BUDAPESTI GAZDASÁGI EGYETEM PÉNZÜGYI ÉS SZÁMVITELI KAR

SZAKDOLGOZAT

Csonka Bertalan
Nappali munkarend
Gazdaságinformatikus
Üzleti adatelemző informatikus

2024

BUDAPESTI GAZDASÁGI EGYETEM
ALKALMAZOTT TUDOMÁNYOK EGYETEME

PÉNZÜGYI ÉS SZÁMVITELI KAR

BUDAPESTI GAZDASÁGI EGYETEM

PÉNZÜGYI ÉS SZÁMVITELI KAR

Generatív modellek a művészetben: Az emberi és gépi kreativitás határai

Belső konzulens: Kuknyó Dániel

Külső konzulens: Dr. Kovács Endre

Csonka Bertalan
Nappali munkarend
Gazdaságinformatikus
Üzleti adatelemző specializáció

2024

1149 Budapest, Buzogány utca 10-12.
Telefon: (+36-1) 469-6600
Fax: (+36-1) 469-6610
www.uni-bge.hu

NYILATKOZAT

Alulírott Csonka Bertalan büntetőjogi felelősségem tudatában nyilatkozom, hogy a szakdolgozatomban foglalt tények és adatok a valóságnak megfelelnek, és az abban leírtak a saját, önálló munkám eredményei.

A szakdolgozatban felhasznált adatokat a szerzői jogvédelem figyelembevételével alkalmaztam.

Ezen szakdolgozat semmilyen része nem került felhasználásra korábban oktatási intézmény más képzésén diplomaszerzés során.

Tudomásul veszem, hogy a szakdolgozatomat az intézmény plágiumellenőrzésnek veti alá.

Budapest, 2024 év 12 hónap 06 nap

Csonka Bertalan

hallgató aláírása

Tartalomjegyzék

1. Bevezetés

Kiskorom óta érdekel a technológia és a művészetek világa. Felettébb érdekesnek tartom, hogy az emberek milyen innovatív módon használják fel a technológiai újdonságokat alkotótevékenységük új szintre emeléséhez. A generatív mesterséges intelligencia fejlődése lehetőséget adott arra, hogy a technológia és a művészetek iránti érdeklődéseimet ötvözni tudjam, melynek köszönhetően egyszerre foglalkozhatok kreatív és technológiai kihívásokkal egyaránt.

A mai világban a mesterséges intelligencia fogalmával az élet minden területén lehet találkozni. Az utóbbi években hatalmas ütemben fejlődött az iparág, melyben úttörőnek számított a nagy nyelvi modellek elterjedése, mint például az OpenAI által fejlesztett ChatGPT megjelenése.

Az AI fejlődése olyan szintre jutott, hogy a mesterséges intelligencia már nemcsak manuális munkát képes helyettesíteni, hanem aktív alkotóként is képes megjelenni a művészetek világában. A generatív modellek különösen érdekesek ezen a területen, mivel nemcsak automatizálni tudnak különböző folyamatokat, hanem teljesen új tartalmakat is képesek létrehozni. Ezek a modellek képesek például képeket, szövegeket, sőt, akár zenéket is generálni, amelyek egyre magasabb, az emberi alkotásokhoz hasonló minőséget képviselnek.

A ruhaiparban például generatív AI segítségével új stílusokat és dizájnokat hoznak létre a ruhatervezők. A Tommy Hilfiger például az IBM-mel együttműködve indított el egy projektet, melyben generatív mesterséges intelligencia segítségével gondolták újra a klasszikus stílusokat, és egy olyan egyedi kollekciót hoztak létre, amely a márka hagyományos megjelenését innovatív, mesterséges intelligencia által generált dizájnokkal ötvözi. A ruhagyártó cégek emellett a generatív AI segítségével csökkenteni tudják a gyártási folyamatok során keletkező szemetet is, az anyagfelhasználás optimalizálásával (TechStoryteller, 2023).

Ez nem csak a ruhaiparban figyelhető meg, számos vállalat használja a fenntartható fejlődés és gyártás megvalósítására. Az orvostudományban is egyre több területen implementálják a generatív mesterséges intelligenciát, az Insilico Medicine például új kezelések fejlesztéséhez, a Sanofi pedig a BioMap generatív mesterséges intelligencia platformjának segítségével gyógyszerek molekuláris struktúrájának optimalizálására használják, így jelentősen csökkentve a költségeket és lerövidítve a gyógyszerkutatás időtartamát (Pratik, 2024).

A generatív modellek rendkívüli hatással vannak az informatika világára is, legyen szó új médiatartalmak generálásáról, vagy az informatika kreatív iparágainak támogatásáról, mint például építőipari látványtervek készítése vagy a játékfejlesztés. A játékfejlesztésben új karakterek, játékkörnyezetek létrehozásához, háromdimenziós modellezéshez használják, vagy akár a hibatesztelésben is nagy szerepet játszik (Bernard, 2024).

1.1. Probléma ismertetése

Az én témámat is egy internetes játék ihlette, melynek neve „Quick, Draw!”. A Quick, Draw! egy internetes böngészőből elérhető játék, melyet a Google fejlesztői készítettek. Azzal a céllal jött létre, hogy teszteljék, illetve bizonyítsák, hogy lehetséges egy neurális hálóval ember által készített rajzokat felismerni és osztályozni.

A játék menete a következő: az internetes oldal megnyitásakor kapni fogunk egy szót, amely lehet például egy állat vagy egy tárgy. A játékosnak az a feladata, hogy ezt a megadott szót lerajzolja a kurzor segítségével megadott időn belül úgy, hogy a rajzot az erre tanított képfelismerő neurális háló felismerje, ezáltal sikeresen kitalálva mit rajzolt a játékos. A google csapata előtérbe helyezi a technológiai fejlődést, illetve ösztönzi az embereket új dolgok kipróbálására, ezért az összes rajz, melyet a játékosok rajzoltak elérhető egy nagy szabad felhasználású adathalmazban. Az adathalmaz már számtalan projekt alapjául szolgált, az én modellem is ezekből az adatokból fog tanulni.

1.2. Kutatási módszerek bemutatása

A kutatásom célja feltárni és megvizsgálni a generáló modelleken belül a feltételes generatív ellenséges hálózatok (Conditional GAN) működését, és tesztelni, hogy milyen minőségű képeket lehetséges generálni egy ilyen generatív modell segítségével. Megkeresem a választ a hipotézisekre, miszerint:

- Hipotézis 1 (H1): Lehetséges olyan magas minőségű képek (rajzok) generálása a „Quick, Draw!” adathalmazon tanított feltételes generatív ellenséges hálózat (conditional Generative Adversarial Network, cGAN) használatával, amelyek eddig nem léteztek.
- Hipotézis 2 (H2): A generált képek megkülönböztethetetlenek az ember által készített képektől, kvantitatív metrikák, illetve kvalitatív emberi értékelések által egyaránt.

Ennek a hipotézisnek a vizsgálata érdekében kvantitatív és kvalitatív módszerekkel is fogom értékelni a generált képek minőségét. A kvantitatív elemzésekhez olyan metrikákat fogok használni, mint az anomália detekció, Inception Score és a Frechet Inception Distance, amelyek segítségével számszerűen mérhetővé válik a képek minősége, változatossága és realizmusa.

A dolgozatom következő részében a Mesterséges intelligencia, gépi tanulás, neurális hálók, illetve a Generatív modellek elméleti hátterét fogom ismertetni. Ezt követően bemutatom a Google által készített Quick, Draw! adathalmazt, majd az általam tanított feltételes generatív ellenséges hálózat architektúráját, felépítését. Részletesen ismertetem a modell tanításának folyamatát, a generátor és diszkriminátor közötti kapcsolatot, majd bemutatom a generálás eredményét. Ezek után az eredmények osztályspecifikus elemzése és értékelése fog következni, majd dolgozatom végén összefoglalom a feldolgozott témát, illetve megválaszolom az állított hipotéziseket. Be szeretném bizonyítani, hogy lehetséges hasonló modellekkel olyan magas minőségű műveket generálni, melyek az emberi alkotóképesség szintjét közelítik meg.

2. Irodalmi áttekintés

2.1. Gépi tanulás alapjai

Az emberiség régóta álmodik olyan gépek megalkotásáról, amelyek képesek emberi módon gondolkodni. Már az első programozható számítógépek megjelenése előtt is tűnődtek azon, hogy vajon lehetséges lesz e valaha a mai gépekhez hasonló technikai eszközök intelligensé válása. Mára a Mesterséges Intelligencia (AI) az informatika világának a leggyorsabban fejlődő ágazata, számos gyakorlati alkalmazással és megannyi kutatási, illetve elvi kérdésekkel. Az intelligens gépektől elvárhatjuk, hogy kisegítsék mindennapi rutinmunkánkat, felismerjék az emberi beszédet, illetve képeket, vagy akár diagnosztikákat végezzenek az orvostudományban.

A gépi tanulás ezekkel a gép számára kihívást jelentő problémákkal veszi fel a harcot, lehetővé téve a számítógépek számára, hogy tapasztalatból tanuljanak, és a világot koncepciók hierarchiáján keresztül ismerjék meg, ahol minden komplex fogalom a gép számára is érthetőbb, egyszerűbb fogalmakra épül. Ezzel a megközelítéssel elkerülhető, hogy a felhasználónak kelljen minden szükséges tudást formálisan előírni a gép számára, amely rendkívül komplex és hatalmas számítási teljesítményt igénylő feladat lenne.

