosításnak. A modern, gépi tanulás alapú arcfelismerő rendszerek működése több lépésből áll, ezt mutatja be a 14. ábra.

Ahhoz, hogy az arcfelismerő rendszer egy személyt sikeresen azonosítani tudjon kamerafelvétel alapján, először a rendszernek sikeresen meg kell találja az emberi arcokat a képen belül. Az arcok észlelését egy képen arcdetektálásnak nevezzük. Miután sikerült lokalizálni az arcokat, azokat szükséges lehet igazítani képtranszformációk segítségével, hogy úgy tűnjön, mintha az arcokat szemből látnánk. A harmadik lépés, az arc képekből jellemzővektorok (későbbiekben: arclenyomatok) kinyerése, ami leírja az arc főbb vonásait, jellemzőit. Az arclenyomat vektorok összehasonlításával már képes a rendszer összevetni a kameraképen látható személy arcát az adatbázisban tárolt, ismert személyekkel. Ha van találat az adatbázisban, akkor sikeresen be lett azonosítva a személy. [20]



14. ábra. Arcfelismerő rendszer folyamatábrája. [20]

Az arcfelismerő rendszerek működésének kifejtése:

1. **Arc detektálása:** Az arcdetektálás az első lépés a sikeres azonosítás érdekében. Itt a cél meghatározni az arcot befoglaló téglalap koordinátáit a képen belül. Ehhez többféle képfeldolgozó módszer is alkalmazott, az egyik a *Haar cascade* alapú módszer, aminek fő előnye a gyors működés, de ennél pontosabb eredmény lehet elérni a *Histogram of Oriented Gradients* (HOG) módszer alkalmazásával. Gyakorlatban, a modern képfeldolgozó szoftvercsomagok mint az OpenCV [2] tartalmaz arcdetektáló megoldásokat.

2. **Arc igazítása:** Miután az arcdetektor segítségével sikerült meghatározni az arcot tartalmazó befoglaló téglalapot, valószínű, hogy az illető nem pont a ka-

merába néz, valamennyire el van fordulva. Az arc további feldolgozása előtt ezt szükséges képtranszformációs módszerekkel korrigálni. Ehhez előbb mély neurális háló segítségével felismernek az arcon valamennyi kulcspontot (angolul: facial landmark [33]). A kulcspontok relatív helyzetéből lehet következtetni, hogy az arcképet milyen irányba, milyen mértékkel szükséges igazítani.

3. **Arclenyomat kinyerése:** A következő lépés az arcból egy arclenyomat kinyerése. Ehhez szintén egy mély neurális hálót alkalmaznak, amely a eredeti képből képes jelentősebb kisebb méretű jellemzővektort előállítani, ami tipikusan 128, vagy 512 darab lebegőpontos értékből áll. Azt, hogy az arclenyomatok pontosan az arc mely jellemzőit tartalmazza, azt a 2.4.1. fejezetben bemutatott mély metrika tanulás alkalmazásával képes megtanulni a neurális háló. Az arclenyomat előnye, hogy jelentősen kisebb méretű az eredeti képnél, mégis alkalmas megkülönböztetés szempontjából jellemezni az emberi arc struktúráját.

4. **Arc azonosítása:** Az utolsó lépés az arcról készült arclenyomat összevetése egy adatbázisban tárolt, ismert forrású arclenyomatokkal. Az összevetéshez jellemzően valamilyen távolságmetrikát alkalmaznak, mint a 13. ábrán bemutatott az euklideszi távolságot. Ha megfelelően közeli a két vektor közötti távolság, akkor a képen látható személyt sikerült azonosítania a rendszernek.

Bár az arcfelismerő rendszerek nagyon hasznosak, és használatuk gyorsan terjed, jelentős adatvédelmi kockázatokat is hordoznak magukkal. [7]

Más biometrikus adatokkal ellentétben (mint az ujjlenyomatok, genetikai minták), az arcfelismerő rendszerek egy személy tudata és beleegyezése nélkül képesek távolról, érintkezés mentesen felvételeket készíteni a személyről. Az európai általános adatvédelmi rendelet (GDPR) értelmében ez komolyan fenyegeti az emberek személyes adataik védelméhez való jogát. Az arcfelismerő rendszerek alacsony költséggel kiépíthetőek, így tömeges megfigyelést tesznek lehetővé.

E mellett kockázatot jelent az is, hogy az arcképek már nagy mennyiségben elérhetőek az interneten. Az állami, vagy magán cégek korábbról, más célból gyűjtött kép adathalmazokkal rendelkeznek, amiket fel tudnak hasznosítani. Másfelől a közösségi médiák világszintű elterjedésével nagy mennyiségű publikusan elérhető arcképet lehet begyűjteni. Mivel az emberi arc vonásait, jellemzőit utólag nem lehet könnyen

megváltoztatni, ezért az egyszer begyűjtött arcképekről készített arclenyomatok a jövőben is alkalmasak lehet az adatalany azonosítására.

Annak ellenére, hogy az utóbbi években a nagy méretű adathalmazok kihasználásának és a mély tanulás fejlődésének köszönhetően az arcfelismerő rendszerek teljesítménye sokat javult, a megbízhatóságuk még továbbra is meglehetősen korlátozott. A kép készítésének módjától függően magas lehet a téves pozitív (tévesen felismert) esetek száma. E mellett, néhány arcfelismerő rendszer jobban működik fehér embereken mint sötétebb bőrszíneken, eltérően működik férfiak és nők esetén, gyerekek és felnőttek esetén. Az ilyen elfogultságok problémákhoz vezethetnek főleg automatikus döntéshozó rendszereknél.

Mint azt láthattuk a 14. ábrán, az arcfelismerő rendszerek működéséhez szükséges egy arclenyomat adatbázis, amivel össze tudja hasonlítani az azonosítatlan személy arclenyomatát. Az arclenyomatok tárolása viszont adatvédelmi kockázatokkal járhat. Számos arcfelismerési kockázat kapcsolódik ezen adatbázisok kezeléséhez, integritásához vagy titkosságához. Ha az arclenyomatok titkosítás nélkül egy nagy, centralizált adatbázisban vannak tárolva, előfordulhat, hogy ahhoz egy rosszindulatú fél hozzáférést szerez.

Kimutatták, hogy az arclenyomatok alapján lehetséges az eredeti arcok részleges visszaállítása [40], ebből arra következtethetünk, hogy az arclenyomatok magukban érzékeny információkat hordozhatnak az adatalanyról. Ha ezeket az információkat egy rosszindulatú fél képes megbízható módon kinyerni az arclenyomatokból, akkor az is jelentős adatvédelmi kockázat lenne.

A diplomamunkám során az arcfelismerő rendszerek által használt arclenyomatok vizsgálatával foglalkozom. Egyik feladatom azt elemezni hogy, hogyan van kódolva a személyes adat az arclenyomatokban, illetve bemutatni az arclenyomatokhoz kapcsolódó adatvédelmi kockázatokat. Ezekkel a kérdésekkel a 4. és 5. fejezetben foglalkozom. Végül a 6. fejezetben védekezési javaslatokat teszek, milyen módszerek működhetnek az adatvédelmi kockázatok kezelésére.

4. Arclenyomatok adatvédelmi elemzése

Az arcfelismerő rendszerek általános működését a 3. fejezetben korábban bemutattam. A következőkben az arcfelismerő rendszerek sebezhetőségét fogom elemezni.

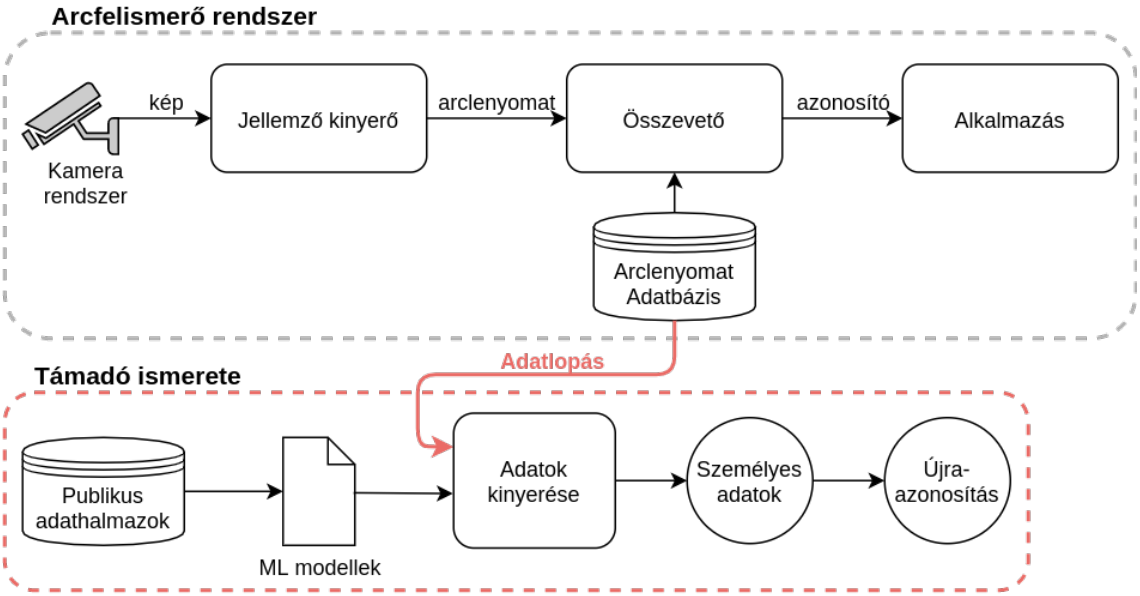
4.1. Támadó modellezése

Az adatvédelmi elemzéshez szükséges definiálnom egy támadó modellt. Tegyük fel, hogy egy cégnél alkalmaznak arcfelismerő rendszert a dolgozók azonosítására. Az épület számos helyisége le van fedve CCTV térfigyelő kamerákkal, amelyek egy központi arcfelismerő rendszernek továbbítják a felvételeket. Új dolgozó regisztrációja során az arcfelismerő rendszer néhány felvétel alapján kiszámolja a dolgozó arcát legjobban leíró arclenyomatot, amit egy központi szerveren tárol. Miután a dolgozó arclenyomata szerepel az adatbázisban, a térfigyelő kamera felvételek alapján lehetséges őt azonosítani. Egy ilyen arcfelismerő rendszer főbb részei (lásd: 15. ábra):

1. **Kamera rendszer**: amely a nyers képkockákat továbbítja az adatfeldolgozó egységnek.
2. **Jellemző kinyerő**: amely az egyes képkockákon végzi a gyorsan lefuttatható arcdetektálást. Ha sikeresen talál emberi arcot az egyik képkockán, arra elvégzi az arc geometriai transzformációját, majd az arcból kinyeri az arclenyomatot.
3. **Adatbázis**: Az ismert, címkézett arclenyomatok tárolására szolgál.
4. **Összevető**: A kinyert arclenyomatot összehasonlítja az adatbázisban tárolt dolgozók arclenyomataival, majd azokon valamilyen távolság metrika (pl: euklideszi távolság) alapján megállapítja a legvalószínűbb találatot. Ha a legkisebb távolság egy bizonyos küszöbértéknél kisebb, akkor a dolgozót sikeresen azonosította, ellenkező esetben ismeretlen személynek nyilvánítja.

Míg az ilyen arcfelismerő rendszer több módon is támadható, a dolgozatom során azzal az esettel foglalkozom, hogy a rosszindulatú fél valamilyen módon hozzáférést nyer az arclenyomat adathalmazhoz, ami lehet egy belső alkalmazott aki kiszivárogtatja az adatbázist, vagy akár egy külső támadó, például hacker aki sikeresen feltöri rendszert. Előfordulhat, hogy a támadó az adatbázisnak egy kisebb részéhez fér hozzá, de dolgozatom során a legerősebb támadót feltételezem, aki a

teljes adatbázishoz hozzáfér, illetve feltételezem, hogy az arclenyomatok titkosítás nélkül vannak tárolva a szerveren.



15. ábra. Az arcfelismerő rendszer és a támadó modellezése.

A támadó modellezését a 15. ábra mutatja be. Miután a támadó valamilyen módon hozzáférést szerzett a központi szerverhez, amin az arclenyomatok vannak tárolva, elképzelhető, hogy képes lesz az arclenyomatokból személyes adatokat kinyerni az adatalanyokról. Személyes adatnak minősül bármilyen adat, ha közvetlenül beazonosítható általa az érintett személy (közvetlen azonosítók) [11]. Személyes adat lehet például a személy demográfiai adatai, mint például a személy életkora, a neme, vagy a rassza.

Feltételezhetjük, hogy a támadó a személyes adatok kinyeréséhez egy saját fejlesztésű algoritmust használ, ami lehet például egy gépi tanulási modell is. Ha a támadó képes megfelelően nagy valószínűséggel, megbízható módon kinyerni a személyes adatokat, akkor a támadó feltételezheti, hogy a kinyert adatok helyesek. A sikeresen kinyert adatokat majd saját célból fel tudja használni a támadó, például újraazonosításos támadáshoz [13]. A személyes adatok kiszivárgásának adatvédelmi kockázatait a 4.4. fejezetben fejtem ki bővebben.

A kérdés az, hogy milyen eljárással lehetséges egy arclenyomatból kinyerni az adatalany érzékeny információit? A munkám során erre a kérdésre kerestem választ. Feltételezésem az volt, hogy az interneten ingyenesen, publikusan elérhető emberi

arcokról készült fotókból összeállítható egy nagy méretű adathalmaz, amihez rendelkezésre állnak a fotón látható valamennyi személy demográfiai adatai, például a pontos vagy becsült életkora, neme és rassza, vagy bármilyen adat ami fotó alapján ember által megadható, és később segíthet az illető azonosításában. Ha sikerül egy ilyen adathalmazt összeállítani, akkor az arcképek alapján az arcfelismerő rendszer működéséhez hasonlóan minden arcképről generálhatunk arclenyomatokat. Ezt követően rendelkezünk az arclenyomatokkal, és a hozzájuk tartozó címkékkel.

Miután megvan az arclenyomat adathalmaz, az alapján képesek vagyunk betanítani egy gépi tanulási modellt, ami az arclenyomatok és a hozzájuk tartozó címkék alapján képes tanulni. A betanítás után a modell használható arra, hogy új, nem látott arclenyomat mintákra becslést adjon. Visszatérve a támadóhoz, ha a támadó rendelkezik egy előre betanított gépi tanulási modellel, azt felhasználhatja arra, hogy a szerverről szerzett arclenyomat adatbázisból érzékeny adatokat nyerjen ki.

A támadó sikerességének feltétele az, hogy hozzá tud férni a szerveren tárolt arclenyomatokhoz, illetve az, hogy a gépi tanulási modell megbízhatóan képes becslést adni a személyes adatokra. A két feltétel közül a másodikkal foglalkozom a továbbiakban. Azt vizsgáltam, hogy modern tanuló algoritmusokkal milyen eredmény érhető el.

4.2. Adathalmazok

Első lépésként szükségem volt egy megfelelően felcímkézett arclenyomat adathalmazra. Ezt az adathalmazt használom fel arra, hogy az alapján tanítsak be gépi tanulási modelleket, amelyek majd képesek becslést adni személyes adatokra. Egy-egy modell csak egy személyes adatra tud becslést adni, ezért több modellre van szükség. Célom az volt, hogy személyes adatok közül az illető életkorát, nemét és rasszát becsüljem meg. Ehhez szükséges tanító mintakészlet, amiben az életkor, nem és rassz meg van címkézve. Elvárás volt még az is, hogy az arclenyomat adathalmazban minden személynek saját azonosító címkéje legyen, illetve lehetőleg minél több arclenyomat tartozzon egy-egy személyhez. Erre azért van szükség, mert az arclenyomatokon vizsgáltam az identifikációs modell teljesítőképességét is.