Érdekes módon, azok a feladatok, amelyek az emberek számára a legnehezebb szellemi kihívást jelentik, a számítógépek számára a legkönnyebbek. A hasonló típusú feladatokra kiélezett gépek már régóta képesek legyőzni a legjobb sakkozókat is, de csak a közelmúltban kezdték el megközelíteni az átlagember képességét például objektumok vagy beszéd felismerése terén. A mindennapjainkban rengeteg világgal kapcsolatos tudásra van szükségünk, amelyeket nehéz lefordítani a gép nyelvére. Az egyik legnagyobb kihívás a mesterséges intelligenciában az, hogy miként lehet az ilyen informális tudást a számítógépnek átadni (Goodfellow et al., 2016). Miként tudjuk átadni a gépnek, hogy mi a különbség egy kutya és egy macska között, amikor mindkettő szőrös, 4 lába és 2 füle van? Hogyan tudjuk megkülönböztetni, hogy valaki azért beszél hangosan, mert mérges, vagy azért, hogy mindenki hallja a teremben?

Egy híres érték alapú determinisztikus modell volt például az IBM Deep Blue sakk-rendszere, amely 1997-ben sikeresen legyőzte az akkori világbajnok sakkozót, Garry Kasparovot. A sakk kiváló példája a gépi tanulás erősségének, hiszen a sakktábla 64 mezőt és 32 figurát tartalmaz, amelyek csak és kizárólag előre meghatározott szabályok szerint

mozoghatnak. A sakk szabályai tehát könnyen formalizálhatók, és adhatók meg a gép számára. Tudásbázis-alapú mesterséges intelligenciának nevezik azt, amikor az ezekhez hasonló, világról szóló tudást próbálják lekódolni a gép számára. A mesterséges intelligencia a korai sikereit először olyan feladatok megoldásában érte el, ahol nem volt szükség nagy mennyiségű világról szerzett tudásra (Goodfellow et al., 2016).

A tudás alapú mesterséges intelligenciára épülő gépek nehézségeiből arra következtethetünk, hogy a mesterséges intelligenciának képesnek kell lennie saját tudásának bővítésére, nyers adatokból. Ezt a képességet gépi tanulásnak nevezzük. A gépi tanulás lehetővé tette, hogy a számítógépek olyan problémákat oldjanak meg, amelyek valós világbeli tudást igényelnek (Russell & Norvig, 2020). A gépi tanulás világában többféle módszert is megkülönböztetünk. Ilyen például a felügyelt tanulás, a felügyelet nélküli tanulás, vagy a megerősítéses tanulás.

2.1.1. A felügyelt tanulási módszerek

A felügyelt tanulási algoritmusok lényege, hogy egy bemenethez egy kimenetet rendelnek hozzá, tanulási minták alapján, ahol a bemeneti adatok már a kimeneti címkéket is tartalmazzák. A felügyelt tanulást azért hívjuk így, mert gyakran egy „felügyelő” ember látja el a rendszert a helyes kimeneti címkékkel, de a módszer akkor is felügyeltnek minősül, ha ez a „felügyelő” nincsen jelen a folyamatban (Bishop, 2006; Goodfellow et al., 2016).A felügyelt tanulási algoritmusok közé tartozik például a logisztikus regresszió vagy a tartó vektor gépek.

Osztályozási algoritmusok

Az osztályozás, vagy más néven klasszifikáció olyan feladat, ahol a modellnek egy bemeneti adatot diszkrét osztályokba kell sorolnia. Az ilyen típusú feladatoknál a bemenet lehet például egy kép, a kimenet pedig a modell predikciója, hogy pontosan mi szerepel a képen. A felügyelt tanulásnál a modell megtanulja, hogy a bemeneti adatokhoz milyen címkéket rendeljen, a tanító adatok alapján, amelyben minden adatpont rendelkezik egy helyesen hozzárendelt címkével (Bishop, 2006).

A regresszió célja egy folytonos változó becslése. Ez olyan feladatok esetén alkalmazható, amikor az output egy folytonos változó, nem pedig egy diszkrét kategória. Ilyen például a hőmérséklet becslése. A regressziós modellek az osztályozó modellekhez nagyon hasonlóan tanulnak, de a kimenetük egy folytonos változó lesz nem pedig egy előre definiált osztály (Bishop, 2006).

A felügyelt tanulás célja tehát az, hogy a modell képes legyen a tanítotthoz hasonló struktúrákat felismerni az új, még a modell által nem ismert adatokon. Ehhez megfelelő mennyiségű adat, optimalizációs módszerek és megfelelő modell architektúra szükséges.

2.1.2. A felügyelet nélküli tanulási algoritmusok

A felügyelet nélküli tanulási algoritmusok olyan algoritmusok, amelyek csak a bemeneti jellemzőket használják, hozzárendelt címkék nélkül (Russell & Norvig, 2020). A felügyelet nélküli tanulás célja, hogy az adatokban rejlő mintákat, struktúrákat és rejtett sajátosságokat felfedezze előre definiált címkék nélkül. Ilyen módszerekkel egy hasonló modell képes felismerni az adatok tulajdonságainak természetes eloszlását, csoportosulásait vagy jellemzőit, amelyek segíthetnek a további elemzésekben vagy más tanulási feladatokban (Bishop, 2006; Goodfellow et al., 2016).

Az egyik legfontosabb felügyelet nélküli tanulási módszer a klaszteranalízis, amely az adatok hasonlóság szerinti csoportokba rendezését jelenti. A legismertebb klaszterezési algoritmus a k-közép (k-means), amely előre rögzített számú klasztereket próbál meghatározni az adatok között. Az algoritmus célja, hogy az adathalmaz pontjait úgy csoportosítsa, hogy az azonos klaszterben lévő adatok egymáshoz a lehető legközelebb legyenek (Hastie, Tibshirani, & Friedman, 2009; Bishop, 2006).

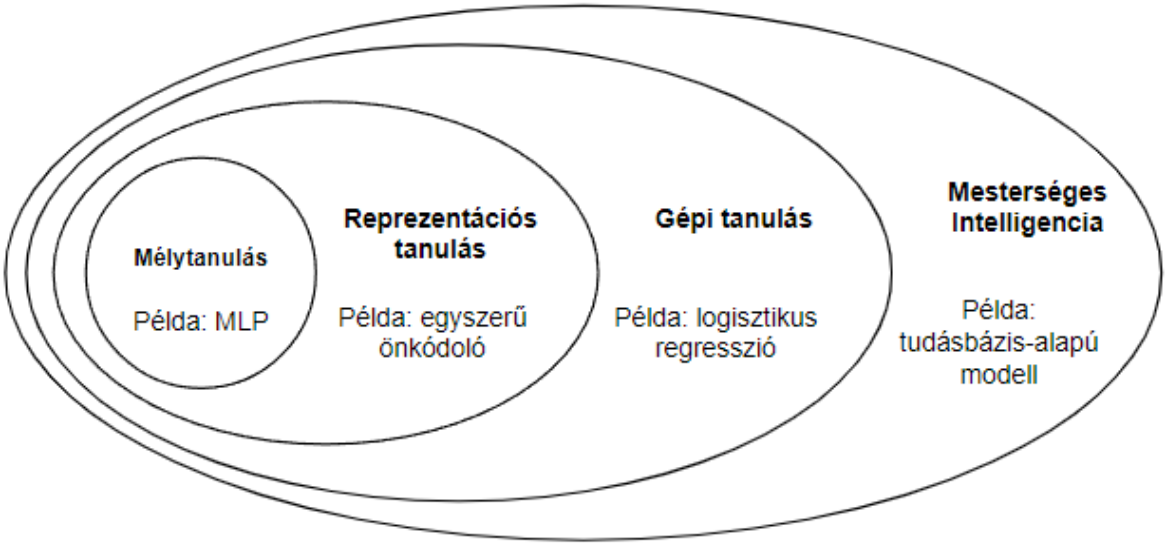
Egy másik felügyelet nélküli tanuláshoz tartozó fogalom a dimenziócsökkentés. A főkomponens-analízis (PCA) egy széles körben alkalmazott többváltozós dimenziócsökkentési módszer. Az elmúlt évtizedekben egyre elterjedtebbé vált a digitális számítógépek fejlődésével, mivel lehetőséget biztosít a magas dimenziószámú adathalmazok egyszerűsítésére. Ez a dimenziócsökkentés lineáris transzformációval történik, amely során az eredeti változókat új, kollektív változókká alakítják át, így néhány „főkomponens” reprezentálja az adathalmaz fő jellemzőit. Manapság a PCA-t felügyelet nélküli gépi tanulási technikaként tartják számon (Kitao, 2022). A felügyelet nélküli tanulás alá sorolandók még a különböző generatív modellek, amelyekről a későbbiekben fogok majd részletesen beszélni.

2.1.3. Mélytanulás

A mélytanulás a mesterséges intelligencia egyik megközelítése. Pontosabban, a gépi tanulás egy típusa, amely lehetővé teszi, hogy a számítógépes rendszerek adatok és valódi tapasztalatok alapján tanuljanak, fejlődjenek (Mehrabi et al., 2019). A következő Venn-

diagram szemlélteti a különböző AI diszciplínák közötti kapcsolatot. A Venn-diagram minden részében egy-egy technológia példája is szerepel.