Mivel a feladatból adódóan az adathalmazzal szemben támasztott elvárások eléggé specifikusak, ezért szükség volt saját adathalmaz létrehozására. Azért, hogy meggyorsítsam a munkámat, kiindulásnak kerestem olyan online elérhető, arcképe-

ket tartalmazó adathalmazt, ami már előre fel van címkézve. Ha létezik egy megfelelő arckép adathalmaz, akkor a képekből egyesével kinyerhetőek az arclenyomat vektorok. Az arckép adathalmazzal szemben az elvárások a következők:

- Lehetőleg kutatási célra lettek létrehozva, az enyémhez hasonló feladatra. Ennek megfelelően az arcképek már elő vannak készítve a feldolgozáshoz (pl. arc kivágása a képből, rossz minőségű képek szűrése).
- Az adathalmaznak megfelelően nagynak kell lennie ahhoz, hogy gépi tanulási modelleket sikeresen be lehessen tanítani.
- Egy emberről lehetőleg minél több kép legyen ahhoz, hogy a generált arclenyomatok alapján az azonosítás jól működjön. Minél több képünk van egy illetőről, annál biztosabban lehet meghatározni az ember arcát legjobban leíró arclenyomatot.
- A képekhez megfelelő címkék tartoznak a demográfiai adatokról. Esetemben a szükséges címkék: az életkor, nem és a rassz.

Nehéz olyan adathalmazt találni, amiben mindhárom számunkra fontos demográfiai adat szerepel. Több arckép adathalmaz jónak tűnt elsőre, de az elvárások közül legalább egynek nem felelt meg. Néhány ilyen adathalmaz amivel foglalkoztam: Labled Faces in the Wild [24], Face Image Project [12], CelebA [38]. A FairFace [30] egy viszonylag új adathalmaz, ami nagyon ígéretesnek tűnt, mert mindhárom demográfiai adatot tartalmazza, és az egyes osztályok aránya kiegyensúlyozott, viszont nem tartalmaz identifikációs címkét.

Végül nem sikerült olyan adathalmazt találnom ami minden kritériumnak megfelel, ezért két külön adathalmazt használtam fel. Az egyik a VGGFace2 [6] amit rassz és nem becslésére használtam fel, a másik pedig az IMDB-WIKI dataset [50], amit életkor becsléshez használtam fel. A két választott arckép adathalmazt szükség volt feldolgozni ahhoz, hogy azokból gépi tanulási modellt lehessen tanítani. A feldolgozás menetét mutatom be a következő részekben.

VGGFace2 adathalmaz

Jelenleg az egyik legnagyobb publikusan elérhető, kutatási célra készült arckép adathalmaz a VGGFace2. Az adathalmaz 3,31 millió arcképet tartalmaz mindössze 9131

emberről, átlagosan 362,6 kép van mindenkiről. A képeket a Google képkeresőjével gyűjtötték össze. Az adathalmaz képein sokféle ember szerepel, eltérő demográfiai adatokkal és eltérő szakmával (pl. vannak színészek, sportolók, politikusok). A képeken látható emberek többféle pózban vannak, sokféle megvilágításban. Az összegyűjtött képeket automatikusan és manuálisan is szűrték.

A VGGFace2 alapból nem tartalmaz rassz címkéket, viszont a Salernói Egyetem MIVIA kutatócsoportja manuálisan felcímkézte az adathalmazt rassz címkékkel, és a munkájukat publikusan elérhetővé tették VMER néven [19]. A VMER-rel kiegészítve a VGGFace2 címkéit, így már van rassz, nem és identifikációs címke is az összes mintához.

Az arcképeket egyesével dolgozta fel egy általam írt Python script, ami az arcképekből kinyeri az arclenyomatokat. Ehhez a face_recognition Python könyvtárat [15] használtam fel, ami a 3. fejezetben taglalt módon találja meg, és alakítja át a képen látható arcokat arclenyomattá. Az arclenyomatok kinyeréséhez a dlib-et [34] használja, így a kapott arclenyomatok 128 dimenziós lebegőpontos vektorok lesznek. Az adathalmazban előfordulnak olyan képek, ahol több ember arca is látható, (például a háttérben elsétál valaki). Ez problémás, mivel ilyenkor nem egyértelmű, hogy a képen látható emberek közül kihez tartozik az annotáció. E miatt a scriptem csak olyan képekkel foglalkozott, ahol pontosan egy arcot sikerült azonosítani. A script-ből egy függvény látható a 1. kódrészleten.

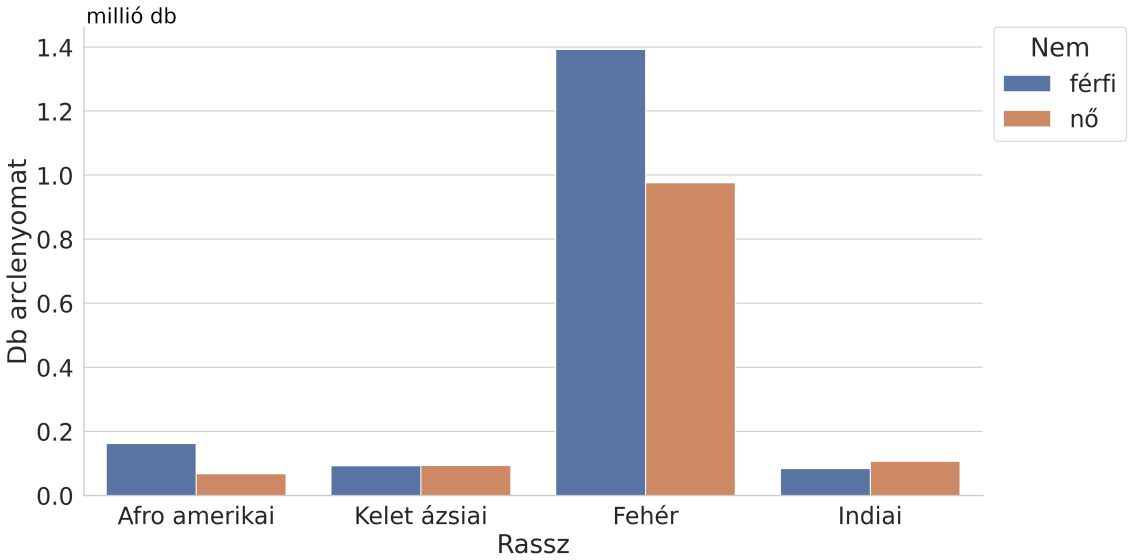
```
def get_encoding(filepath):
  # kep beolvasasa
  image = face_recognition.load_image_file(filepath)
  # arcok detektalasa
  face_locations = face_recognition.face_locations(image,
    number_of_times_to_upsample=1, model="cnn")
  if (len(face_locations) == 1):
    # arclenyomat vektor
    return np.array(face_recognition.face_encodings(image,face_locations))[1]
  return None
```

1. kódrészlet. Arclenyomat vektorok kinyerése.

Mivel 3,3 millió arcképet kellett feldolgozni, a Python scriptet Google Colab-en futtattam, aminek az az előnye, hogy ingyenesen lehet használni korszerű GPU-kat, illetve az adathalmaz feldarabolásával egyszerre párhuzamosan több session-t is lehet futtatni, ami jelentősen felgyorsítja a képek feldolgozását.

A következő lépés az adathalmaz tisztítása volt, azaz kiszűrni azokat a képeket, amelyek valamilyen okból rossz minőségűek (például távoli fotó, rossz fényviszony vagy fura szögből készült a kép), vagy az illető a többi képhez képest nagyon eltérően néz ki. Az arclenyomatok szűréséhez minden személyhez kiszámítottam az átlagos arclenyomatot (centroid), és a centroidtól vett távolság alapján a túl nagy távolságra lévő ($d > 0{,}5$) arclenyomatokat szűrtem az adathalmazból.

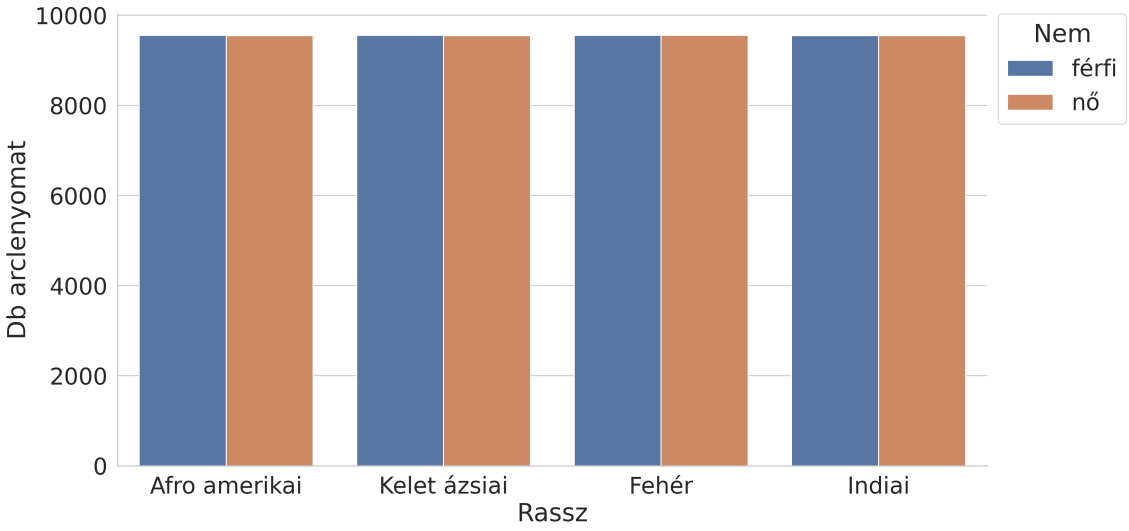
A feldolgozás és szűrés után kapott adathalmaz közel 3 millió arclenyomatot tartalmaz. Minden arclenyomathoz tartozik egy ID címke ami azonosítja a képen látható személyt. Egy ID-hoz legalább 50 arclenyomat tartozik. E mellett van még nem címke (férfi, nő), és rassz címke (African American, East Asian, Caucasian Latin, Asian Indian). Ekkor probléma volt az, hogy az egyes osztályokhoz sokkal több minta tartozott mint más osztályokhoz. A leggyakoribb a fehér férfiak aránya volt, míg a legritkább az afroamerikai nők. Az osztályok arányát a 16. ábra mutatja be.



16. ábra. Osztályok aránya az arclenyomat adathalmazban.

Az osztályok eloszlásának aránytalansága problémát jelent az osztályozó modellek tanításakor, ezért szükséges volt az adathalmazt kiegyensúlyozni. Az adathalmaz kiegyensúlyozása minták eltávolításával érhető el. Legkevesebb kép az afroamerikai nőkről van az adathalmazban, ezért ehhez mérten szűkítettem a többi csoportot. Az embeddingek kivételénél fontos volt, hogy továbbra is legalább 50 kép maradjon minden személyről, ezért nem véletlenszerűen vettem ki a képeket, hanem ID-k alapján csoportosítva. Az egyes demográfiai csoportoknál kilistáztam az oda tartozó

ID-kat, és egyes ID-khoz tartozó képek számát. Az ID-kat képszám szerint csökkenő sorrendben távolítottam el, amíg a csoport meg nem közelítette a szükséges méretet. Ezzel a módszerrel sikerült elérni, hogy minden rassz-nem párhoz ugyanannyi ember tartozott. A kiegyensúlyozott után az osztályok eloszlása a 17. ábrán látható.



17. ábra. Osztályok aránya az kiegyensúlyozás után.

Az eredeti 3,3 millió képből végül 76410 arclenyomat készült, ami továbbra megfelelő méretű adathalmaznak tekinthető. Ez elegendően nagy ahhoz, hogy ez alapján osztályozó modelleket lehessen rajtuk betanítani.

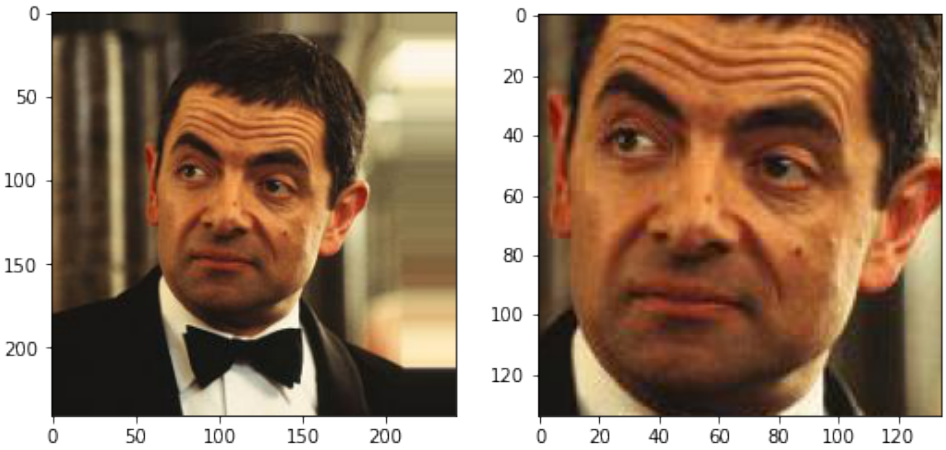
IMDB-WIKI adathalmaz

Mivel a VGGFace2 adathalmaz nem tartalmaz életkor címkét, ezért tovább folytattam a keresést. Választásom az IMDB-WIKI [50] adathalmazra esett. Ez az egyik legnagyobb, nyilvánosan elérhető arckép adathalmaz, ami tartalmaz azonosító címkét, nemet és életkort is. Az adathalmaz két részből áll: IMDB filmekről és filmszínészekből álló adatbázisból kinyert fotókból, illetve a Wikipédiáról szerzett fotókból. Sajnos a Wikipédiáról szerzett képekhez nem tartozik azonosító címke, így csak az IMDB-ről szerzett fotókat használtam fel.

Az IMDB-WIKI adathalmaz képekből, és hozzájuk tartozó címkékből áll. A címkék egy metadata fileban találhatóak, ami többek között tartalmazza a képen látható személy nevét, nemét, a születési dátumát, illetve azt, hogy mikor készült a fotó. Az életkort a képen látható személy születési dátumából, és a kép keletke-

zésének dátuma alapján lehet kiszámolni. A képek jelentős része csoportkép, azaz több arc is látható rajtuk. A képek közül csak azokat használtam fel, ahol pontosan egy arcot lehetett detektálni. Továbbá, kiválasztottam azokat az azonosítókat, amelyekhez legalább 30 fotó tartozik.

A weboldalról letölthetőek az eredeti, teljes méretű képek, illetve a már megvágott csak arcokat tartalmazó képek is. A képeken az arcok középre vannak rendezve 40%-os ráhagyással. A képek feldolgozásánál ezért levágtam ezt a 40%-ot, így gyorsabb a feldolgozás és jelentősen több képen sikerült arcot detektálni. Az arcokhoz tartozó képeket a VGGFace2-nél bemutatott módszerrel alakítottam át arclenyomat vektorokká.



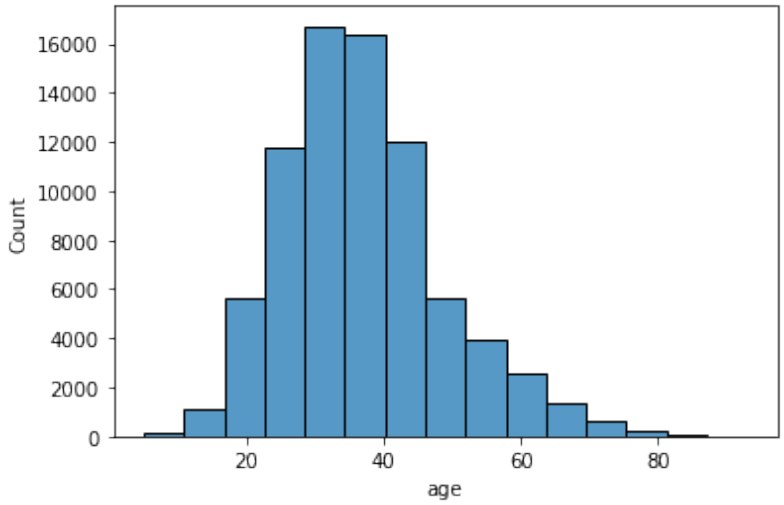
18. ábra. Balra az eredeti kép, jobbra a megvágott kép.

Az adathalmaz képeit vizsgálva azt tapasztaltam, hogy néhány fotón nem a címkének megfelelő személy szerepelt. A hibásan címkézett képek kiszűréséhez a már feldolgozott arclenyomat vektorokat használtam fel. Egy személyhez tartozó arclenyomat vektorok alapján kiszámítottam a vektorok súlypontját (centroidját), és ehhez mért Euklideszi távolságok alapján szűrtem ki a kiugró értékeket. Azt a határt, ami alatt elfogadta az arclenyomat vektor értékét 0,5-re állítottam be, ami viszonylag szigorúnak számít. Ezt az eljárást szemlélteti a 2. kódrészlet.

```
def filter(df, cutoff=0.5):
  encodings = df.iloc[:,4:].values  # egy személyhez tartozó arclenyomatok
  centroid = np.mean(encodings, axis=0)  # súlypont számítás
  distance = np.linalg.norm(encodings - centroid, axis=1)  # euklideszi távolság
  return df.index[distance > cutoff]
```

2. kódrészlet. Arclenyomatok szűrése távolság alapján.

Feldolgozás után közel 90000 arclenyomat vektort kaptam eredményül. Az adathalmazon belül a nemek aránya közel azonos, viszont az életkor eloszlása már kevésbé egyenletes (lásd: 19. ábra). Mivel a képek az IMDB weboldalról lettek összegyűjtve, ezért főleg színészekről, filmrendezőkről vannak képek, akik tipikusan a 20-40-es életkor tartományba esnek. E miatt kiskorúakról és idős emberekről viszonylag kevés kép van az adathalmazban. Ezt az kiegyenlítettlenséget nem tudtam egyszerű módszerekkel megoldani, és később problémát is okozott.



19. ábra. IMDB adathalmazban az életkor eloszlása.

4.3. Modellek betanítása, eredmények

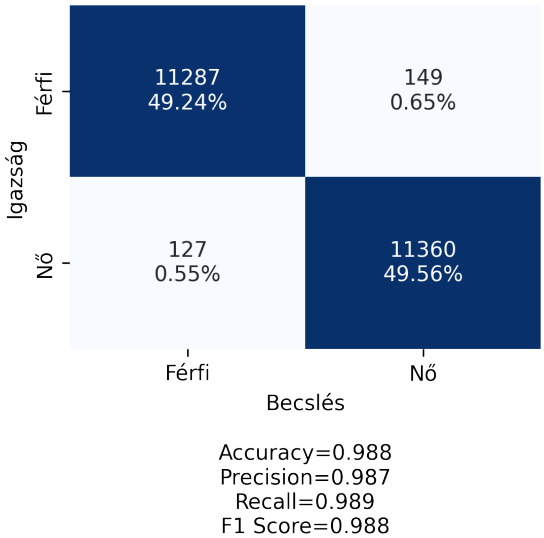
Az elkészült arclenyomat adathalmazok alapján már be lehet tanítani egy osztályozó tanuló algoritmust, ami az arclenyomatok alapján becslést tud adni a személy demográfiai adataira. Gyakorlatban ehhez három osztályozó modellt tanítottam be: egyet a nem predikcióhoz, egyet a rasszhoz, egyet az életkorhoz. E mellett készítettem egy identifikációs modellt is, ami az arclenyomatok alapján azonosítani tudja a személyt. A három osztályozó modellhez véletlen erdő struktúrát használtam.

Az egyik legalapvetőbb osztályozási algoritmus a döntési fa (angol szakirodalomban decision tree). A döntési fa tanítása során a tanító halmazt lépésenként két halmazra bontja szét az adatok különböző jellemzőire teljesülő vagy nem teljesülő feltétel alapján. Ezt a szétválasztó lépést majd sokszor megismétli, minden lépésnél egy-egy elágazást hoz létre. A feltételek alapján szétválogatott adatok így különböző osztályokba kerülnek. A döntési fák általánosítási képessége javítható, ha azokat egy

szakértő együttes (ensemble) struktúrába rendezzük. Ekkor a tanító mintakészletre több, eltérően inicializált döntési fát tanítunk be, amik együttesen egy „véletlen erdőt” képeznek. Ekkora egy bemeneti mintára a véletlen erdő minden fája képez egy becslést, hogy melyik osztályba tartozik a minta. A véletlen erdő kimenete az az osztály lesz, amelyikre a legtöbb döntési fa szavazott. Az osztályozó modellekhez a Scikit-learn Python könyvtár döntési fa implementációját használtam. Ennek a modellnek az az előnye, hogy a használata egyszerű, mivel hiperparaméterek rögzítése nélkül is jó eredményeket lehet elérni.

Nem becslése

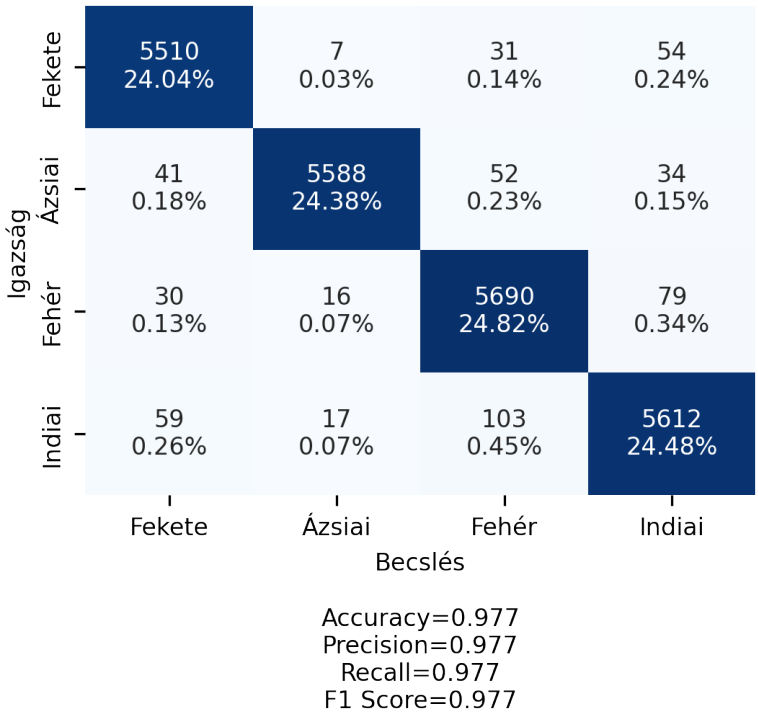
A nem becslő osztályozó modell tanításához a VGGFace2 arclenyomat adathalmazt használtam fel. Az arclenyomatok $N \times 128$ dimenziós mátrix formában vannak, a hozzájuk tartozó címkék $N \times 1$ méretű vektorban, ahol N az adathalmaz mintáinak száma. A teljes mintakészletet felosztottam tanító halmazra és teszt halmazra úgy, hogy a tanító halmaz a minták 70%-át, a teszt halmaz a minták 30%-át tartalmazza. A minták osztályozásához 100 döntési fából álló véletlen erdő modellt használtam. A modell betanítása után a pontosságát a teszt adathalmazon ellenőriztem. A teszt halmaz kb. 23000 mintából állt, amiből a modell csak 276 mintánál tévesztett. A modell jóságát a 20. ábra mutatja, ahol az osztályozó igazságmátrixa látható.



20. ábra. A nemet osztályozó modell igazságmátrixa és pontossági metrikái.

Rassz becslése

A rassz osztályozó modellt a nem osztályozóhoz hasonlóan tanítottam be, azzal a különbséggel, hogy itt bináris osztályozás helyett már 4 osztályunk van, ami nehezebb feladatnak számít. A négy osztály: afro amerikaiak, kelet ázsiaiak, fehérek, és indiaiak. A VGGFace2 adathalmaz kiegyensúlyozottságának köszönhetően itt is sikerült jó pontosságot elérni. A modell igazságmátrixát a 21. ábra mutatja be. A többosztályos osztályozó modell pontossági metrikáinak kiszámításához „macro" átlagoló módszert használtam [52].

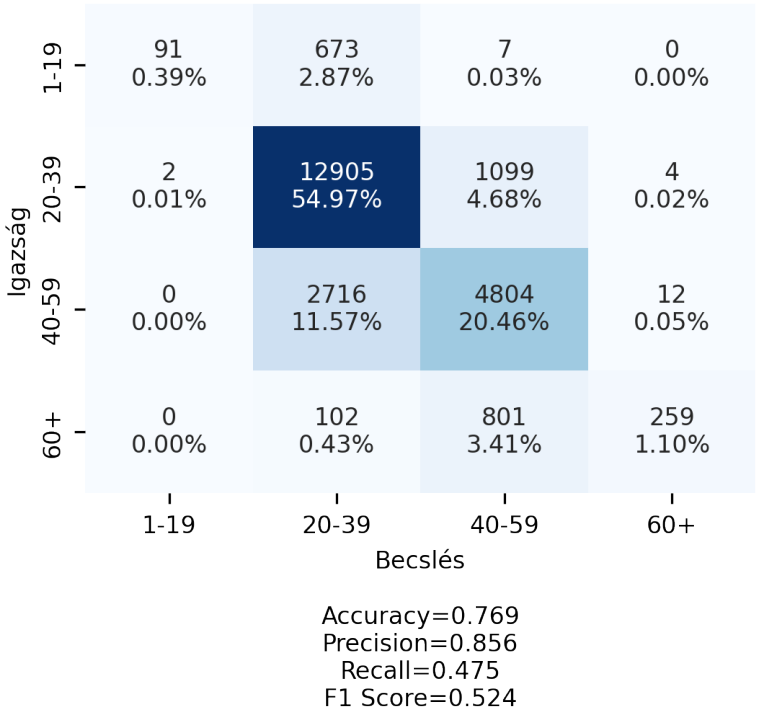


21. ábra. A rassz osztályozó modell igazságmátrixa és pontossági metrikái.

Életkor becslése

Az életkor osztályozó modell tanításához az IMDB-WIKI arclenyomat adathalmazt használtam. Mivel az adathalmazban pontosan szerepelnek az életkorok, először regressziós modellel próbálkoztam, de annak a pontossága nem lett túl jó (R^2 értéke 0,6 körüli volt). Ezt követően az életkorokat csoportosítottam 20 éves intervallumokon. Így négy életkor osztályt kaptam: 1-19 éves, 20-39 éves, 40-59 éves és 60 év fölöttiek

csoportját. Ezzel a módszerrel már jobb eredményt értem el, ami továbbra sem olyan jó, mint amit a korábbi modelleknél sikerült elérnem. Ez belátható annak, hogy az adathalmazban főleg fiatal - középkorú felnőttek vannak (lásd: 19. ábra). A 20 év alattiak és 60 év fölöttiek aránya elég kicsi. A modell jóságát az alábbi 22. ábra mutatja. Mint az ábrán láthatjuk a modell pontossága Acc = 0,77 ami elmarad a korábbi eredményektől. Jobb eredmény eléréséhez több tanító mintára lenne szükség az adathalmazban az 1-19 éves és 60 év feletti osztályokról.



22. ábra. Az életkor osztályozó modell igazságmátrixa és pontossági metrikái.

Identifikációs módszer

Az identifikációs módszerrel meg lehet becsülni a bemeneti arclenyomat alapján, hogy az melyik személyhez (ID-hoz) tartozik. Erre a feladatra a korábbiaktól eltérően nem gépi tanulási modellt alkalmaztam, hanem távolságmérés alapján határoztam meg a legvalószínűbb személyt (a továbbiakban modellként hivatkozom erre a módszerre). Az identifikációs modellnek szüksége van arclenyomat vektorokra, és hozzájuk tartozó címkékre. Ez alapján az egyes címkékhez tartozó arclenyomatok

átlagát számítja ki (amik a centroidok). Új minta becsléséhez a centroidoktól vett távolság alapján találja meg azt az ID-t ami legvalószínűbb találat. A kód kiegészíthető azzal, hogy ha a legkisebb centroid távolság egy határérték felett van, akkor eredményül „nincs találat”-ot vagy „None”-t adjon. Az identifikációs modellel is jó pontosságot sikerült elérni az adathalmazon, csak néhány esetben adott téves becslést. A modell működését a 3. kódrészlet mutatja be.

```
class IDModel:
  def __init__(self, embeddings, ids):
    '''
    embeddings : [np.ndarray] Nx128 - tanito mintak (arclenyomatok)
    ids : [np.ndarray] Nx1 - mintakhoz tartozo azonosito cimkek (pl: 'id00001')
    '''
    self.embeddings = embeddings
    self.ids = ids
    self.unique_ids = np.unique(ids)  # egyedi azonositok
    self.centroids = np.zeros((self.unique_ids.shape[0], embeddings.shape[1]))
    # egy azonositohoz tartozo osszes arclenyomat atlagat szamolom
    for i in range(len(self.unique_ids)):
      id = self.unique_ids[i]
      idx = np.where(ids == id)[0]
      centroid = np.mean(embeddings[idx], axis=0)
      self.centroids[i] = centroid

  def predict(self, new_embeddings):
    # adott arclenyomat es a centroidok tavolsagok osszevetese
    y_hat = np.zeros(len(new_embeddings)).astype('object')
    for i in range(len(new_embeddings)):
      # egy adott arclenyomat tavolsaga az osszes centroidtol
      d = np.linalg.norm(new_embeddings[i] - self.centroids, axis=1, ord=2)
      d_min_idx = np.argmin(d)  # a legkisebb tavolsag kivalasztasa
      y_hat[i] = self.unique_ids[d_min_idx]
    return y_hat

  def score(self, X_test, y_test):
    # becsles pontossaga
    y_hat = self.predict(X_test)
    return np.sum(y_hat == y_test) / len(y_test)
```

3. kódrészlet. Az identifikációs modell működése. Az megadott tanító minták és címkék alapján képes eldönteni egy ismeretlen arclenyomatról, hogy az melyik személyhez tartozik.

Összegezve, a három demográfiai adat közül a nemet és a rasszt nagy pontossággal lehet becsülni, míg az életkor becslése nehezebb feladat. Jobb eredmény érhető el akkor, ha kiegyensúlyozottabb életkor arckép adathalmazból indulunk ki. Tekintve, hogy én csak publikusan és ingyenesen elérhető arckép adathalmazokat vizsgáltam, egy valós támadó ennél jobb pontosságot is elérhet, így az eredményem egy alsó becslésnek tekinthető. A négy modell eredményeit a 1. táblázat foglalja össze.