1. ábra. Venn-diagram a különböző AI diszciplínákról



Forrás: saját szerkesztés (Goodfellow et al., 2016 Deep Learning című művét felhasználva)

A képelemző módszerek mélytanulás során a nyers, szenzorokból kinyert adat (pl: pixelhalmazt) és egy objektumazonosító közötti bonyolult összefüggést egyszerűbb lépésekből építik fel, sok kis lépésre bontják. Az első rétegek alapvető jellemzőket, például éleket azonosítanak, míg a későbbi rétegek már összetettebb struktúrákat, például a képeken szereplő tárgy éleit, sarkait, kontúrjait ismeri fel. A sok egymásra épülő réteg segítségével a modell végül felismeri az adott objektumot a képen (LeCun, Bengio, & Hinton, 2015).

Egy számítógép számára nehéz a nyers adatok megértése, mint például egy képen szereplő dolog jelentése. A pixelek értéke és az azonosított jelentés közötti kapcsolat nagyon összetett, és közvetlenül megoldhatatlannak tűnik a számítógép számára. A mélytanulás ezt a problémát úgy oldja meg, hogy az összetett leképezést egyszerűbb, egymásba ágyazott lépések sorozatára bontja, amelyek mindegyike a modell egy-egy rétegében található meg (LeCun, Bengio, & Hinton, 2015).

Az input, vagyis a bemeneti réteg azokat a változókat tartalmazza, amelyeket közvetlenül megfigyelhetünk. Ezt követően a rejtett rétegek sorozata egyre absztraktabb, ember számára már-már érthetetlen jellemzőket von ki a képből. Ezeket a rétegeket rejtett rétegeknek nevezzük, mivel értékeiket nem a rendelkezésre álló adatok határozzák meg, a

modellnek kell megtanulnia, mely koncepciók segítenek megmagyarázni a megfigyelt adatok közötti összefüggéseket (Russell & Norvig, 2020; Goodfellow et al., 2016) .

Minél mélyebb egy réteg, annál absztraktabb jellemzőkkel dolgozik. Az első réteg magasabb szintű jellemzőket ismer fel, például az éleket a szomszédos pixelek fényerejének összehasonlításával. Az első rejtett réteg leírása alapján a második réteg már sarkokat és kontúrokat keres és ismer fel. A harmadik réteg már képes felismerni bizonyos részeit a tárgynak, amelyek az előző rétegek által összegyűjtött kontúrok és sarkok kombinációjából állnak elő. Végül ezek a rétegek által létrehozott leírás lehetővé teszi a modell számára a képen szereplő objektum azonosítását (LeCun, Bengio, & Hinton, 2015).

Nincs egyetlen helyes válasz arra, hogy mekkora is pontosan egy architektúra mélysége. hasonlóan egy számítógépes program hosszához, ennek sincs egyetlen, univerzálisan helyes értéke. Nincs konszenzus annak terén sem, hogy mekkora mélység szükséges, hogy egy modell „mélynek” minősüljön, azonban a mélytanulás általában olyan modellekre értendő, amelyek a hagyományos gépi tanulásnál nagyobb mértékben építenek egymásra tanított függvényeket vagy koncepciókat (Goodfellow et al., 2016).

2.2. Neuronhálók és gépi tanulás

A neurális hálók megalkotásához a kutatók a biológiai idegrendszer, különösen az emberi agy felépítéséből merítettek ihletet. A neurális hálók is hasonló módon működnek, csak egy leegyszerűsített formában (Haykin, 1998).

Ian Goodfellow, Yoshua Bengio, és Aaron Courville Deep Learning című könyvében kifejtik, hogy a mesterséges neurális hálók bár biológiai inspirációt nyertek, jelentősen leegyszerűsítik az agyban végbemenő folyamatokat. Az emberi agy neuronjaival ellentétben, amelyek különböző komplex kémiai folyamatokat alkalmaznak, a mesterséges neuronok egyszerű matematikai függvényekkel dolgoznak. Ettől függetlenül az alapelv hasonló: egy bemeneti jel több lépésen keresztül halad át, és a végén egy új, „tanult” információt kapunk.

Az ilyen algoritmusok teljesítménye nagymértékben függ az adatok reprezentációjától. Az algoritmusok csak akkor működnek jól, ha helyes és releváns adatokat kapnak. Az adatok megfelelő használata, strukturált felépítése és indexek használata exponenciális hatással lehet az algoritmusok teljesítményére. Sok esetben azonban nehéz meghatározni milyen jellemzőket érdemes kiemelni az adatokból. Például egy algoritmusnak, amely autókat ismer fel, nehéz leírni pontosan hogyan is néz ki egy kerék pixelértékek szintjén, mivel az függhet a kormánykerék állásától, a rávetülő fény erősségétől, vagy éppen az autó alatti talajtól. Az egyik megoldás erre a problémára a reprezentációs tanulás, ahol a gép nem csak az adatokat dolgozza fel, hanem megtanulja a releváns jellemzők kinyerését is (Ng, n.d.; Goodfellow et al., 2016).

A reprezentációs tanulás kiváló példája az autoencoder, amely egy olyan neurális háló, amely a bemeneti adatot új reprezentációvá tömörít (alacsonyabb dimenziós vektortérbe), majd visszaalakítja eredeti formátumba úgy, hogy eközben megőrzi a lényegi információkat. Az ilyen algoritmusok segítenek elkülöníteni a változó tényezőket, amelyek az adat változatosságát magyarázzák, mint például az autó képe esetén a pozíció, szín és napsugárzás szöge (Goodfellow et al., 2016).

Az ilyen tényezők gyakran nem közvetlenül megfigyelhetőek, inkább rejtett vonások, amelyek hatással vannak az adatokra. Ezek lehetnek elméleti konstrukciók az emberi elmében, amelyek segítenek gyorsabban kinyerni adatokból a megértéshez szükséges információkat. Felfoghatók fogalmakként, amelyek segítenek eligazodni az adatok sokszínűségében. Például egy beszédhang elemzése során ilyen változók közé tartozik a beszélő életkora, neme, akcentusa (Goodfellow et al., 2016).

Az egyik legnagyobb nehézség a mesterséges intelligencia alkalmazásában az, hogy sok ilyen tényező befolyásolja az adat minden egyes elemét. Az ilyen jellemzők közül csak azokat kellene kinyerni, amelyek valóban fontosak az adott feladat szempontjából, és amelyekre feltétlenül szükségünk van. A nyers adatokból ezen magas szintű absztrakt jellemzők kivonása rendkívül nehéz feladat. Ha a reprezentáció előállítása van olyan nehéz, mint maga a probléma megoldása, a reprezentációs tanulás nem túl hasznos (Goodfellow et al., 2016).

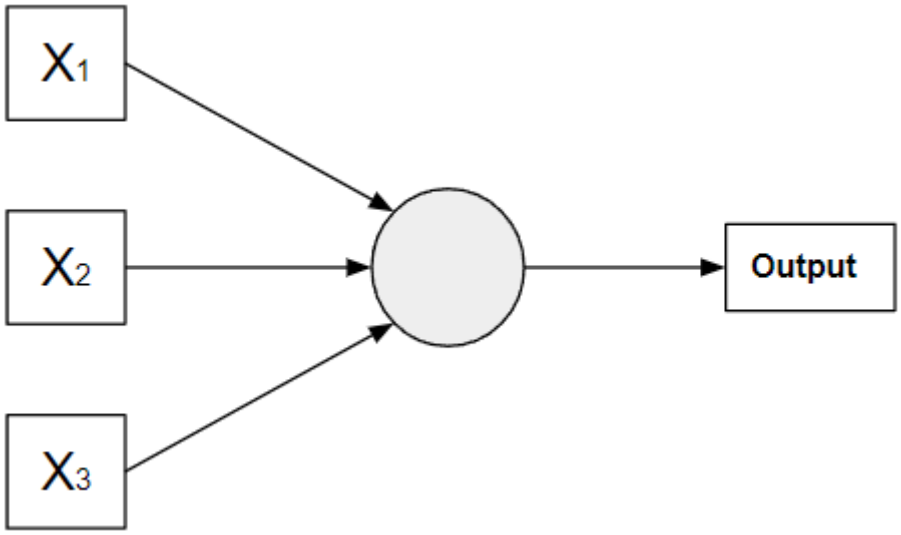
2.2.1. Perceptron és működése

Az egyszerűbb gépi tanulási algoritmusok nehezen kezelik az összetett adatokban rejlő bonyolult mintázatokat. A mélytanulás, vagyis Deep Learning ezt a problémát úgy orvosolja, hogy az összetett feladatot egyszerűbb fogalmakra bontja. Klasszikus példa a mélytanulásra a többrétegű perceptron (MLP). Ebben a matematikai függvényben minden réteg újabb reprezentációt ad az adatokról, lehetővé téve a számítógép számára, hogy bonyolultabb, többlépcsős programot tanuljon meg (Fazekas, 2013, Nielsen, 2015).

A perceptron a neurális hálózatok egyik alapvető egysége, amely a biológiai neuronok működését veszi alapul. Minden perceptronhoz több bemenet tartozik, amelyekhez egy-egy súly kapcsolódik. Ezeket a bemeneteket a súlyokkal szorozza meg, majd a szorzatokat összegzi, végül a perceptron egy aktivációs függvényt alkalmaz az eredményen, hogy meghatározza a kimenetet. Az eredeti perceptron modellt Frank Rosenblatt fejtette ki, amely bináris kimeneteket adott (Fazekas, 2013; Nielsen, 2015).

A perceptron több bemenettel is rendelkezhet, mint pl. x_1, x_2 és x_3, és egy bináris kimenetet állít elő, amely lehet 0 vagy 1 (Fazekas, 2013; Nielsen, 2015).