	ACC	PREC	REC	F1
Nem	0.988	0.987	0.989	0.988
Rassz	0.977	0.977	0.977	0.977
Életkor	0.769	0.856	0.475	0.524
ID	0.998	0.998	0.998	0.997

1. táblázat. Az osztályozó modellek pontossági metrikái. ACC a pontosságra, PREC a precizitásra, REC a visszahívásra, F1 az F1 pontszámra utal.

4.4. Adatvédelmi elemzés

Az előző részben bemutattam, hogy ha a 4.1. fejezetben definiált támadó jó minőségű arclenyomat adathalmaz birtokában van, akkor az alapján olyan modern gépi tanulási modelleket taníthat be, amelyek képesek korábban nem látott arclenyomatokból jó pontossággal személyes adatokat kinyerni. Ebben a részben azzal a kérdéssel foglalkozom, hogy a támadó hogyan tud visszaélni az arclenyomatokból kinyert adatokkal? Egy központi arclenyomat adathalmaz kiszivárgása milyen adatvédelmi kockázatokkal jár?

Kiindulásnak a 29-es munkacsoport anonimizálási eljárásokról szóló véleményét vettem [44], amelyben szó van az adatvédelmi kockázatok csoportjairól. A csoport véleménye alapján, kockázat elemzés során az alábbi három fő kockázattípust célszerű szem előtt tartani:

- **Kiválasztás** (singling out): erről akkor beszélhetünk, ha egy rosszindulatú fél sikeresen be tud azonosítani egy adott személyhez tartozó rekordot az adathalmazon belül.
- **Összekapcsolhatóság** (linkability): legalább két rekord összekapcsolásának képessége, amely ugyanahhoz az érintetthez vagy érintettek csoportjához tartozik. Nem szükséges pontosan beazonosítani az adott személyeket.

- **Következtetés** (inference): ami annak a lehetőségét takarja, hogy nagy valószínűséggel új információ következtethető ki az attribútumok értékeiből.

Mivel egy ember arca tipikusan csak kicsit változik idővel, az arcról készült arclenyomat megváltozhatatlan objektumnak tekinthető. Ez azt jelenti, hogy egy személyről készült arclenyomatok különböző alkalmazásokhoz több adatbázisban is szerepelnek nagyon hasonló formában. Ebből adódóan az egyik kockázati lehetőség az, ha a támadó hozzáfér egy privát arclenyomat adathalmazhoz, majd azt egy másik publikusan elérhető adathalmazzal összekapcsolva képes lehet azon személyek újraazonosítására, akik mindkét adathalmazban szerepelnek [13].

E mellett következtetéses támadás is kockázatot jelent, mivel a támadó az arclenyomat alapján többlet információt képes kikövetkeztetni. Mint azt korábban bemutattam, az arclenyomatok demográfiai adatokat szivárogtathatnak ki.

A kiválasztás veszélye akkor állhat fent, ha a támadó egy bizonyos személyt próbál beazonosítani, és az arclenyomatokból kinyert extra információk alapján képes leszűkíteni a lehetséges személyek halmazát.

Egy másik adatvédelmi kockázat lehet az arclenyomatok alapján az eredeti arckép rekonstrukciója. A [40] szerzői létrehoztak egy szomszédossági dekonvolúciós neurális hálót (NbNet), amely képes visszaállítani az eredeti arcképet egy arclenyomatból. A publikációban bemutatott eredmények alapján, az NbNet magas pontossággal képes visszaállítani az arcképeket, amelyek alapján 2D-s vagy 3D-s arc modell hozható létre. A rekonstruált kép alkalmas lehet arra, hogy arcfelismerő rendszert átverje.

5. Az arclenyomatban kódolt személyes adatok vizsgálata

Az előző fejezetben bemutattam, hogy az arclenyomatokból jó pontossággal kinyerhetőek személyes adatok. A továbbiakban azzal foglalkozom, hogy a személyes adatok hogyan vannak kódolva az arclenyomatokban, melyek azok a jellemzők amelyek alapján következtetni lehet az adatokra. Ehhez kiindulásnak vettem a betanított osztályozó modelleket, és azokon kísérleteztem.

Az elképzelésem az volt, hogy ha képesek vagyunk meghatározni az arclenyomatnak azon jellemzőit, amelyek például az illető rasszáról tartalmaznak információt, akkor azon jellemzők módosításával elérhető lehet, hogy a módosított arclenyomatból ne lehessen következtetni az illető rasszára. Tehát valamilyen technikával maszkolná az információt. Ahhoz, hogy az arclenyomat a hasznosságából ne veszítsen, annak továbbra is használhatónak kell lennie az eredeti célra: az arcfelismerésre.

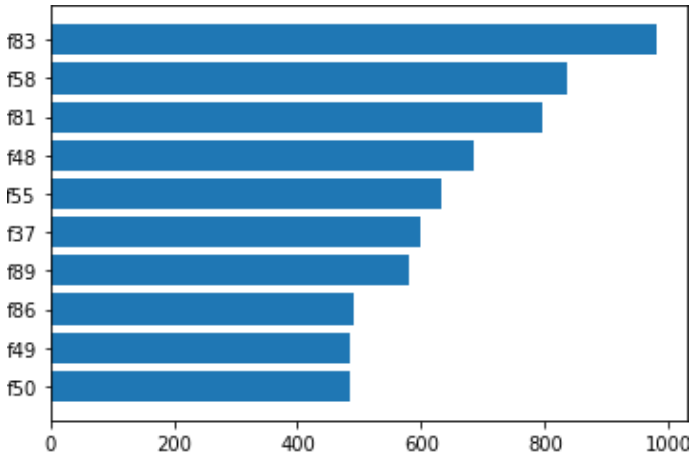
Első próbálkozásom az volt, hogy meghatározzam az arclenyomat azon jellemzőit amelyek a legtöbb információt tartalmazták, azaz azokat amelyek az osztályozó modell számára a legfontosabbak. Feltételezésem az volt, hogy ha módosítom az osztályozó modell számára legfontosabb jellemzők értékeit, akkor annak hatására jelentősen romlani fognak a modell pontossági metrikái. Így elérhető lehet az, hogy az érzékeny adat becslése pontatlan lesz, míg az identifikáció továbbra is jó pontossággal működik. Az elképzelés igazolásához az arclenyomatokat úgy próbáltam módosítani, hogy abból ne lehessen pontosan következtetni a képen látható személy rasszára. Ehhez szükségem volt meghatározni, hogy egy adott modellnek mely jellemzők járulnak leginkább hozzá a modell predikciójához.

A módszer többi demográfiai adatra is használható, de ezek közül a továbbiakban csak a rasszhoz tartozó eredményeket mutatom be.

5.1. Top jellemzők meghatározása

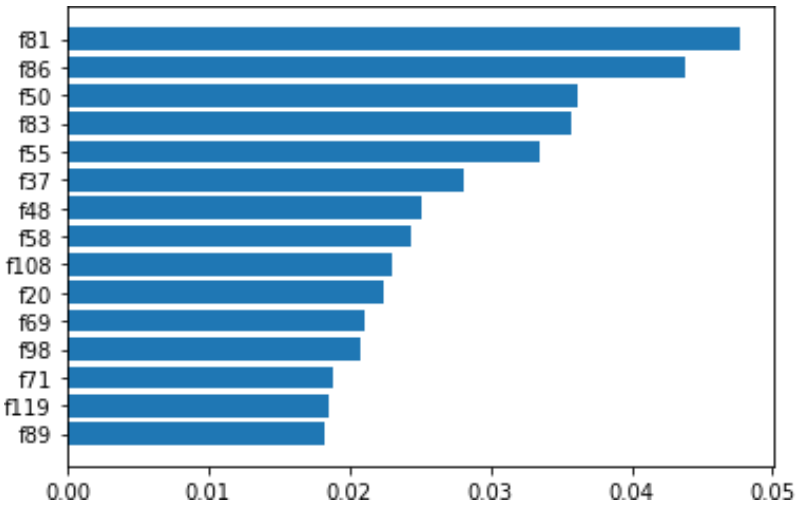
Először a LIME [49] könyvtárat használtam a legfontosabb jellemzők meghatározására, ami kifejezetten gépi tanulási modellek értelmezésére szolgál. Egy adott mintához tartozó modell becsléshez a LIME-mal készíthetünk egy magyarázatot, ami megmondja nekünk, hogy mely jellemzők értéke befolyásolta leginkább a modell

kimenetét. Én azt vizsgáltam, hogy 1000 mintához generált magyarázatok közül melyek azok a jellemzők amelyek leggyakrabban szerepeltek a top 10 legfontosabb jellemzők között. Az eredményeket a 23. ábra mutatja.



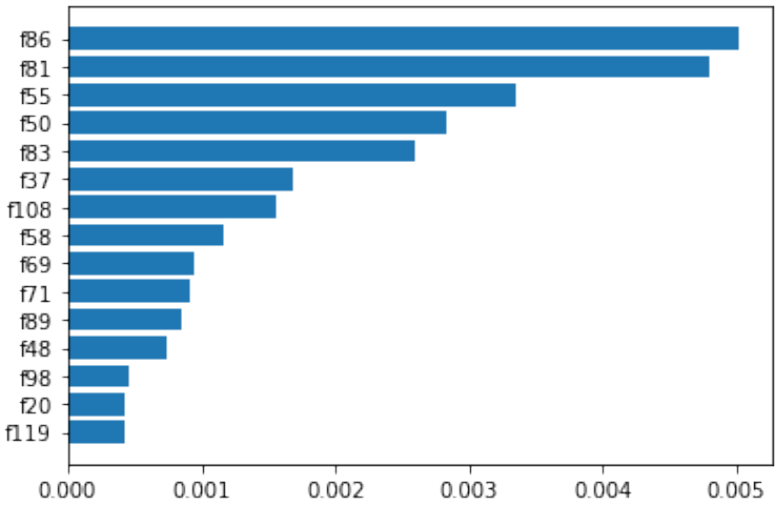
23. ábra. A Top 10 jellemző meghatározása LIME-mal.

Annak érdekében, hogy az így kapott eredményeket megerősítsem, több, eltérő módszerrel is megvizsgáltam a modelleket. A Scikit-learn saját, modell-specifikus implementációját használtam. Ez a módszer kifejezetten döntési fa alapú modellekre alkalmazható. A döntési fákban magasabban található jellemzők nagyobb fontosságúak, mint az alacsonyabban találhatóak. A módszer a Mean Decrease in Impurity és a Gini importance-en alapul [3].



24. ábra. Top 15 jellemző meghatározása a Scikit-learn módszerével.

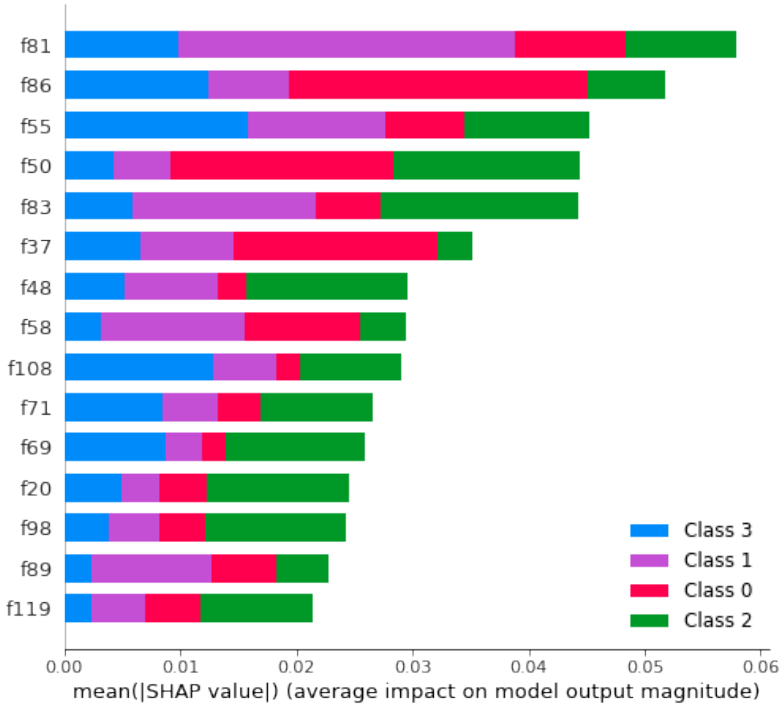
Ez után az eli5 [58] könyvtár Permutation importance implementációját használtam. Egy jellemző fontosságát meghatározhatjuk úgy, hogy megnézzük mennyivel csökken a modell pontossági metrikái (F1, R^2 stb.) ha egy adott jellemzőt eltávolítunk. Egy jellemző eltávolítása után újratanítjuk a modellt, és összehasonlítjuk a pontosságát az eredeti modell pontosságával. Mivel újratanítás szükséges, ez egy eléggé számításigényes eljárás.



25. ábra. Top 15 jellemző meghatározása permutation importance módszerrel.

Tree-SHAP [39] használatával elemezhetjük egy-egy megfigyelés esetén a jellemzők fontosságát, illetve globális összegző képet is kaphatunk az adathalmazról. A Tree-SHAP kifejezetten döntési fa alapú modellekre optimalizált módszer, de használható Kernel SHAP is. Rassz predikció esetén halmozott oszlopdiagramon láthatjuk, hogy osztályonként (itt rasszonként) mely jellemzők a legjelentősebbek. Összegezve őket a korábbiakhoz hasonló eredményeket kapunk. A SHAP-pal kapott eredményt a 26. ábrán láthatjuk.

Összegezve a négy módszerrel hasonló eredményeket kaptam. A legfontosabb jellemzők sorrendje csak kis mértékben tér el a módszerek között. Az egyes módszerek eredményeit összegeztem, majd az alapján létrehoztam egy fontossági listát az összes jellemzőről.



26. ábra. Top 15 jellemző meghatározása Tree-SHAP könyvtárral.

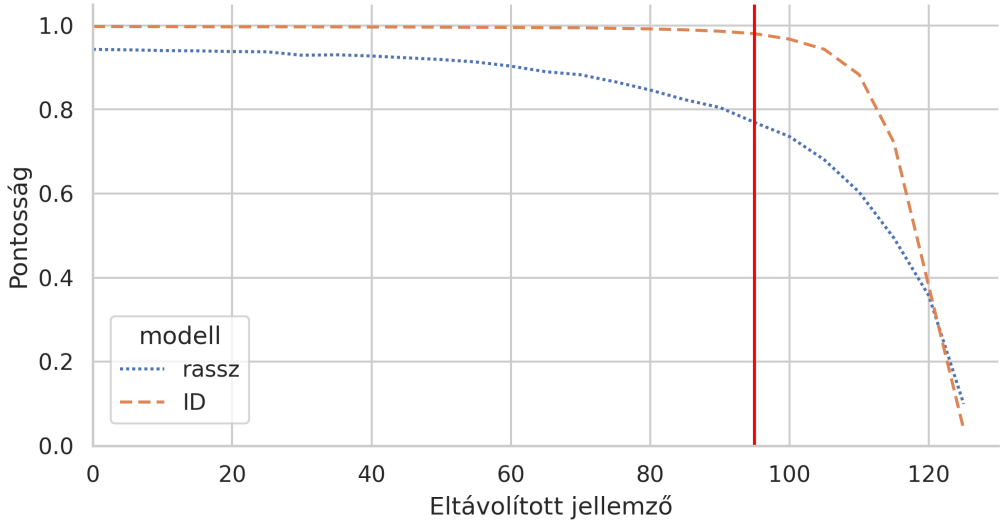
5.2. Legfontosabb jellemzők kivétele.

Miután meghatároztam a rassz osztályozó modell jellemzőinek fontossági sorrendjét, kipróbáltam, hogy a jellemzők eltávolítása milyen hatással van a modell pontosságára. Itt az elgondolás az volt, hogy ha a legfontosabb jellemzők közül 10-20-at eltávolítunk, azzal jelentősen lerontható a rassz modell pontossága. Nyilván ezt azt is jelentené, hogy az identifikációs modell pontossága is romlana valamennyire, viszont várhatóan kevésbé, mint a rassz predikciós modell pontossága. Az identifikációs modell az arclenyomatok alapján minden személyhez kiszámol egy centroidot, ami egy-egy pontként képzelhető el 128 dimenziós térben. A dimenziók csökkentésével a centroidok közelebb kerülnek egymáshoz, de amíg jól elkülönülnek egymástól, addig az identifikáció működhet.

Az elgondolás igazolásához írtam egy Python scriptet, ami az eredetileg $N \times 128$ dimenziójú tanító mintakészlet alapján betanítja a rassz osztályozó modellt, illetve az identifikációs modellt. Ez után iterációnként eltávolítja az 5 legfontosabbnak vélt jellemzőt a mintákból, így az első iterációnál $N \times 123$ dimenziójú mintakészlet marad. A csökkentett dimenziójú mintákon majd egy új rassz osztályozó modellt és

identifikációs modellt tanítok be. Minden iterációban a teszt mintakészleten kiszámolom a modellek pontosságát. A jellemzők kivételét egészen addig folytatom, míg azok el nem fogynak.

A mérés eredményeit a 27. ábra mutatja be. Az x tengelyen a kivett jellemzők számát láthatjuk, az y tengelyen a modellek pontosságát. A narancs szaggatott vonal az identifikációs modell pontossága, a kék pontozott pedig a rassz predikciós modell pontossága. Számomra meglepő módon a modellek rendkívül ellenállóak a jellemzők eltávolításával szemben. A 128 jellemzők felének eltávolításánál sem változott jelentősen a modellek pontossága. Az identifikáció modell egészen a piros függőleges vonalig ($x = 95$, $y_{id} = 0.980$, $y_{race} = 0.769$) nem romlott jelentősen, majd utána meredeken letörik.



27. ábra. Az identifikációs és rassz predikciós modellek pontosságának változása.

A mérésekből azt a következtetést vontam, hogy az egyes jellemzők eltávolítása nem elég az érzékeny adatok védelmére. Az tapasztaltam, hogy a modellek eléggé robusztusok, mivel nagyon sok jellemzőt (80-100) kell eltávolítani ahhoz, hogy a rassz/nem/életkor osztályozó modell pontossága jelentősen romoljon. A korábbi módszerekkel kapott eredmények nem az elvártak szerint alakultak, ezért az egyes jellemzők összefüggőségét kezdtem vizsgálni. Ez a viselkedés az összetett hálózatoknál megfigyelt robusztusságra hasonlít [1]. Azt feltételeztem, hogy a jellemzők értékei között kölcsönös függések állhatnak fent. A következő fejezetben a jellemzők közötti kölcsönös összefüggéseket vizsgálom.

5.3. Hálózat effektus vizsgálata

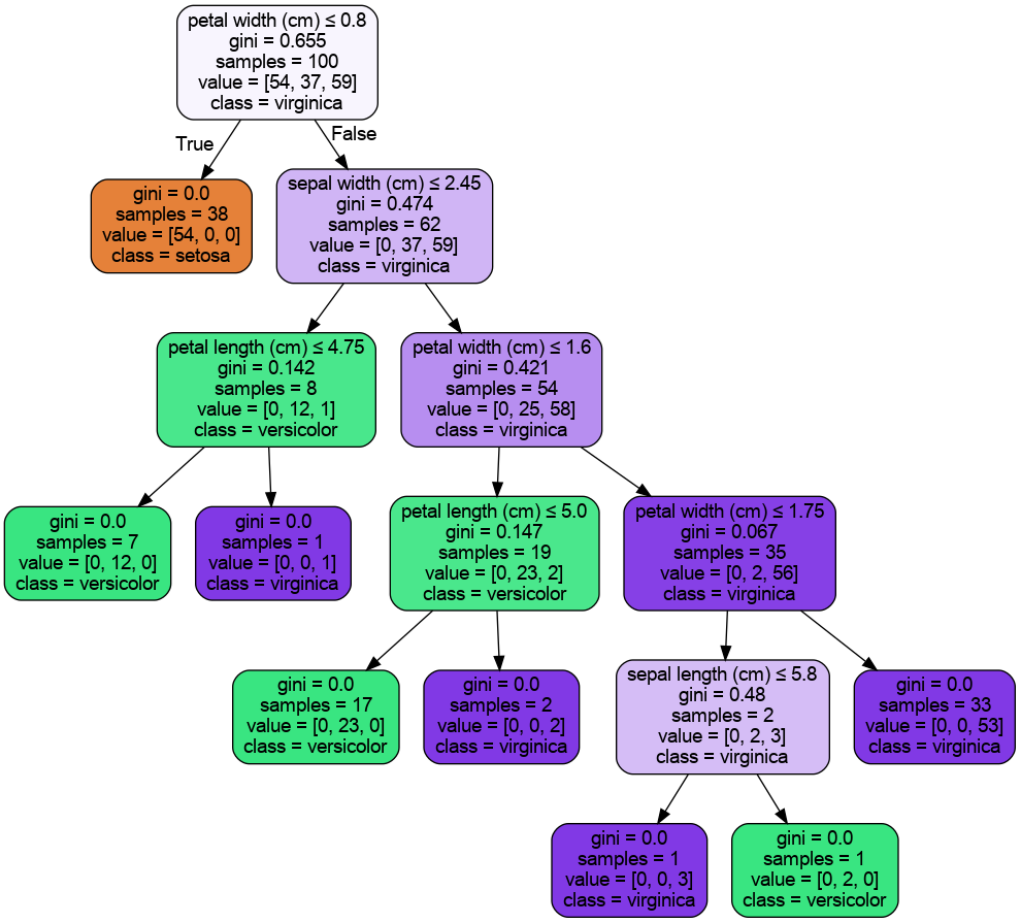
Az arclenyomat vektorok korábbi vizsgálata kapcsán beláttam, hogy a véletlen erdő (angol szakirodalomban random forest) modell döntési fáiban kölcsönös összefüggések állhatnak fent. Az volt a megfigyelésem, hogy a kölcsönös összefüggés hasonlít ahhoz, amit komplex hálózatoknál tapasztalhatóak. Az ilyen hálózatoknál egy csomópont eltávolítása nincs nagy hatással a hálózatra, hanem több csomópont együttes kivételével lehet jelentősen rontani a hálózat összefüggőségén.

Először megnéztem, hogy az arclenyomat vektorokon betanított véletlen erdő modellt át lehet-e transzformálni egy gráf reprezentációba. Miután ez sikerült, az volt a feltételezésem, hogy a rassz predikciós modell és az arc identifikációs modellekből létrehozott gráfok együttes elemzésével találhatok olyan csomópontokat, amelyek szignifikánsak a rassz predikciós gráfnál, viszont nem szignifikáns az identifikációs modell esetén. Így találhatok módot az arclenyomat vektorokban hordozott érzékeny információk védelmére.

Hálózat effektus vizsgálata Iris adathalmazon

A véletlen erdő modell több döntési fát foglal magában. A vizsgálat során alap esetben 100 fából álló modellt alkalmaztam. Egy adott arclenyomat vektor bemenet esetén az erdőben található összes döntési fa képez egy szavazatot az egyik osztályra. Mivel az arclenyomat vektorokra betanított véletlen erdő rendkívül bonyolult döntési fákból áll, melyek működése ember számára már nem igazán értelmezhető, ezért kezdetben egy jelentősen kisebb adathalmazon: az Iris adathalmazon [14] végeztem a döntési fák vizsgálatát. Ez az adathalmaz négy jellemzőt (petal width (PW), petal length (PL), sepal width (SW), sepal length (SL)) tartalmaz, és három kimeneti osztályt, ami háromféle virágnak felel meg: setosa, versicolor, virginica.

Az általam írt algoritmus célja a Random Forest Classifiert átalakítani egy gráf reprezentációba. Az algoritmus először végighalad az erdőben lévő döntési fákon, és egy rekurzív függvény segítségével végig lép a döntési fa egyes csomópontjain. Egy adott csomópont lehet elágazás vagy levél. Elágazás esetén vizsgálni tudjuk az elágazás feltételét, ami egy adott jellemzőből, és egy küszöbértékből (threshold) áll. Az algoritmus működése során feltérképezi a teljes döntési fát, és kigyűjti az egyes levél csomópontokhoz tartozó feltétel rendszereket, azaz milyen feltételeknek kell teljesülnie ahhoz, hogy a döntési fa az adott levélre jusson. Feltételezésem az volt,



28. ábra. Az Iris datasetre betanított véletlen erdő egyik döntési fája vizualizálva.

hogy a fának azon feltételei lesznek szignifikánsak, amelyek gyakran fordulnak elő más feltételekkel együtt. Definiáltam egy NxN-es mátrixot, ahol N a jellemzők számát jelenti, és a mátrixot 0 értékekkel inicializáltam. A fa feltérképezése során a két jellemzőhöz kapcsolódó feltétel együttes megjelenéskor az NxN-es mátrix megfelelő cellájába 1-gyel növeltem az értéket. Ezt a módszert alkalmazva feltérképeztem a véletlen erdő összes döntési fáját, és a fákon belül az összes útvonalat, és feltöltöttem a mátrixot. A 2. táblázatban láthatjuk az eredményt. A jellemzők sorra: SL, SW, PL, PW. Láthatjuk, hogy leggyakrabban a PW és PL feltételek szerepeltek leggyakrabban (1584 alkalommal) feltöltött NxN-es mátrix (N=4).

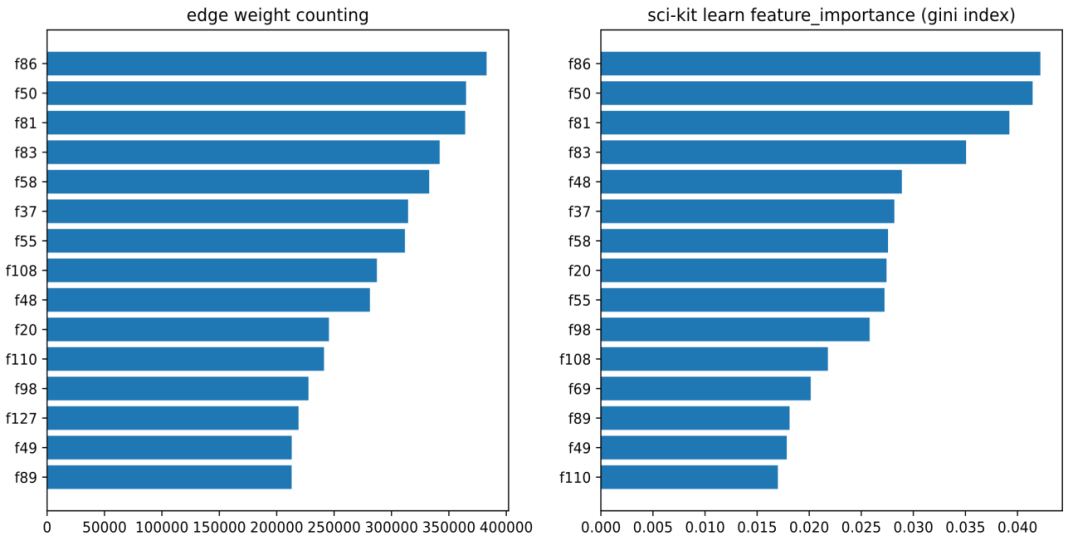
Ezt követően létre tudtam hozni egy olyan gráfot, aminek szomszédossági mátrixa a feltöltött NxN-es mátrix. A létrehozott gráf egy teljes gráf, aminek élei súlyozva vannak feltételek együttes megjelenésének számával.

$$\begin{bmatrix} 0 & 245 & 921 & 842 \\ 0 & 0 & 459 & 462 \\ 0 & 0 & 0 & 1584 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

2. táblázat. Az Iris dataset esetén az értékekkel feltöltött NxN-es mátrix (N=4)

Hálózat effektus vizsgálata arclenyomat vektorokon

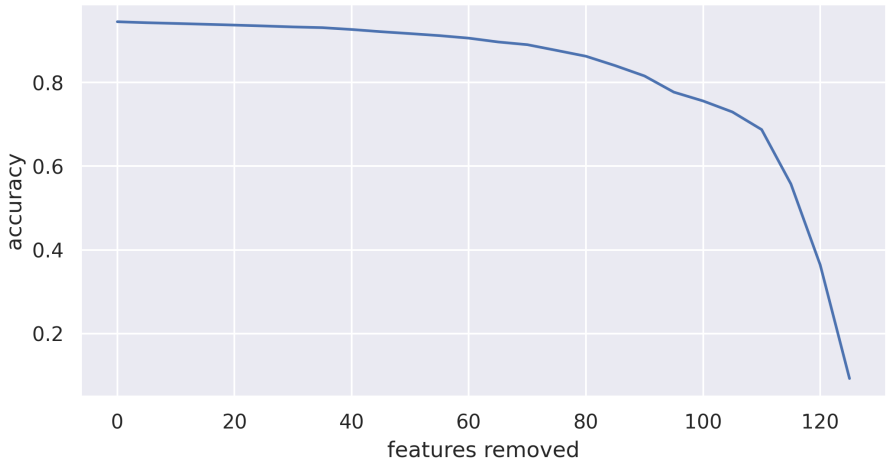
Miután megírtam a fent bemutatott algoritmust az Iris datasetre és leteszteltem a működését kisebb, értelmezhető példákon, áttértem az arclenyomat vektorok vizsgálatára. Az általam használt adathalmaz több mint 10 000 arclenyomat vektort tartalmaz, hozzájuk pedig 4 címke tartozik az adott ember rasszára vonatkozóan (fehér, fekete, ázsiai, indiai). Egy arclenyomat vektor 128 jellemzőt tartalmaz, amelyek valós számok (többnyire (-1, 1) közötti lebegőpontos értékek). Ezen az adathalmazon tanítottam be a rassz predikciós modellt, ami egy Random Forest Classifier 100 döntési fával. A betanított modellen lefuttattam a feltérképező algoritmust, az a korábban leírt módszerek szerint végighalad az összes fa összes útvonalán, és feltöltötte a szomszédossági mátrixot (ami ebben az esetben 128x128-as méretű). A szomszédossági mátrix alapján már létrehozható egy teljes, súlyozott gráf.



29. ábra. Az RFC legfontosabb jellemzői. Bal oldalon az általam írt algoritmus eredménye, jobb oldalon a sci-kit learn függvényével kapott eredmény.

Feltételezésem az volt, hogy a gráf struktúrájából adódóan meg tudok határozni a modell számára szignifikáns jellemzőket amelyeket, ha eltávolítunk az adathalmazból, a modell pontossága jelentősen romlani fog. Egy jellemző fontosságát úgy határoztam meg, hogy a súlyozott gráf jellemzőhöz tartozó csomópontjának vettem a fokszámát (csomóponthoz tartozó élek súlyainak összegét). A fokszámokat ki tudtam számolni a szomszédossági mátrix alapján. A kapott eredményeket csökkenő sorrendbe rendeztem, és összehasonlítottam a Sci-kit learn feature_importance módszerével kapott eredménnyel. A két módszerrel nagyon hasonló eredményt kaptam, amit láthatunk a 29. ábrán.