2. ábra. Egyszerű perceptron ábrázolva



Forrás: saját szerkesztés (Fazekas István: Neurális hálózatok című művét felhasználva)

A bemenetek fontosságát súlyok határozzák meg, mint w_1, w_2, w_3. A kimenet attól függ, hogy a súlyozott összeg meghaladja-e a küszöbértéket. Ha a súlyozott összeg nagyobb a küszöbértéknél, a perceptron kimeneti értéke 1 lesz, ha kisebb annál, akkor pedig 0 (Fazekas, 2013; Nielsen, 2015).

$$output = \begin{cases} 0\,, ha \sum_j w_j x_j \ \leq \ küszöbérték \\ 1\,, ha \sum_j w_j x_j \ > \ küszöbérték \end{cases}$$

A perceptronok egyszerűsített változata a bemeneti értékek súlyozott összegét a súlyok (w) és a bemenetek (x) vektorainak skalárszorzataként ($w*x$) írja fel. A küszöbérték helyett bevezeti az eltolási értéket, vagyis a torzítást (bias), amely megegyezik a küszöbérték negatív értékével (Fazekas, 2013; Nielsen, 2015).

A torzítást úgy is értelmezhetjük, mint egy mérték, hogy milyen könnyű rávenni a perceptront, hogy 1 legyen a kimenete, vagy biológiai fogalmat használva, hogy az adott neuron (perceptron) tüzeljen. Ebben az esetben a kimenet előállításához használt függvény a lépésfüggvény, amely a bemenet értéke alapján dönt a kimenetről (Fazekas, 2013; Nielsen, 2015).

A lépésfüggvény képlete: $f(x) = \begin{cases} 1\,, ha\, x \geq \ 0 \\ 0\,, ha\, x < \ 0 \end{cases}$

A többrétegű perceptron (MLP) ennek a modellnek a kiterjesztése. Az MLP bevezeti a rejtett rétegek használatát, amelyek, mint már korábban is említettem, a bemeneti és kimeneti

rétegek között helyezkednek el. Ezek a rétegek teszik a hálózat számára lehetővé, hogy összetett mintázatokat ismerjen fel az adatokban (Nielsen, 2015).

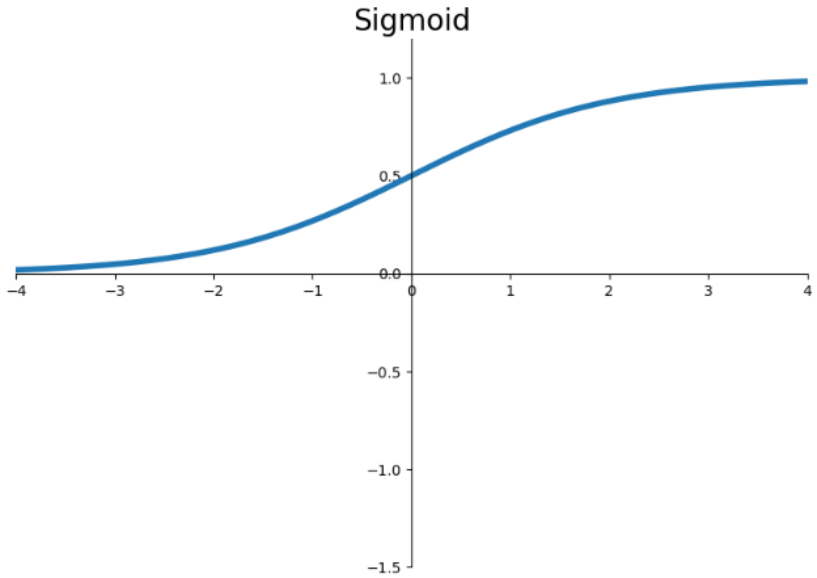
2.2.2. Aktivációs függvények

Az aktivációs függvények fontos szerepet játszanak abban, hogy a hálózat képes legyen nemlineáris problémák megoldására. Míg a perceptron csak egyszerű feladatokra alkalmas, a több rétegű struktúra és különféle aktivációs függvények lehetővé teszik, hogy az MLP komplexebb, valós világban is alkalmazható problémákra nyújtson megoldást (Nielsen, 2015).

A leggyakrabban használt aktivációs függvény a szigmoid (másik nevén logisztikus szigmoid) függvény (σ), amelynek a következő a képlete: $\sigma(z) \equiv \frac{1}{1+e^{-z}}$

Amikor tehát a $z = w * x + b$ értéke nagy és pozitív, a sigmoid (σ) 1-hez nagyon közeli értéket fog adni. Amikor $z = w * x + b$ értéke nagy és negatív, kimeneti értéke 0-hoz nagyon közeli értéket fog adni. Tehát a szigmoid függvény segítségével lineáris kimenetet kapunk, azonban ez a linearitás csak akkor figyelhető meg, amikor a bemeneti érték, vagyis $w * x + b$ értéke mérsékelt nagyságú, mivel, ha nagyon alacsony vagy nagyon magas, akkor egy perceptronhoz hasonlóan 0 vagy 1 lesz a kimenet (Nielsen, 2015; Fazekas, 2013).

3. ábra. Szigmoid függvény ábrázolva



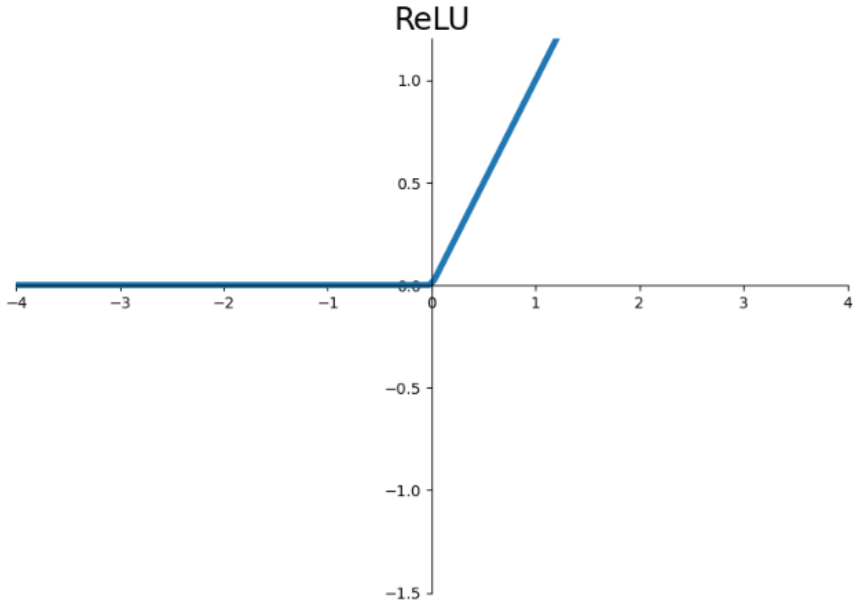
Forrás: saját szerkesztés

Egy másik aktivációs függvény, ami a kutatásom szempontjából releváns a ReLU (Rectified Linear Unit). A ReLU egy gyakran használt aktivációs függvény a neurális hálózatokban, mivel egyszerűsége ellenére jól működik. Bár nemlineáris transzformációt

végez, közel lineáris marad, hiszen két lineáris szakaszból áll, így megőrzi a lineáris modellek könnyű optimalizálhatóságát és jó általánosító képességeit. Mint ahogy a számítástechnika világában összetett rendszereket egyszerű elemekből építünk fel, a ReLU is egy alapvető építőelem a neurális hálózatokban (Goodfellow et al., 2016).

A ReLu aktivációs függvény képlete: $g(z) = max\{0, z\}$

4. ábra. A ReLu aktivációs függvény ábrázolva



Forrás: saját szerkesztés

2.2.3. Költségfüggvények

A gépi tanulás alapvető alkotóelemei a veszteség, vagy más néven költségfüggvények. Ezek segítségével tudjuk megmérni a modell pontosságát úgy, hogy a költségfüggvény megmutatja a modell jelenlegi predikciója és az elvárt eredmény közötti különbséget, vagyis, hogy mennyire áll távol a modell céljának elérésétől. Minden egyes predikció során a modell egy költségértéket számol ki, amely segítségével irányítani tudja a tanulás folyamatát. Minél kisebb ez a költség, annál közelebb állnak a becsült értékek a célértékhez, a modell tanításának célja tehát a költségfüggvény minimalizálása. Például egy lineáris regresszió esetében a költségfüggvény gyakran az átlagos négyzetes hiba (mean squared error, MSE), amely az eltérések négyzetének átlagát, vagyis a becsült és a valós értékek közötti különbségek négyzetének átlagát méri. Ez egy egyszerű, de hatékony eszköz a modell teljesítményének mérésére (Goodfellow et al., 2016).

A neurális hálózatok esetében a költségfüggvények még nagyobb jelentőséggel bírnak, mert ezek segítik a hálózat paramétereinek (súlyok és torzítások) finomítását. A hálózat minden rétege különböző szintű absztrakciókat tanul, a költségfüggvény pedig visszacsatolást ad arról, hogy ezek az absztrakciók milyen szinten járulnak hozzá a végső predikcióhoz.