Miután a modell számára legfontosabb jellemzőket meghatároztam, megnéztem hogyan változik a RFC pontossága, ha eltávolítom a legfontosabb jellemzőket. Mit láthatjuk a 30. ábrán, a modell pontossága nem romlik jelentően még akkor sem, ha jellemzők több mint felét eltávolítottuk.



30. ábra. A rassz predikciós modell pontosságának kirajzolása az eltávolított jellemzők számától függően.

6. Javaslat a kockázatok kiszűrésére

A korábban vizsgált megközelítéssel sajnos nem sikerült jó eredményeket elérni. A modell robusztusságából adódóan néhány jellemző kivételével nem romlik a modell pontossága, és a hálózat effektus feltételezés sem bizonyult helyesnek. Következő próbálkozás az adversarial attack módszerek irányába indult.

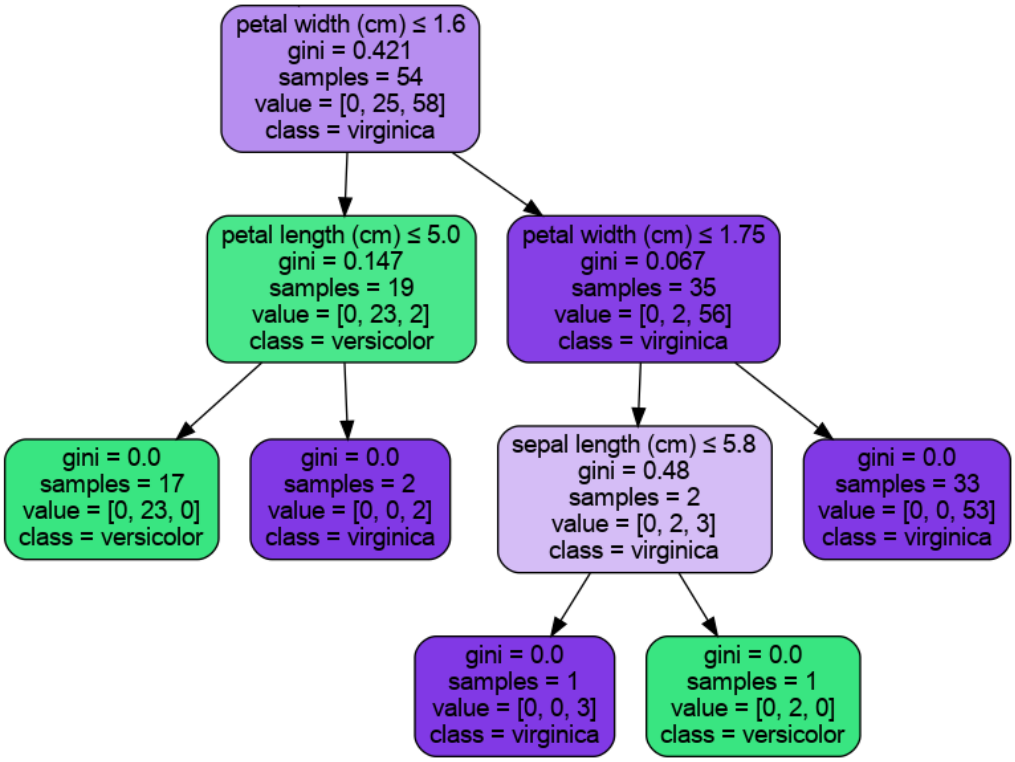
6.1. Adversarial módszerek

Adversarial examplenek nevezzük azt a mintát, amelyet, ha kis mértékben módosítunk képes becsapni egy adott machine learning modellt. Ezt a gyakorlatban egy kis mértékű pontosan kiszámított zaj hozzáadását jelenti, miközben a modell hibáját maximalizálja.

Annak érdekében, hogy belássuk, működhetnek az ilyen módszerek azt tűztem ki célul, hogy mutassam be néhány félig manuálisan készített példán keresztül, hogy az arclenyomatok kis mértékű módosításával be lehet csapni a random forest classifier modellünket. Ehhez készítettem egy újabb algoritmust, amelyet előbb Iris dataseten teszteltem a könnyebb értelmezhetőség érdekében, majd miután működött áttértem az arclenyomat vektorok használatára.

Az algoritmus mögöttes ötlete abból ered, hogy ha megvizsgáljuk az egyes döntési fákat, találhatunk olyan elágazásokat (feltételeket) ahol egy adott arclenyomat vektor jellemzőinek értéke a döntési küszöbértékhez meglehetősen közeli értéket vesz fel. Az ilyen esetekben ezeknek a jellemzőknek kis módosításával el lehet érni azt, hogy a döntési fa feltétele átbillenjen, és másik útvonalon halad a következő elágazásig. Ez a módosítás önmagában még nem garantálja azt, hogy a döntési fa kimenete megváltozik, mivel előfordulhat, hogy a kitérítés után egy másik levélre jut, ami ugyanazt az eredményt adja, mint az eredeti kimenet. Ezt szemlélteti a következő példa.

Tegyük fel, hogy az általunk vizsgált mintában a petal width jellemző értéke 1,55 cm, petal length értéke 4,5 cm. Ezt a mintát a 31. ábrán látható döntési fa alap esetben „versicolor” kategóriába sorolná. A döntési fa a legfelső elágazásnál petal width $\leq$ 1.6 cm feltétel szerepel. Belátható, hogy a bemeneti érték kis módosításával (1.55 + 0.05) átbillenthető a feltétel teljesülése, és az eredeti bal ág helyett jobb ág irányába halad tovább a döntéshozatal, ezzel potenciálisan téves predikci-



31. ábra. Döntési fa részlete, ami az Iris datasetre lett betanítva.

óra vezethető a fa. Azt is láthatjuk ebben a példában, hogy ha a mintánkban a sepal length jellemző értéke nagyobb mint 5,8 cm, akkor újból „versicolor” levélre jutunk, tehát előfordulhat, hogy hiába tévesztettük meg a döntési fát egy ponton, a fa predikciója továbbra is változatlan marad.

A véletlen erdő sok döntési fát tartalmaz, amelyek mind képeznek egy-egy szavazatot az egyik kimenetre, ezért egyetlen fa becsapása legtöbb esetben nem elegendő a teljes véletlen erdő kimenetének módosítására, így a bemeneti arclenyomat vektor olyan módosítása szükséges, ami egyidejűleg több döntési fa predikcióját is képes eltéríteni.

Adversarial módszer fejlesztése Iris adathalmazon

Az Iris dataseten betanított random forest classifieren a korábbihoz hasonlóan egy rekurzív függvényt használtam, ami végighalad az erdőben található egyes döntési fákon, és az elágazásoknál vizsgálja az adott feltételhez tartozó jellemző bemeneti értékét, és küszöbértékét. Az algoritmus első verziója egy adott bemenetre keresi

azokat a küszöbértékeket, amelyek a bemenettől maximum 10%-kal térnek el. Ezeknél a feltételeknél van lehetőségünk kis módosítással megtéveszteni a fát. Az ilyen eseteknél a program kiírja melyik fáknál, és melyik jellemző értéknél van megfelelően kis eltérés a küszöbértékhez képest.

```
Tree: 0 out: 1  feature: PL  4.9 <= 4.85, diff: -0.05
Tree: 1 out: 1  feature: PL  4.9 <= 4.95, diff: 0.05
Tree: 2 out: 1  feature: PW  1.5 <= 1.55, diff: 0.05
Tree: 2 out: 1  feature: PL  4.9 <=  5.0, diff:  0.1
Tree: 3 out: 1  feature: PL  4.9 <= 4.95, diff: 0.05
Tree: 4 out: 1  feature: PL  4.9 <= 4.95, diff: 0.05
Tree: 5 out: 1  feature: PL  4.9 <= 4.95, diff: 0.05
  ⋮
Tree: 99 out: 1  feature: PW  1.5 <= 1.55, diff: 0.05
```

4. kódrészlet. A döntési azon elágazási pontjai, amelyek kis jellezőmódosítással átbillenthetőek.

Az eredményeket megpróbáltam valamilyen módon összesíteni, de voltak esetek mikor a küszöb érték alatt volt a jellemző értéke kicsivel, volt mikor fölötte. Az könnyen látható volt, hogy a petal length feltétel szerepelt a 100 fából leggyakrabban, így annak a módosításával próbálkoztam.

A módszer szemléltetésére kerestem egy mintát, ahol a véletlen erdő fái egyértelműen döntöttek az egyik osztály mellett, és a bemenet kis módosításával próbáltam elérni, hogy minél több fa téves predikcióra jusson. Az alábbi mintánál eredetileg a 96 szavazott a „versicolor” osztályra, majd egyetlen jellemző viszonylag kis módosítással (petal width 4,9-ről 5.05-re lett növelve) sikerült elérni, hogy a fák többsége „virginica” osztályra szavazzon.

```
Eredeti értékek:
[6.9 3.1 4.9 1.5]
Módosított értékek:
[6.9  3.1  5.05 1.5 ]
Módosítás mértéke:
0.88%

Fa szavazatok:
módosítás előtt [ 0. 96.  4.]
módosítás után  [ 0. 40. 60.]
```

5. kódrészlet. A véletlen erdő kimenetének manipulációja.

Adversarial módszer arclenyomat vektorokon

Az Iris datasetes példán elért eredmények után az algoritmus fejlesztését az arclenyomat vektorokról készült adathalmazon folytattam. Az algoritmus működése során továbbra is végighalad a döntési fán és a bemenet értékeitől függően lép egyik elágazásról a következőre. Ennél a verziónál is detektáltam az olyan elágazásokat, ahol a bemeneti érték egy előre definiált tartományon belül tér el a küszöbértéktől. Minden alkalommal, mikor detektál az algoritmus egy ilyen elágazást, az elágazás mindkét ágán folytatja a fa feltérképezését egészen addig, amíg levél csomóponthoz nem jut. A levélhez eljutva megnézi, hogy az adott levélhez tartozó kimeneti osztály megegyezik-e a mintához tartozó címkével, vagy sem.

Itt az az elgondolás, hogy minden esetben mikor könnyen átbillenthető csomóponthoz jutunk, eldönthetjük, hogy melyik úton érdemes tovább haladni. Az egyes lehetséges utakat feltérképezve ki tudjuk választani azt az utat, ami téves predikcióhoz vezet, illetve, ha több ilyen út van akkor ki tudjuk választani azt, amelyikhez a legkisebb beavatkozás szükséges.

Az algoritmus végigmegy a véletlen erdő összes döntési fáin, kivéve azokat, amelyek alapból is téves predikciót adnak, mivel ezeken nem kell módosítanunk.

```
Tree: 4, output: 3
    f10  diff: -0.0267
Tree: 4, output: 2
    f88  diff: -0.04035
Tree: 9, output: 3
    f77  diff: -0.02279
Tree: 11, output: 3
    f93  diff: 0.002902
    f57  diff: -0.008323
    f82  diff: -0.007353
Tree: 11, output: 3
    f93  diff: 0.002902
    f57  diff: -0.008323
    f91  diff: 0.02286
  ⋮
Tree: 48, output: 1
    f93  diff: 0.002753
```

6. kódrészlet. Algoritmus futtatása egy adott arclenyomat vektorra.

Az algoritmus lefuttatható egy adott arclenyomat vektorra. Lefuttatás során a 6. kódrészlethez hasonló információt kapunk. Eredményül egy listát kapunk, ami felsorolja azokat a fákat a véletlen erdőn belül, amelyek kis (10% alatti) input módosítással megtéveszthetőek. Fel van tüntetve a fa sorszáma, illetve a fa megtévesztés utáni predikciója (ennél a példánál helyes predikció 0 lenne.) Egy fához fel vannak sorolva azok a jellemzők amelyeket módosítani kell a fals predikció eléréséhez, ahol a 'diff' érték mutatja a szükséges módosítás mértékét. Ha több jellemző van felsorolva, például Tree 11-nél akkor azok a módosítások ÉS kapcsolatban állnak egymással.

Megfigyelhetjük ezen a példán, hogy egy fához több téves predikció is tartozhat, attól függően, hogy melyik jellemzők értékét változtatjuk. Például a Tree 4 esetén elérhetünk 3-mas vagy 2-es predikciót is. Az is előfordulhat, hogy egy döntési fán belül több útvonal vezet egy téves predikcióhoz, például Tree 11 esetén háromféleképpen kaphatunk 3-mas predikciót. Ilyenkor a kisebb módosítást igénylő út a preferált.

Jelenleg az algoritmus félig automatikus működésű. A lefuttatás során nem fog módosítani az eredeti mintán, azt manuálisan kell elvégezni. Ez az egyik terület, ahol tovább szeretném fejleszteni ezt a megoldást, hogy képes legyen kiszámolni az optimális átalakítást, ahol a legkisebb zaj hozzáadásával megtéveszthető a véletlen erdő becslése. Jelenleg kézzel lehet összeválogatni az egyes jellemzők módosításait, és az alapján összerakni egy zaj vektort. A feladat nehézségét az adja, hogy lehetnek ellentmondásos feltételek (egyik fa növelni szeretné a jellemző értékét, másik pedig csökkenteni), illetve az is előfordulhat, hogy egy jellemző módosítás hatására az egyik fánál korábbi rossz predikció átvált helyesre. A jellemző módosítások optimális kombinációjának megtalálása nem egyszerű feladat.

```
# feature modositasok
x_ = x.copy()

x_[:,10] += -0.0268  # Tree 4, Tree 17
x_[:,77] += -0.0228  # Tree 9
x_[:,43] += -0.02737  # Tree 13, 29, 31
x_[:,8] += 0.02539  # Tree 24
x_[:,88] += -0.0314  # Tree 35
x_[:,105] += -0.02175  # Tree 37
x_[:,44] += 0.02302  # Tree 41

print('tree votes:')
print(count_votes(x))
```

```
print(count_votes(x_))

# kimenet:
'''
tree votes:
[30. 4. 3. 13.]
[20. 4. 4. 22.]
'''
```

7. kódrészlet. Korábbi elemzés alapján manuálisan elvégzett jellemző módosítások.

A 7. kódrészleten láthatjuk, hogyan végezhető el a korábban megállapított jellemzők módosítása. Az eredeti arclenyomat vektor értékeit az „x” változó tárolja. A random forest classifier 50 fából áll, melyek közül 30 fa eredetileg 0 predikciót képez. Egy-egy jellemző módosításnál kommentben megneveztem azokat a fákat, amelyeknek a módosítás hatására átváltanak a helyes (0) predikcióról egy másik értékre. Összesen 10 fa megtévesztésével elérhettük azt, hogy a teljes erdő téves predikciót produkál a módosított mintára.

Fontos még megjegyezni, hogy összességében elég kis módosítással sikerült elérni ezt az eredményt. Az arclenyomat vektor 128 jellemző közül elegendő volt 7-et kiválasztani, és azokat is maximum 10%-ban módosítani. A feltételezésem az, hogy ez a módosítás elegendően specifikus és kis méretű, hogy ezzel még megőrizhető egy arclenyomat vektorok alapján betanított identifikációs modell pontossága, de ezt még nem teszteltem le.