Az egyik leggyakrabban használt költségfüggvény a keresztentrópia (cross-entropy**)**, amely különösen osztályozási feladatoknál népszerű. A kategorikus keresztentrópia két diszkrét valószínűségeloszlás közötti különbséget méri, és nagyon jól működik olyan feladatoknál, ahol a kimenet valószínűségi eloszlásokat ad vissza (például softmax rétegek után). A keresztentrópia annál alacsonyabb értéket ad, minél jobban hasonlít a modell által becsült valószínűségeloszlás a címkék által definiáltakhoz. A modellt ezzel kényszeríti arra, hogy magas valószínűséggel rendeljen a helyes osztályhoz egyedeket és alacsony valószínűséggel osztályozzon tévesen egyedeket. (Goodfellow et al., 2016). Ebből következik, hogy a modell célja a költség csökkentése lesz, mivel ez a modell jobb teljesítését vonja magával.

2.2.4. Optimalizálás

A mélytanulási algoritmusok többsége valamilyen optimalizálást végez. Az optimalizálás egy olyan feladatot jelent, amely során egy függvényt szeretnénk minimalizálni vagy maximalizálni azáltal, hogy módosítjuk a bemenetet (x). Ha valamit maximalizálni akarunk, azt átalakíthatjuk minimalizálási problémává úgy, hogy a maximalizálandó függvény negatívját minimalizáljuk. Ez azért van, mert a maximalizálás matematikailag megegyezik a minimalizálással, vagyis $max(f(x)) = min(-f(x))$. A legtöbb optimalizálási problémát általában valamilyen függvény minimalizálásaként fogalmazzuk meg. A függvényt, amit minimalizálni szeretnénk céfüggvénynek hívjuk. Ha minimalizáljuk, akkor gyakran költségfüggvénynek (cost function), veszteségfüggvénynek (loss function) vagy hibafüggvénynek (error function) nevezzük. Sok módszer létezik a minimalizálására, de a probléma komplexitása miatt a iteratív algoritmusokkal történik az optimalizáció (Goodfellow et al., 2016).

Gradiens Ereszkedés

Tegyük fel, hogy van egy $y = f(x)$ függvény, ahol x és y is valós szám. Ennek a függvénynek a deriváltját $f'(x)$-szel vagy dy/dx-szel jelöljük. A derivált $f'(x)$ megadja a függvény meredekségét x-nél, vagyis azt, hogy hogyan változik a kimenet egy kis változás

hatására a bemenetben: $f(x+\epsilon) \approx f(x) + \epsilon\, f'(x)$. A derivált így alkalmazható a függvény minimalizálásához, mivel megmutatja, hogy hogyan változtassuk x-et ahhoz, hogy egy kis javulást érjünk el y-ban (Goodfellow et al., 2016).

Ahhoz, hogy a függvényt minimalizáljuk, egy frissítési szabályt alkalmazunk, amely megadja, hogyan változtassuk x-et úgy, hogy csökkentsük a kimenetet. Ez a frissítési szabály a gradiens ereszkedés alapja

Frissítés szabálya:

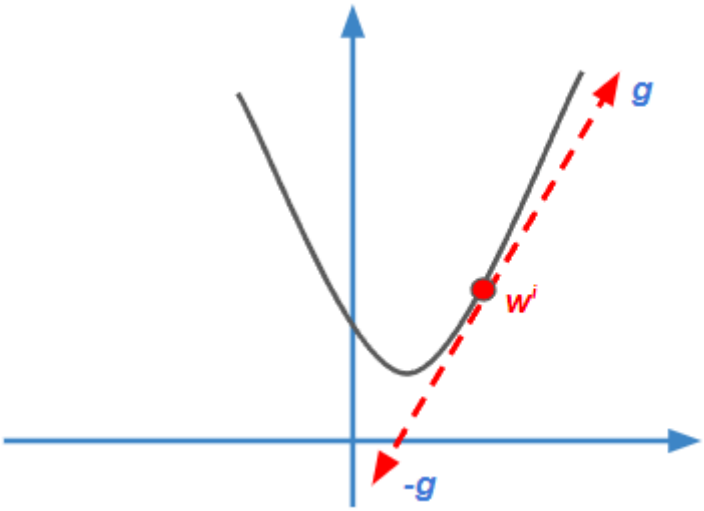
$$x_{i+1} = x_i - \eta \nabla f(x_i)$$

Magyarázat:

- x_i az i-edik lépésben vett hely
- $\nabla f(x_i)$: a függvény $f(x)$ gradiense az x_i-nél
- x_{i+1}: az új érték, amit a következő iterációban fogunk használni
- η: a tanulási ráta, amely meghatározza, mekkora lépést teszünk a gradiens mentén.

Ez a technika, amely a derivált segítségével iteratív lépésekben csökkenti a függvény értékét, a gradiens ereszkedés. A gradiens ereszkedés akkor fejeződik be, ha eléri az előre definiált lépésszámot, vagy ha a gradiens értéke egy előre meghatározott határértéket átlépve megközelíti a nullát, vagyis elér egy lokális minimum helyet (Goodfellow et al., 2016; Fazekas, 2013).

5. ábra. A képen a gradiens ereszkedés illusztrációja szerepel



Forrás: saját szerkesztés (Wu, 2017 felhasználásával)

Amikor $f'(x) = 0$, a derivált nem ad információt arról, hogy melyik irányba kellene haladni. Az ilyen pontokat kritikus pontoknak nevezzük. A lokális minimum egy olyan pont,

ahol $f(x)$ értéke kisebb, mint minden szomszédos pontban, tehát $f(x)$-et már nem lehet tovább csökkenteni végtelenül kicsi lépésekkel. Lokális maximumról beszélünk, amikor $f(x)$ értéke nagyobb, mint minden szomszédos pontban. Ezeken kívül még beszélhetünk nyeregpontról, amelyek olyan kritikus pontok, amelyek se nem maximumok, se nem minimumok. Az a pont, ahol $f(x)$ a lehető legkisebb értéket éri el, globális minimumnak számít. Az is előfordulhat, hogy vannak olyan lokális minimumok, amelyek nem globálisan optimálisak (Goodfellow et al., 2016).

A mélytanulás során általában olyan függvényeket optimalizálunk, amelyeknek több, nem optimális lokális minimuma is lehet, illetve nyeregpontokkal is rendelkezhetnek. Mindez megnehezíti az optimalizálást, különösen akkor, ha a függvény bemenete többdimenziós. Ezért általában kielégítő megoldásnak számít, ha találunk egy olyan értéket, ami nagyon alacsony, akkor is, ha az nem feltétlenül globálisan minimális (Goodfellow et al., 2016).

Sztochasztikus Gradiens Ereszkedés

A mélytanulás egyik kulcsfontosságú eleme a Sztochasztikus Gradiens Ereszkedés (stochastic gradient descent / SGD). A nagy méretű tanító adathalmazok feldolgozása nagyon költséges lehet, de az SGD lehetővé teszi, hogy egyszerre csak egy kis részhalmazt (köteget) használjunk a gradiens kiszámításához. Ennek alkalmazásával minden lépésnél véletlenszerűen választunk ki néhány példát az adathalmazból, és ezek alapján módosítjuk a modell súlyait (Goodfellow et al., 2016).

Az SGD egyik előnye, hogy függetlenül a tanító adatok méretétől, a modell frissítéseinek számítási költsége stabil marad. Az algoritmus hatékonysága miatt az SGD-t nemcsak a mélytanulásban, hanem más, nagy adatbázisokkal dolgozó modellekben is alkalmazzák (Goodfellow et al., 2016).

Számtalan manapság használt optimalizációs eljárás valamilyen SGD variáns. Ilyen például az AdaGrad, amely a gradiens korábbi értékei alapján állítja paraméterenként a tanulási ráta nagyságát, vagy az RMSProp amely szintén a tanulási ráta nagyságát finomhangolja, ez azonban a gradiens értékek négyzetének mozgóátlaga alapján végzi a frissítést. Az Adam a gradiens ereszkedés egy olyan fajtája, amely kimbinálja az AdaGrad és az RMSProp működését, ennek köszönhetően ez az egyik leggyakrabban használt optimalizációs algoritmus a mélytanulásban.

A CNN modell paramétereinek optimalizálása a veszteség minimalizálására irányul, azaz arra törekszik, hogy a modell becslései minél jobban egyezzenek az elvárt eredményekkel. Tegyük fel, hogy van egy tanítópont példa x1, amely alapján a paramétereket tanítjuk. A tanulási folyamat során a CNN hálózat két irányban fut. Először egy előrecsatolás során megkapja az előrejelzését a jelenlegi paraméterekkel, majd összehasonlítja azt a célcímkével, és kiszámolja a veszteséget (Wu, 2017).

A veszteség ad iránymutatást a modell paramétereinek frissítéséhez, amit az SGD szabálya szerint végzünk:

Paraméterek frissítése: $x_{t+1} = x_t - \eta \cdot \nabla_x J(x)$

Ahol:

- x_t : az aktuális állapot vagy iteráció értéke, amelyet optimalizálunk
- $\nabla_x J(x)$: a költségfüggvény teljes gradiense az összes minta alapján
- η: tanulási ráta
- $\nabla_x J(x)$: a gradiens $J(x)$-re nézve, az x pontban számítva

Ahol η a tanulási ráta. A gradiens azt mutatja meg, hogy hogyan változik a veszteség a súlyok változásával, és a gradiens irányával ellentétesen frissítjük a súlyokat a veszteség minimalizálása érdekében (Wu, 2017).

Túl nagy lépés esetén a negatív gradiens irányába a veszteség nőhet, ezért a paramétereket csak kis lépésekben változtatjuk, amit a tanulási ráta η szabályoz ($\eta > 0$ általában nagyon kis szám, például $\eta = 0{,}001$). A tanulási ráta megválasztásánál tehát figyelni kell, hogy ne legyen se túl magas, se túl alacsony. A paraméterek egy frissítése után a veszteség csökkenhet az adott tanító példán, de más példák vesztesége növekedhet ettől függetlenül. Ezért az összes tanító példát használni kell a paraméterek frissítésekhez. Ha ez megtörtént elmondhatjuk, hogy modellünk sikeresen feldolgozott egy teljes időszakot (epoch). Az epoch-ok ismétlésével általában a veszteség csökken, amíg a rendszer túl nem tanulja a tanító adatokat (Wu, 2017).

Az SGD során kis kötegeket (mini-batch) használunk, nem pedig az összes példát egyszerre, mert a teljes köteg feldolgozása túl költséges lenne nagy adathalmazok esetén. A mini-batch tipikusan 32 vagy 64 példából áll, ez az eljárás például a konvolúciós neurális hálók (CNN) paramétereinek tanításának fő módszere (Wu, 2017).

Hibavisszaterjesztés (Backpropagation)

A hibavisszaterjesztés a gradiens alapú optimalizálás során segít kiszámítani, hogy a neurális háló súlyainak milyen irányban kell változniuk a veszteség csökkentése érdekében. Lehetővé teszi, hogy az aktuális réteg adatait visszaterjessze az előző rétegeknek, így minden réteg folyamatosan módosítani tudja a súlyait. A cél, hogy minden réteg paramétereit úgy finomhangolja, hogy csökkentse a veszteségfüggvényt, és javítsa a hálózat predikcióinak pontosságát. A rétegekben szereplő gradiens a láncszabály segítségével van kiszámítva, így a tanulási folyamat során a paraméterek frissítése fokozatosan történik, ami a modell hatékony tanulását teszi lehetővé (Wu, 2017).

A hibavisszaterjesztés (backpropagation) folyamatában minden réteghez kétféle gradienst (parciális deriváltat) számítunk ki: az egyik a réteg súlyaihoz (w_i), a másik a bemenethez (x_i) kapcsolódik. A réteg paramétereinek frissítéséhez a z függvény súlyokra vett deriváltját használjuk, míg a bemenetre számított derivált lehetővé teszi a hibák továbbítását az előző rétegek felé. Ez azt jelenti, hogy a bemeneti deriváltak ($\frac{\partial z}{\partial x_i}$) továbbítják a hiba információkat az előző réteghez, így rétegenként haladva javíthatjuk a modell teljesítményét. Minden rétegben, miután a következő réteg gradienseit már kiszámítottuk, a láncszabály segítségével kiszámítjuk az adott réteg súlyainak és bemeneteinek gradienseit. Ezek a gradiensek a hibafüggvény csökkentéséhez szükséges irányt mutatják. Az egyes mátrix-átalakítások és transzponálások megkönnyítik a számításokat, így a hibák visszaterjesztése hatékonyan történik (Wu, 2017).

Ez a rétegenkénti hibavisszaterjesztés az egyik legfontosabb mechanizmus, amely lehetővé teszi, hogy a konvolúciós neurális hálózatok (CNN-ek) tanuljanak, mivel így minden réteg pontos visszajelzést kap arról, hogy mennyiben felelős a teljes hibáért, és ez alapján hogyan módosítsa az adott paramétereit, hogy csökkentse a hibát és javítsa az eredményt (Wu, 2017).

Hiperparaméterek

A legtöbb gépi tanulási algoritmus több állítható paraméterrel rendelkezik, amelyekkel szabályozhatjuk, optimalizálhatjuk az algoritmusunk működését. Ezeket a változókat hiperparamétereknek nevezzük. A hiperparaméterek értékeit nem maga a tanulási algoritmus módosítja, nincs egy meghatározott matematikai képlet, amely megadja értéküket, hanem általunk előre meg kell adni őket. (Goodfellow et al., 2016). Ilyen hiperparaméter például a

korábban is említett tanulási ráta, klaszterezési algoritmusokban a klaszterek száma, vagy az sztochasztikus gradiens ereszkedésnél használt köteg nagysága.

2.2.5. Konvolúciós Neurális Hálózatok (CNN)

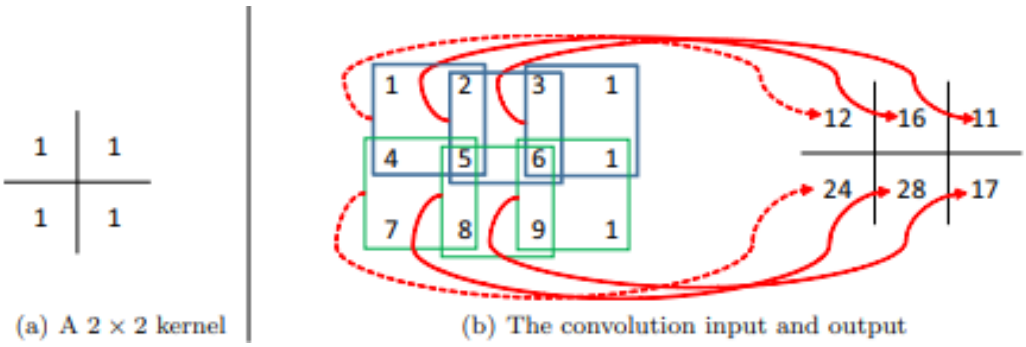
A Konvolúciós Neurális Hálózatok a mesterséges neurális hálózatok egy olyan speciális változata, amely különösen hatékony a rácsos felépítésű adatstruktúrák, például képek feldolgozásában. Az utóbbi években nagy sikereket értek el a nagyméretű képek és videók felismerésében. Ez a fejlődés elsősorban a nyilvánosan hozzáférhető nagy képi adatbázisoknak, mint az ImageNet, valamint a nagy teljesítményű számítási eszközöknek, például a videókártyák fejlődésének köszönhető (Simonyan & Zisserman, 2015).

Az elmúlt években nagy hírnévre tettek szert a mesterséges látás feladataiban, mint például az objektum detektálás, képfelismerés és szegmentáció. A hagyományos, teljesen összekapcsolt rétegeket használó hálózatokkal ellentétben a CNN-ek konvolúciós rétegeket használnak, amely segítségével képesek hierarchikus térbeli jellemzőket kivonni a bemeneti adatokból (Krizhevsky, Sutskever, & Hinton, 2012).

A CNN-ek felépítése

A konvolúció működése során a bemeneti képen vagy adathalmazon egy „kernelnek” nevezett kis mátrixot mozgatunk. Ez a kernel lokálisan, azaz a bemenet kisebb részletein számol ki értékeket. A bemenet és a kernel elemeit szorozza össze, majd az így kapott eredményeket összegzi, és egy új értéket állít elő, amelyet az úgynevezett jellemzőtérkép (feature map) tartalmaz. A képen egy 2 x 2-es méretű kernel és egy 4 x 3 méretű jelfüggvény konvolúciója látható.

6. ábra. 2 x 2-es kernel működése



Forrás: Wu, 2017.

A konvolúciós művelet tehát egy képet vagy adatot régiónként, apró részleteiben dolgoz fel, különböző jellemzőket felismerve, például vízszintes és függőleges éleket, textúrákat vagy alakzatokat. Ezek a jellemzők egyre összetettebb mintázatokat alkotnak a hálózat mélyebb rétegeiben. A konvolúciós réteg előnye, hogy ugyanaz a kernel minden helyen alkalmazható az adaton, így kevesebb paraméter szükséges a működéséhez, csökkentve ezzel a számításhoz elhasznált erőforrások mennyiségét (Wu, 2017) .

Az alábbi képsorozaton látható, ahogyan a bal oldali bemeneti képből kinyeri egy konvolúciós réteg a különböző jellemzőket, például a vízszintes és a függőleges éleket. A második képen egy olyan konvolúciós kernel szerepel, amely a vízszintes éleket, a harmadikon pedig egy olyan, amely a függőleges éleket emeli ki.

7. ábra. Példa a konvolúciós réteg működésére

Forrás: Saját szerkesztés

A konvolúció során gyakran alkalmaznak kitöltést (padding), hogy a bemenet és a kimenet mérete azonos maradjon, valamint lépésközt (stride) is használnak, hogy a kernel ne minden egyes pixelnél végezze el a számításokat, hanem meghatározott lépésközökkel lépkedjen végig a bemeneti adaton, ezzel kisebb méretű kimenetet eredményezve (Wu, 2017).

Egy tipikus CNN több komponensből épül fel: konvolúciós rétegekből, pooling rétegekből és teljesen összekapcsolt rétegekből.

A konvolúciós réteg (convolutional layer) alapvetően kép, vagy más rácsos struktúrájú adat feldolgozására szolgál. Ez a CNN alapvető építőeleme, a nevét is erről a rétegről kapta. Ha már 1 darab konvolúciós réteg szerepel a neurális hálónkban, akkor az egy konvolúciós neurális háló. A konvolúciós rétegben a konvolúciós kernel tanítható, a réteg kimenete pedig a bemeneti kép és a kernel konvolúciójával áll elő (Goodfellow et al., 2016).