Összefoglalva, eleinte a hálózat effektus feltételezést vizsgáltam, ami végül nem vezetett eredményre. Bemutattam a hipotézis mögött rejtő gondolatmenetet és a kísérleteket. Később az adversarial attack jellegű algoritmus fejlesztésével foglalkoztam. Bemutattam a döntési fák vizsgálatának módszerét és implementálását. Részleteztem az algoritmus fejlesztésének menetét, az egyszerűbb feladatok megoldását, majd az arclenyomat vektorokon elért eredményeket. Jelenleg az algoritmus félig automatikus működésű, azaz a döntési fák elemzése után kézzel kell összeállítani az arclenyomat vektorok módosítását. Az egyik főbb továbbfejlesztési irány az lenne, hogy az algoritmus képes legyen közel optimális zaj generálására, amellyel a véletlen erdő predikciója becsapható.

Ezt az eredményt a munkával töltött első félév végére értem el, 2021 tavaszán. 2021 júniusban megjelent egy új preprint a Stanford Egyetemről, amiben a szerzők az általam bemutatott adversarial módszerhez hasonló adatvédelmi eszközöket véleményezik [46]. A szerzők szerint az ilyen módszerek hatása limitált. Az egyik érvük az, hogy ha az adatot adversarial módszerekkel védjük, és azokat a támadó megszerzi, akkor a támadónak lehetősége van adaptálnia a modelljét, azaz a védett arclenyomatokat felhasználva képes újratanítania a modelljét, ami ellenálló lesz a védekezési mechanizmussal szemben.

Erre hoznak egy példát: a Fawkes rendszert [53], ami a felhasználói számára erős védelmet ígér az arcfelismerő rendszerekkel szemben. A Fawkes a felhasználók által feltöltött képek kisebb perturbálásával érik el azt, hogy az arcfelismerő rendszerek tévesen azonosítsák a képen látható személyeket. A Fawkes rendszer hatékonyan működött és elhíresült, több mint 500 000 felhasználó letöltötte. Ennek ellenére később a Microsoft arcfelismerő rendszerét úgy módosították, hogy a Fawkes védelmével szemben ellenálljon. A Fawkes fejlesztői erre kiadtak egy új frissítést, ami újból védelmet tud nyújtani a Micosofttal szemben, amiről a Fawkes weboldalán olvashatunk [54].

Mivel mindkét fél képes adaptálni a támadó / védekező módszereit, így ez egy „fegyverkezési versenyre” vezethet. Viszont ebben a versenyben a támadónak jelentős előnye van. A Fawkes példában hiába fejlesztenek ki egy újabb verziót a védelmi rendszernek, a támadó már sikeresen feltörte a régi verzió által védett képeket, így azok már nincsenek védve.

A [46] szerzői szerint ez az adversarial módszerek csak ideiglenes védelmet tudnak nyújtani, de nem adnak hosszútávú megoldást. Szerintük megoldásnak az arcfelismerő rendszerek jogszabályokkal való korlátozása jelenthet.

6.2. Kriptográfiai módszerek

A korábban bemutatott módszerek célja az volt, hogy a 4.1. fejezetben bemutatott támadót feltételezve, egy arcfelismeréshez használt központi arclenyomat adathalmaz feltörése esetén az arclenyomatokból ne lehessen érzékeny adatokat kinyerni. Ennek érdekében olyan módszereket vizsgáltam, amelyek az arclenyomatok kis módosításával meggátolná a személyes adatok szivárgását. Ebben a fejezetben egy má-

sik megközelítést, olyan titkosítási módszereket mutatok be, amelyek alkalmazhatóak az arcfelismerés területén.

Egy arclenyomat adathalmaz feltörése azért kritikus biztonsági szempontból, mert az ember arcáról készült arclenyomatok nem változnak idővel. Míg felhasználói jelszavak kiszivárgása esetén van lehetőség a jelszó módosítására, arcfelismerő rendszerek esetén nincs lehetőség arra, hogy valaki megváltoztassa az arclenyomatát. Emellett kockázatot jelen az is, hogy ugyanazt a biometrikus azonosítót több hitelesítési rendszerben is használhatnak. Ha az egyik alkalmazás kiszivárogtatja a biometrikus azonosítót, az összes többi rendszert veszélyezteti, ami ugyanazt a biometrikus adatot használja.

A titkosítási módszerek lényege az, hogy az arclenyomatokat nem eredeti formájukban tároljuk, hanem titkosított formában. A biometrikus adatokat valamilyen módszerrel áttranszformálják, majd transzformált állapotban tárolják. A transzformáció lehet invertálható, vagy nem invertálható (azaz végleges). A titkosítás módjára léteznek biometrikus sablonvédelmi módszerek [45], amelyeket biometrikus adatok (pl. ujjlenyomat, arclenyomat) tárolására fejlesztettek ki. Ezeket mutatom be részletesebben a továbbiakban.

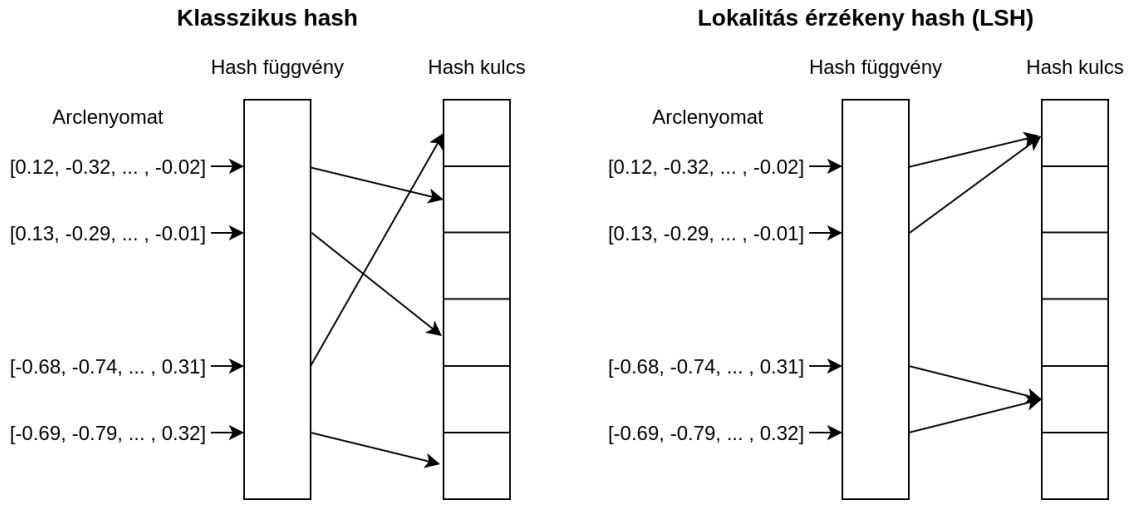
6.2.1. Hashelési módszerek

Az arclenyomatok titkosításához egy egyszerű és bevált módszer lenne a szabványos titkosítási technikák, mint például a hash függvények alkalmazása. A hash függvények a biometrikus adatok védelmére lehet használni, mivel hashelés során az adat végleges átalakításon megy keresztül. A hash függvényeket szinte lehetetlen invertálni, így ha a hash-selt arclenyomatok kiszivárognak, azokból az eredeti arclenyomatokat nem lehet visszanyerni. Emellet előnye még a módszernek az, hogy különböző applikációk eltérő hash függvényt használnának, így ha az egyik applikáció adatbázisát feltörik, az nem jelent veszélyt a többi applikációra.

Egy probléma van a kriptográfiai hashelési eljárásokkal, hogy a hash függvény kimenete már kicsiben eltérő bemenetre is teljesen eltérő lesz. Ez azért probléma, mert a legtöbb biometrikus adat esetén, beleértve az arclenyomatokat is, a környezeti hatások befolyásolhatják az arclenyomatok pontos értékeit. Például egy személyről eltérő fényviszonyokban, vagy más szögekből készült arcképekből származtatott arclenyomatok kis mértékben eltérnek egymástól. Az egymáshoz közeli arclenyomatok

hashelés során nagyban különböző kimenetet kapnak, és emiatt nem alkalmasak összehasonlításra [45].

Ez a probléma megoldható, ha egy olyan hash függvényt használunk, ami a közeli pontoknak azonos kimenetet képez. Az ilyen módszereket lokalitás érzékeny hash függvényeknek hívjuk (angol szakirodalomban locality sensitive hashing, röviden LSH) amelyeket használnak például képek hasonlóságának mérésére [26]. A lokalitás érzékeny hash függvények létezését a Johnson-Lindenstrauss lemma mondja ki, miszerint több magas dimenziójú térbeli pont leképezhető alacsonyabb dimenziójú térbe oly módon, hogy bármely két pont közötti távolság közel azonos marad [27]. Az LSH függvények segítségével adatpontok csoportosíthatóak, hiszen a hasonló bemeneteket nagy valószínűséggel azonos csoportba sorolja az algoritmus. A 32. ábrán látható a különbség a klasszikus hashelés és az LSH között.



32. ábra. Bal oldalon a klasszikus hashelés látható, jobb oldalon az LSH.

Az LSH módszerek már alkalmasak lehetnek arcfelismerő rendszerek esetén az arclenyomatok biztonságos tárolására. Az arcfelismerő rendszer tárolja a hash függvényt, illetve a személyek arclenyomatait hashelt formában. Hitelesítés során az ismeretlen személy arcképéből az arcfelismerő rendszer arclenyomatot képez, amelyet a rendszerben tárolt LSH függvénnyel hashelik, majd összevetik a tárolt hashekkel. Ha a bemeneti képből képzett hash megfelelően közel van az egyik eltárolt sablonhoz, akkor sikeres a hitelesítés.

6.2.2. Titkosítási módszerek

A titkosítás vagy rejtjelezés olyan kriptográfiai eljárás, amellyel valamilyen adatot egy titkosító algoritmus kulcs segítségével átalakítja olyan formára, amely ember számára olvashatatlan. Az adat olvasását csak az teheti meg, aki rendelkezik az olvasáshoz szükséges kulccsal. A titkosított adatok gyakorlatban szinte feltörhetetlenek. Ennek ellenére arclenyomatok védelmére mégsem ideális a klasszikus titkosítási módszerek.

Ennek oka az, hogy az arcfelismerő rendszer működéséhez szükséges az azonosítandó személy arclenyomatát a rendszerben tárolt arclenyomatokkal összevetni valamilyen módszerrel, például az arclenyomatok közötti euklideszi távolság alapján. Ha az adatbázisban tárolt arclenyomatok titkosítva vannak, akkor azokat először szükséges dekódolni az összehasonlításhoz, mert a titkosított adatokon közvetlenül nem tudjuk elvégezni a szükséges műveleteket.

A titkosított adatokkal általánosságban az a probléma, hogy ahhoz, hogy műveleteket végezzünk rajtuk dekódolni kell. Dekódolással viszont egy lehetséges támadási pont nyílik meg a rendszerben, ahol lehetséges hozzáférni az eredeti adatokhoz.

Van erre a problémára egy hatékony megoldás amit homomorfikus titkosításnak hívnak. A homomorfikus titkosítás egy publikus kulcsot használ az adat titkosítására. A klasszikus titkosítási módszerektől eltérően a homomorfikus titkosítással lehetséges a titkosított adatokon matematikai műveleteket végezni. A titkosított adatokon végzett műveletek eredménye is titkosított adat lesz, amelyet dekódolva ugyanazt az eredményt kapjuk, amit a titkosítatlan adatokon elvégzett ugyanezen műveletek eredményeztek volna. Így a hitelesítési fázisban nincs szükség a titkosított adatokat dekódolni, azokon közvetlenül el lehet végezni az összehasonlítást.

A homomorfikus titkosításnak három fő típusa van:

- **Partially homomorphic encryption**: Ebben az esetben csak bizonyos matematikai műveleteket lehet végrehajtani a titkosított adatokon.
- **Somewhat homomorphic encryption**: Korlátozott számú műveletet támogat, amelyek csak meghatározott számú alkalommal hajthatóak végre.
- **Fully homomorphic encryption**: A teljesen homomorfikus titkosítás (FHE) minden műveletet támogat, és a műveletek korlátlan számmal hajthatóak végre. Ez a legerősebb típusa a homomorfikus titkosításnak.

A homomorfikus titkosításnak jelenleg az a legnagyobb hátránya, hogy magas a számításigénye az eljárásnak, ezért elég lassú, és emiatt nem praktikus még az alkalmazása. Az arcfelismerő rendszereknél célszerű lehet egy olyan részlegesen homomorfikus titkosítást alkalmazni, ami gyorsabb működésű, mint az FHE támogatja az összeadást, így távolságméréshez euklideszi távolság helyett lehet például Manhattam távolságot használni.

7. Összefoglaló

Az elmúlt években egyre szélesebb körben elterjedt a gépi tanulás használata, ami a mély tanulás területén elért áttöréseknek, az interneten elérhető nagyméretű adathalmazoknak, és a videokártyák fejlődésének köszönhető. A gépi tanulást számos területen alkalmazzák, többek között a modern arcfelismerő rendszerek is erre a technológiára építenek.

Az arcfelismerő rendszereket széles körben alkalmazzák mind az állami szektorban, illetve a magánszektorban. Az állami szektorban az arcfelismerés hasznos lehet keresett személyek és körözött bűnözők azonosítására, repülőtéri biztonsági ellenőrzésekhez, vagy akár katonai célokra fejlesztett eszközökben is. A magánszektorban is rendkívüli gyorsasággal terjed az arcfelismerés használata. 2008-ban Levono által fejlesztett laptopon jelszó helyett arcfelismeréssel lehetett belépnie a felhasználónak. Számos cég használja az arcfelismerést szolgáltatások vagy funkciók részeként. A Facebook, az Apple és a Google arcfelismerést használ, hogy segítse a személyek megjelölését a képeken. Feltehetően a Facebook rendelkezik a világ legnagyobb kép adathalmazával. Jelentős adatvédelmi aggodalomra adhat okot az, hogy a Facebook képes kombinálni az arc biometrikus adatait a felhasználókról szóló információkkal, beleértve az életrajzi adatokat, a helyadatokat és az ismerősökkel való kapcsolattartást [42]. Bár az arcfelismerő rendszerek nagyon hasznosak, és használatuk gyorsan terjed, jelentős adatvédelmi kockázatokat is hordoznak magukkal.

A gépi tanulási eszközök, arcfelismerés és azzal járó adatvédelmi kockázatoknak az elemzése, és kezelése jelenleg is kutatott területek. Én is érdekesnek tartom ezeket a témaköröket, így ez adta a motivációt a diplomamunkámnak.

A diplomamunkámban bemutattam a generatív modellekhez és az arcfelismeréshez kapcsolódó elméleti hátteret, majd bemutattam a modern arcfelismerő rendszerek felépítését és működését. A modern arcfelismerő rendszerek mély metrika tanulásra támaszkodnak. Működésük során az emberi arcokról készült digitális képekből képesek az arcot jellemző metrikákat, az arclenyomatot előállítani. Az arclenyomatok az arcok jellemző vonásait az eredeti képhez képest csökkentett dimenzióban tárolják. Az arclenyomatok segítségével gyorsan azonosítható a képen látható személy.

Ezt követően az arclenyomatokhoz kapcsolódó adatvédelmi kockázatok feltárásával, és azok lehetséges kezelési módszereivel foglalkoztam. A 4. fejezetben be-

mutattam, hogy az arclenyomatok nem csak személyek azonosítására használhatóak fel, hanem olyan információkat is hordoznak a képen látható személyről, ami személyes adatnak minősül. Bemutattam még, hogy a személyes adatok kiszivárgása miért jelentős, milyen adatvédelmi kockázatokkal jár.