A konvolúciós rétegeket gyakran pooling rétegek követik, amelyek csökkentik a jellemzőképek térbeli dimenzióit, ezzel növelve a hálózat hatékonyságát, a fontos információk megtartásával. A pooling általában max-pooling formájában megy végre, ahol a jellemzők egy-egy régiójának maximumát veszik, ezzel kiemelve a legfontosabb jellemzőket, elkülönítve a kevésbé informatív jellemzőktől. (Zeiler & Fergus, 2014) A pooling réteg bevezet egy eltolási invarianciát, tehát a bemenet kisebb mértékű eltolása esetén az output nem változik. Ez különösen hasznos, ha egy jellemző megléte fontosabb, mint annak pontos helye, például egy arc felismerésekor mindegy, hogy pixelpontosan hol helyezkednek el a szemek, elég, ha tudjuk, hogy van egy szem az arc bal oldalán, illetve egy a jobb oldalán (Goodfellow et al., 2016).

A több konvolúciós és pooling réteg után a kimenet általában egy vagy több teljesen összekapcsolt rétegen keresztül halad tovább, amelyek átalakítják a magas szinten absztrahált jellemzőket osztályozási feladatok esetén kimeneti kategóriákká. Ezek a rétegek hasonlóan működnek, mint a hagyományos neurális hálók teljesen összekapcsolt rétegei. (He, Zhang, Ren, & Sun, 2016) Ezek a rétegek általában a neurális hálózatok végén találhatók, és ezek felelősek a végső kimenetek előállításáért. A konvolúciós neurális hálózatokban általában a konvolúciós és pooling rétegek után következnek. Ezek a rétegek alakítják át az adatok kétdimenziós térbeli struktúráját egy egydimenziós vektorrá, amit felhasználnak olyan feladatokhoz, mint a klasszifikáció.

A teljesen összekapcsolt rétegek súlyait és torzításait a tanulási folyamat során alakítják ki, így a réteg az adott problémához igazodik. Az utolsó teljesen összekapcsolt réteg neuronjainak száma általában megegyezik a klasszifikációs probléma kimeneti osztályainak számával. Például egy 10 osztályos klasszifikációs problémánál az utolsó réteg softmax aktivációs függvénnyel rendelkezne, kimenetében 10 neuron lenne, mindegyik egy-egy osztályhoz tartozva. Az utolsó rétegben mindegyik neuron nyers kimenetét egy softmax aktivációs függvény alakítja át valószínűségi vektorrá (Rastogi, 2023).

Összegezve tehát a konvolúciós neurális hálók működését elmondható, hogy a képfelismerés, osztályozás folyamata egy bemeneti képpel kezdődik, amelyet számos konvolúciós, aktivációs és pooling rétegen keresztül megy. Minden réteg különféle jellemzőket emel ki és tanul meg, az első rétegek az alacsony szintű részleteket, például éleket és textúrákat ragadják meg, a mélyebb rétegek pedig absztraktabb koncepciókat, például adott objektumok részeit ismerik fel. Mire a jel eléri a teljesen összekapcsolt réteget, egy rendkívül absztrahált reprezentációt tartalmaz, amely lehetővé teszi a hálózat számára a végső predikció elvégzését.

2.3. Generatív modellek

A generatív modellek alatt az informatika világában általában olyan neurális hálókat értünk, melyek képes felismerni és megtanulni az adatok mögött rejlő összefüggéseket, ismétlődő tulajdonságokat, mintákat, eloszlásokat. Ezeket az információkat és kapcsolatokat ismerve képes ezek használatával új, hasonló adatokat elállítani. Egyszerűbben fogalmazva, olyan, mintha megtanítanánk a számítógépet „álmodni", vagyis eddigi tapasztalatai felhasználásával olyan új tartalmat létrehozni, amely hasonló tulajdonságokkal rendelkezik, mint az eredeti adathalmaz.

Ezek a modellek arra törekednek, hogy megtanulják az adatok és hozzájuk kapcsolódó címkék együttes valószínűségi eloszlását ($p(x, y)$). Miután a modell megtanulta ezt az eloszlást, képes lesz új adatokat generálni az eloszlásból való mintavételezés segítségével.

Ezzel szemben a diszkriminatív modellek csak a $p(y|x)$ feltételes valószínűségi eloszlást tanulják meg, amely az adott bemeneti adatokhoz rendeli hozzá a címkéket. Míg a diszkriminatív modellek elsősorban osztályozási feladatokra és döntéshozatalra használatosak, a generatív modellekkel új adatokat tudunk előállítani (Murphy, 2012).

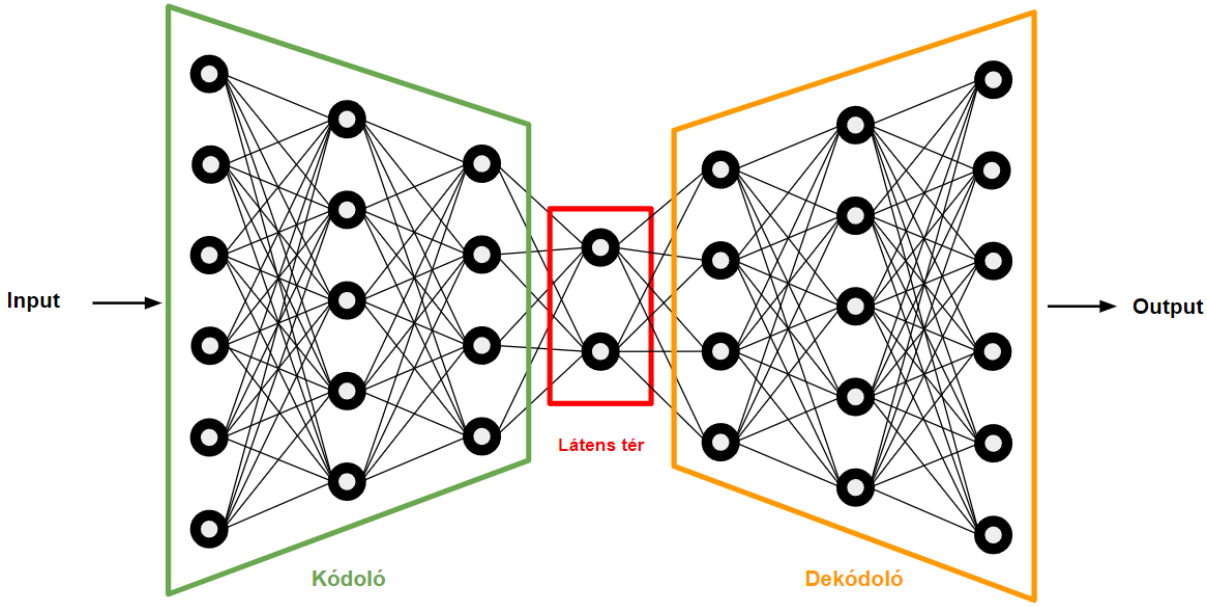
2.3.1. Generatív Modellek Típusai

A generatív modellek is több típusba sorolhatók, melyek mind más és más módszerekkel tanulják meg az adatok eloszlását, és más módszert használnak újak generálásához.

Autoencoder (AE)

Az önkódoló egy olyan felügyelet nélküli algoritmus, amelynek célja a bemeneti adatok hatékony reprezentációinak megtanulása. Három fő komponensből épül fel: van a kódoló, amely az adatokat egy tömörített reprezentációba alakítja át, a látens tér, ami hordozza az adatok legfontosabb jellemzőit, és van a dekódoló, amely ebből a reprezentációból próbálja meg visszaállítani az eredeti adatot (Goodfellow et al., 2016).

8. ábra. Az önkódoló (AE) általános felépítése



Forrás: saját szerkesztés

A Variációs autoencoder az alap autoencoder kibővített változata. A VAE egy valószínűségi struktúrát alkalmaz a látens térben, amely segítségével nem csak egyetlen determinisztikus kódot tud létrehozni, hanem a bemeneti adatok eloszlásából tud mintavételezni. Így a generált kimenet nem pusztán egy rekonstrukció, hanem új, az eredeti adatokkal hasonló minták létrehozására is alkalmas. Ezzel a VAE egyesíti a mélytanulás technikáit és a valószínűségi gráfok elméletét, így különösen hasznos eszközzé válik olyan összetett adatok, mint például képek, hangok vagy más struktúrált adatok kezelésében és előállításában (Kingma & Welling, 2014).

Boltzmann-gép

A Boltzmann-gép egy olyan neurális hálózat, amely valószínűségi modelleket használ az adatok közötti komplex kapcsolatok modellezésére. Ezek a hálózatok sztochasztikus elven működnek, ahol a neuronok közötti energiahatások határozzák meg a rendszer viselkedését. A hálózat célja, hogy egy adott adathalmaz eloszlását modellezze úgy, hogy minimalizálja az elhasznált “energiát”.

A Boltzmann-gépek különösen hasznosak abban, hogy többrétegű reprezentációkat hozzanak létre az adatokból, és ezáltal mélyebb összefüggéseket tárjanak fel. Habár ezek a hálózatok nagy potenciállal bírnak, a tanításuk és paramétereik optimalizálása a magas dimenziószámú látens tér miatt bonyolult. Hinton (2002) kontrasztív divergencia módszere ezt

a problémát hivatott kezelni, mivel egyszerűsíti a számításokat, és jelentősen felgyorsítja a tanítási folyamatot. Hinton által kidolgozott architektúra lényege, hogy a tanulási folyamat során a modell először az adatokat használva generál egy kezdeti eloszlást, majd ezt összehasonlítja a saját mintavételezéssel létrehozott eloszlással. Az eltérést, azaz a divergenciát minimalizálja a hálózat, hogy az adatokhoz egyre közelebb álló eloszlást tudjon generálni. Ennek köszönhetően a Boltzmann-gépek képesek olyan komplex struktúrákat felismerni és modellezni, amelyek más típusú neurális hálózatokkal nehezen megvalósíthatók (Hinton, 2002).