Ehhez először létrehoztam két nagyméretű arclenyomat adathalmazt az interneten szabadon elérhető, címkézett kép adathalmazok feldolgozásával (egyik a VGG-Face2 [6], másik az IMDB-WIKI adathalmaz [50]). A két adathalmaz feldolgozása után azokon gépi tanulási modelleket tanítottam be, amelyek a képen látható személy életkorára, nemére és rasszára tudtak becslést adni. A betanított modellek közül a nem és a rassz becslésére nagyon jó pontosságot sikerült elérnem (rendre 98,8% és 97,7%). Az életkor becslésére kevésbé pontos (76,9%) modellt sikerült betanítanom, ami a tanító adatok kiegyensúlyozatlanságának tudható be.

Ezt követően az 5. fejezetben megvizsgáltam hogyan vannak tárolva a személyes adatok az arclenyomatokban. Az arclenyomat tipikusan 128 lebegőpontos értékből álló vektornak számít. A vektor egyes értékeit jellemzőknek nevezik. Feltételezésem az volt, hogy ha az arclenyomat vektorokból eltávolítjuk azokat a jellemzőket, amelyek legtöbb információt hordozzák magukban az általam vizsgált személy személyes adatairól (életkor, nem, rassz), akkor a módosított arclenyomatokból már nem lehet kinyerni a személyes információt. Ennek igazolására több gépi tanulási modell elemző Python könyvtárral meghatároztam a modellek számára legfontosabbnak vélt jellemzőket, és eltávolítottam azokat fontosság szerinti csökkenő sorrendben. A vártakkal ellentétben néhány jellemző eltávolításával nem sikerült nagy kárt tenni a predikciós modellekben. A modellek rendkívül robusztusok, ellenállóak a jellemzők eltávolításával szemben. Ebből arra következtettem, hogy a jellemzők között kölcsönös összefüggések állhatnak fenn.

A 6. fejezetben adversarial attack módszerekkel kísérleteztem. Az adversarial módszerek lényege az, hogy ha az arclenyomat több jellemzőit együttesen, kis mértékben módosítjuk, úgy megtéveszthető egy gépi tanulási modell, azaz téves becslést ad a módosított arclenyomatra. A módosítás kis mértékű és célzott, ezért a módosított arclenyomat nem veszít a hasznosságából, viszont a személyes adatokat már nem lehet belőlük megbízhatóan kinyerni. A módszer működésére készítettem egy programot, amivel félig manuális módszerrel sikerült példákat létrehoznom, ahol kis módosítás után a modell becslését sikerült félrevezetni. Az adversarial módszerek egyik hátránya, hogy nem nyújtanak hosszútávú védelmet a támadóval szemben,

mivel a támadó a módosított arclenyomatokat felhasználva adaptálni tudja a modelljét, ami képes ellenállni a védekezési mechanizmusnak.

A diplomamunkám végén olyan kriptográfiai módszereket mutattam be, amelyek alkalmazhatóak biometrikus sablonok (pl: ujjlenyomat, arclenyomat) védelmére. A kriptográfiai módszerek lényege az, hogy az arclenyomatokat nem eredeti formájukban tárolják, hanem titkosított formában. A biometrikus adatokat valamilyen módszerrel áttranszformálják, majd transzformált állapotban tárolják. Az általam ajánlott két módszer, az LSH (locality sensitive hashing), illetve a homomorfikus titkosítás. Ez a két módszer megfelelő implementálása már alkalmas lehet az arclenyomatok biztonságos tárolására.

Hivatkozások

[1] Réka Albert – Hawoong Jeong – Albert-László Barabási: Error and attack tolerance of complex networks. *nature*, 406. évf. (2000) 6794. sz., 378–382. p.

[2] G. Bradski: The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. URL `https://opencv.org/`.

[3] Leo Breiman – Jerome H Friedman – Richard A Olshen – Charles J Stone: *Classification and regression trees*. 2017, Routledge.

[4] Jane Bromley – James W Bentz – Léon Bottou – Isabelle Guyon – Yann LeCun – Cliff Moore – Eduard Säckinger – Roopak Shah: Signature verification using a "siamese" time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7. évf. (1993) 04. sz., 669–688. p.

[5] Xianggao Cai – Su Hu – Xiaola Lin: Feature extraction using restricted boltzmann machine for stock price prediction. In *2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE)* (konferenciaanyag), 3. köt. 2012, IEEE, 80–83. p.

[6] Qiong Cao – Li Shen – Weidi Xie – Omkar M Parkhi – Andrew Zisserman: Vggface2: A dataset for recognising faces across pose and age. In *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)* (konferenciaanyag). 2018, IEEE, 67–74. p.

[7] Claude Castelluccia – Daniel Le Métayer Inria: Impact Analysis of Facial Recognition. 2020. február. URL `https://hal.inria.fr/hal-02480647`. working paper or preprint.

[8] Goyal Chirag: Deep understanding of discriminative and generative models in machine learning. https://www.analyticsvidhya.com/blog/2021/07/deep-understanding-of-discriminative-and-generative-models-in-machine-learning/, 2021.

[9] Mengyu Chu – You Xie – Jonas Mayer – Laura Leal-Taixe – Nils Thuerey: Learning Temporal Coherence via Self-Supervision for GAN-based Video Generation (TecoGAN). *ACM Transactions on Graphics (TOG)*, 39. évf. (2020) 4. sz.

[10] Gabe Cohn: Ai art at christie's sells for $432,500. https://www.nytimes.com/2018/10/25/arts/design/ai-art-sold-christies.html, 2018.

[11] European Commission: 2018 reform of eu data protection rules. URL https://gdpr-info.eu/issues/personal-data/.

[12] Eran Eidinger – Roee Enbar – Tal Hassner: Age and gender estimation of unfiltered faces. *IEEE Transactions on Information Forensics and Security*, 9. évf. (2014) 12. sz., 2170–2179. p.

[13] István Fábián – Gábor György Gulyás: De-anonymizing facial recognition embeddings. *INFOCOMMUNICATIONS JOURNAL*, 12. évf. (2020) 2. sz., 50–56. p.

[14] Ronald A Fisher: The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7. évf. (1936) 2. sz., 179–188. p.

[15] Adam Geitgey: Face recognition: recognize and manipulate faces from python or from the command line with the world's simplest face recognition library. https://github.com/ageitgey/face_recognition, 2018.

[16] Zoubin Ghahramani: Unsupervised learning. In *Summer School on Machine Learning* (konferenciaanyag). 2003, Springer, 72–112. p.

[17] Ian Goodfellow: Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[18] Ian Goodfellow – Jean Pouget-Abadie – Mehdi Mirza – Bing Xu – David Warde-Farley – Sherjil Ozair – Aaron Courville – Yoshua Bengio: Generative adversarial nets. *Advances in neural information processing systems*, 27. évf. (2014).

[19] Antonio Greco – Gennaro Percannella – Mario Vento – Vincenzo Vigilante: Benchmarking deep network architectures for ethnicity recognition using a new large face dataset. *Machine Vision and Applications*, 2020.

[20] Masoud Muhammed Hassan – Haval Ismael Hussein – Adel Sabry Eesa – Ramadhan J Mstafan: Face recognition based on gabor feature extraction followed by fastica and lda. *CMC-COMPUTERS MATERIALS & CONTINUA*, 68. évf. (2021) 2. sz., 1637–1659. p.

[21] Luis Herranz: Generative adversarial networks and image-to-image translation. http://www.lherranz.org/2018/08/07/imagetranslation/, 2018.

[22] Geoffrey E Hinton – Richard S Zemel: Autoencoders, minimum description length, and helmholtz free energy. *Advances in neural information processing systems*, 6. évf. (1994), 3–10. p.

[23] Elad Hoffer – Nir Ailon: Deep metric learning using triplet network. In Aasa Feragen – Marcello Pelillo – Marco Loog (szerk.): *Similarity-Based Pattern Recognition* (konferenciaanyag). Cham, 2015, Springer International Publishing, 84–92. p. ISBN 978-3-319-24261-3.

[24] Gary B Huang – Marwan Mattar – Tamara Berg – Eric Learned-Miller: Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition* (konferenciaanyag). 2008.

[25] Ye Jia – Yu Zhang – Ron J Weiss – Quan Wang – Jonathan Shen – Fei Ren – Zhifeng Chen – Patrick Nguyen – Ruoming Pang – Ignacio Lopez Moreno és mások: Transfer learning from speaker verification to multispeaker text-to-speech synthesis. *arXiv preprint arXiv:1806.04558*, 2018.

[26] Yushi Jing – Shumeet Baluja: Visualrank: Applying pagerank to large-scale image search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30. évf. (2008) 11. sz., 1877–1890. p.

[27] William B Johnson – Joram Lindenstrauss: Extensions of lipschitz mappings into a hilbert space 26. *Contemporary mathematics*, 26. évf. (1984).

[28] Jeremy Jordan: Variational autoencoders. https://www.jeremyjordan.me/variational-autoencoders/, 2018.

[29] Kaggle Inc.: Quora question pairs. can you identify question pairs that have the same intent? https://www.kaggle.com/c/quora-question-pairs/overview/, 2017.

[30] Kimmo Karkkainen – Jungseock Joo: Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (konferenciaanyag). 2021, 1548–1558. p.

[31] Tero Karras – Miika Aittala – Samuli Laine – Erik Härkönen – Janne Hellsten – Jaakko Lehtinen – Timo Aila: Alias-free generative adversarial networks. *arXiv preprint arXiv:2106.12423*, 2021. `https://nvlabs.github.io/stylegan3/`.

[32] Mahmut Kaya – H. Bilge: Deep metric learning: A survey. *Symmetry*, 11. évf. (2019. 08), 1066. p. URL `https://www.mdpi.com/2073-8994/11/9/1066`.

[33] Vahid Kazemi – Josephine Sullivan: One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (konferenciaanyag). 2014, 1867–1874. p.

[34] Davis E King: Dlib-ml: A machine learning toolkit. *The Journal of Machine Learning Research*, 10. évf. (2009), 1755–1758. p.

[35] Diederik P Kingma – Max Welling: Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[36] Christian Ledig – Lucas Theis – Ferenc Huszár – Jose Caballero – Andrew Cunningham – Alejandro Acosta – Andrew Aitken – Alykhan Tejani – Johannes Totz – Zehan Wang és mások: Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (konferenciaanyag). 2017, 4681–4690. p.

[37] Yongqi Liu – Qiuli Tong – Zhao Du – Lantao Hu: Content-boosted restricted boltzmann machine for recommendation. In *International Conference on Artificial Neural Networks* (konferenciaanyag). 2014, Springer, 773–780. p.

[38] Ziwei Liu – Ping Luo – Xiaogang Wang – Xiaoou Tang: Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)* (konferenciaanyag). 2015. December.

[39] Scott M. Lundberg – Gabriel Erion – Hugh Chen – Alex DeGrave – Jordan M. Prutkin – Bala Nair – Ronit Katz – Jonathan Himmelfarb – Nisha Bansal – Su-In Lee: From local explanations to global understanding with explainable ai for trees. *Nature Machine Intelligence*, 2. évf. (2020) 1. sz., 2522–5839. p.

[40] Guangcan Mai – Kai Cao – Pong C. Yuen – Anil K. Jain: On the reconstruction of face images from deep face templates. *IEEE Transactions on Pattern Analysis*

and Machine Intelligence, 41. évf. (2019. May) 5. sz., 1188–1202. p. ISSN 1939-3539. URL http://dx.doi.org/10.1109/TPAMI.2018.2827389.

[41] Peter O'Connor – Daniel Neil – Shih-Chii Liu – Tobi Delbruck – Michael Pfeiffer: Real-time classification and sensor fusion with a spiking deep belief network. *Frontiers in neuroscience*, 7. évf. (2013), 178. p.

[42] Office of the Privacy Commissioner of Canada: Automated facial recognition in the public and private sectors. https://www.priv.gc.ca/media/1765/fr_201303_e.pdf, 2013. March.

[43] OpenAI: Generative models. https://openai.com/blog/generative-models/, 2016.

[44] Data Protection Working Party: Opinion 05/2014 on anonymisation techniques. https://ec.europa.eu/justice/article-29/documentation/opinion-recommendation/files/2014/wp216_en.pdf, 2014. April.

[45] Vishal M Patel – Nalini K Ratha – Rama Chellappa: Cancelable biometrics: A review. *IEEE Signal Processing Magazine*, 32. évf. (2015) 5. sz., 54–65. p.

[46] Evani Radiya-Dixit – Florian Tramèr: Data poisoning won't save you from facial recognition. *arXiv preprint arXiv:2106.14851*, 2021.

[47] Ali Razavi – Aaron van den Oord – Oriol Vinyals: Generating diverse high-fidelity images with vq-vae-2. In *Advances in neural information processing systems* (konferenciaanyag). 2019, 14866–14876. p.

[48] Scott Reed – Zeynep Akata – Xinchen Yan – Lajanugen Logeswaran – Bernt Schiele – Honglak Lee: Generative adversarial text to image synthesis. In *International Conference on Machine Learning* (konferenciaanyag). 2016, PMLR, 1060–1069. p.

[49] Marco Tulio Ribeiro – Sameer Singh – Carlos Guestrin: " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (konferenciaanyag). 2016, 1135–1144. p.

[50] Rasmus Rothe – Radu Timofte – Luc Van Gool: Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision*, 126. évf. (2018) 2. sz., 144–157. p.

[51] Vignesh Sampath – Iñaki Maurtua – Juan José Aguilar Martín – Aitor Gutierrez: A survey on generative adversarial networks for imbalance problems in computer vision tasks. *Journal of big Data*, 8. évf. (2021) 1. sz., 1–59. p.

[52] scikit learn: kód dokumentáció: sklearn.metrics.f1_score. `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html`, 2021.

[53] Shawn Shan – Emily Wenger – Jiayun Zhang – Huiying Li – Haitao Zheng – Ben Y Zhao: Fawkes: Protecting privacy against unauthorized deep learning models. In *29th {USENIX} Security Symposium ({USENIX} Security 20)* (konferenciaanyag). 2020, 1589–1604. p.

[54] Shawn Shan – Emily Wenger – Jiayun Zhang – Huiying Li – Haitao Zheng – Ben Y Zhao: Image "cloaking" for personal privacy. `http://sandlab.cs.uchicago.edu/fawkes/`, 2021.

[55] Amir Soleimani – Babak N. Araabi – Kazim Fouladi: Deep multitask metric learning for offline signature verification. *Pattern Recognition Letters*, 80. évf. (2016), 84–90. p. ISSN 0167-8655. URL `https://www.sciencedirect.com/science/article/pii/S0167865516301076`.

[56] Thilo Spinner – Jonas Körner – Jochen Görtler – Oliver Deussen: Towards an interpretable latent space. `https://thilospinner.com/towards-an-interpretable-latent-space/`.

[57] John Tabak: *Geometry: the language of space and form*. 2014, Infobase Publishing, 150. p. ISBN 978-0-8160-6876-0.

[58] TeamHG-Memex: eli5: A library for debugging/inspecting machine learning classifiers and explaining their predictions. `https://github.com/TeamHG-Memex/eli5`, 2016.

[59] Silva Thalles: A short introduction to generative adversarial networks. `https://sthalles.github.io/intro-to-gans/`, 2017.

[60] Mika Westerlund: The emergence of deepfake technology: A review. *Technology Innovation Management Review*, 9. évf. (2019) 11. sz.

[61] Wikipedia: Euclidean distance (csak az ábra). https://en.wikipedia.org/wiki/Euclidean_distance#/media/File:Euclidean_distance_2d.svg, 2021.