A Boltzmann-gépek sztochasztikus elven működő generatív neurális hálózatok, amelyek képesek belső reprezentációk tanulmányozására, mely segítségével képesek megbirkózni nehezen megoldható kombinatorikai problémákkal is. A korai optimalizációs technikák, amelyeket a neurális hálózatokban alkalmaztak, ezekre a gépekre épültek. A Boltzmann-gép koncepcióját Geoffrey Hinton és Terry Sejnowski dolgozta ki 1985-ben. A Boltzmann-gépek kétféle csomópontból (node) épül fel, rejtett és látható csomópontokból. Nincs kimeneti pontjuk, mivel nem determinisztikus elven működnek. A Boltzmann-gépek egy speciális változata, a Korlátozott Boltzmann-gép, amely egy két rétegből álló neurális hálózat. (Patel & Rama, 2020).

2.3.2. Generatív Ellenséges Hálózatok (GAN)

A Generatív Ellenséges Hálózatok (GAN-ok) a gépi tanulás egyik leginnovatívabb és legelterjedtebb módszerévé váltak. Működésüket először 2014-ben, Ian Goodfellow és társai bemutatták őket az "Generative Adversarial Nets" című tanulmányukban (Goodfellow et al., 2014). A GAN alapvető működése két hálózat, egy generátor és egy diszkriminátor közötti versengésre épül, ahol a generátor célja, hogy olyan szintetikus adatokat generáljon, amelyek a lehető legjobban hasonlítanak a valós adatokra, míg a diszkriminátor feladata az, hogy megkülönböztesse a valós adatokat a generáltaktól. Ez a kompetetív, "minimax" típusú interakció a hálózatok között lehetővé teszi a generátor számára, hogy folyamatosan javuljon, és egyre hitelesebb adatokat hozzon létre (Goodfellow et al., 2014).

2.3.2.1. A GAN alapelvei

A GAN-ok működésének alapja az a feltételezés, hogy két mesterséges neurális hálózat egymás elleni versengéséből képes tanulni és fejlődni. A generátor egy zajból indul ki, és

megpróbál valósághű adatokat generálni, amelyek hasonlítanak a tanító adatokhoz. Ezzel szemben a diszkriminátor egy bináris osztályozó szerepét tölti be, amely a generált adatokat és a valódi adatokat próbálja megkülönböztetni. A generátor célja, hogy megtévessze a diszkriminátort, míg a diszkriminátor célja, hogy pontosan felismerje, mely adatok valósak és melyek hamisak (Goodfellow et al., 2014).

A diszkriminátort arra képezzük, hogy maximalizálja annak valószínűségét, hogy helyesen osztályozza mind a tanító, mind a generátor által létrehozott elemeket. Ezzel párhuzamosan a generátor tanításának a célja, hogy minimalizálja a $\log(1 - D(G(z)))$ értékét, vagyis arra, hogy a generált minták minél jobban "átverjék" a diszkriminátort. Matematikailag ez a probléma a következőképpen írható le:

$$\min_G \max_D V(D,G) = \mathbb{E}_{X \sim P_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim P_z(z)}[\log(1 - D(G(z)))]$$

Magyarázat:

- $P_{data}(x)$: a valós tanító adatok eloszlása
- $D(x)$: annak a valószínűsége, hogy a diszkriminátor helyesen azonosítja a valós adatot x-ként
- $P_z(z)$: a nem valós (generált) adatok eloszlása
- $D(G(z))$: annak a valószínűsége, hogy $G(z)$ a szintetikus adatok közé tartozik
- $G(z)$: a generátor által z bemenetből (zajból) előállított adat

A generátor célja tehát az, hogy olyan adatokat állítson elő, amelyek a diszkriminátor számára valódinak tűnnek, ezáltal csökkentve a képlet második felének értékét, miközben a diszkriminátor arra törekszik, hogy helyesen különbséget tegyen a valós és a hamis adatok között (Goodfellow et al., 2014).

2.3.2.2. A GAN-ok elméleti alapjai

A GAN-ok egyik legfontosabb előnye, hogy az adatgenerálás során explicit modelltanulás nélkül képesek új mintákat létrehozni. Ez a valószínűségelméleti megközelítés, ahol a generátor közvetett módon próbálja megközelíteni az adatok eloszlását, lehetővé teszi, hogy komplex, nagy dimenziójú adatokat, például képeket vagy hangokat is képes legyen generálni. Mivel a generátor nem kap közvetlen visszajelzést a hibáiról, hanem csak a

diszkriminátor válasza alapján tanul, a GAN-ok tanítása megannyi kihívást rejteget magában (Goodfellow et al., 2014).

Az egyik fő probléma a modell összeomlása. Ez akkor jelentkezik, amikor a generátor kevés fajta egymástól különböző adatmintát hoz létre, ahelyett, hogy az adat teljes eloszlását lefedné. Ebben az esetben a generált minták különbözősége lecsökken, és a generátor megtanul egyetlen, vagy néhány optimális megoldást produkálni, amit a diszkriminátor nehezen tud megkülönböztetni. Ennek elkerülésére számos technikát dolgoztak ki, beleértve a diszkriminátor és a generátor közötti finomhangolást az egyensúly fenntartásáért, valamint a különböző költségfüggvények alkalmazását (Goodfellow et al., 2014).

Bár a fenti egyenlet elméletben megfelelő a generátor tanításához, a gyakorlatban gyakran nem szolgáltat elégséges gradienst a generátor számára. A tanulás korai szakaszában, amikor a generátor még gyenge, a diszkriminátor könnyedén felismeri a generált mintákat, mivel azok jelentősen eltérnek a valós adatoktól. Ilyenkor a $log\ (1 - D(G(z))$ értéke telítődik, amely nehézségeket okoz a generátor fejlődésében. Ezen probléma megoldására a generátor megfelelő tanítása $log\ (D(G(z))$ maximalizálásával érhető el, ami erősebb gradienst biztosít a tanulás korai szakaszaiban, ezzel kisegítve a generátor tanulását (Goodfellow et al., 2014).

2.3.2.3. A GAN-ok továbbfejlesztése

A GAN-ok fejlődésével újabb modellek jelentek meg az évek folyamán, amelyek javítják a hálózatok teljesítményét és stabilitását egyaránt. Például a Deep Convolutional GAN (DCGAN) konvolúciós hálózatokat alkalmaz a GAN struktúrában, ezáltal javítva a generált képek minőségét, hogy azok élethűek és részletgazdagok legyenek (Radford et al., 2016).

A Deep Convolutional Generative Adversarial Network (DCGAN) a generatív ellenséges hálózatok (GAN-ok) egyik továbbfejlesztett változata, amely a hagyományos GAN struktúrát konvolúciós neurális hálózatokkal (CNN-ekkel) egészíti ki. A DCGAN egyik legnagyobb előnye, hogy képes megragadni a képi adatokban rejlő összefüggéseket, amelyet a teljesen összekapcsolt (fully connected) rétegek nem tudnának ilyen hatékonyan kezelni (Radford et al., 2016).

A DCGAN technikája nemcsak az élethű képek generálásában játszik kulcsszerepet, hanem lehetőséget ad arra is, hogy a generált képeket vizuálisan értékeljük. A CNN-ek képesek felismerni és újraalkotni a képadatok legfinomabb részleteit, így a DCGAN modellek

különösen alkalmasak nagy felbontású képek előállítására. Például a híres CelebA adatbázison való alkalmazásukkal a DCGAN képes volt olyan emberi arcok generálására, amelyek gyakran megtévesztően hasonlítanak a valós fotókhoz (Radford et al., 2016).

A DCGAN használatának egyik legjelentősebb területe a művészet és a szórakoztatóipar, ahol új műalkotásokat, festményeket, zenéket vagy akár videókat lehet generálni a segítségével. A technológia másik kiemelt felhasználási területe a képjavítás vagy a szuperfelbontású képalkotás, ahol az alacsony felbontású képeket a DCGAN segítségével feljavítják, hogy részletgazdagabbak és élesebbek legyenek. A divatiparban is alkalmazzák a technológiát, ahol új dizájnokat vagy ruhamintákat generálnak a modell segítségével, ami jelentős előny lehet a tervezési folyamatok gyorsítása során (Radford et al., 2016).

A Wasserstein GAN (WGAN) egy másik fontos előrelépés volt 2017-ben, amely a Kantorovich-Rubinstein metrikát alapul véve próbálta orvosolni a GAN-ok tanítási stabilitásának problémáit. A WGAN egy újfajta költség-függvényt vezetett be, amely a Föld távolság (Earth Mover's Distance) koncepcióján alapul, ezáltal a tanítás stabilabbá vált, és a mode collapse jelenség kisebb valószínűséggel következik be (Arjovsky, Chintala, & Bottou, 2017).

2.3.2.4. Alkalmazások és gyakorlati felhasználás

A GAN-ok alkalmazása sokrétű, különösen a képgenerálás és adat augmentáció területén terjedtek el. Olyan feladatokra képesek, mint például új, valósághű képek létrehozása, képjavítás (például alacsony felbontású képek feljavítása), vagy hiányos adatok pótlása. A GAN-ok hatékonysága miatt gyakran alkalmazzák őket olyan területeken, mint a számítógépes látás, a gyógyszerfejlesztés és az adatgenerálás (Goodfellow et al., 2014; Radford et al., 2016; Arjovsky et al., 2017).

A GAN-ok különböző változatai, mint például a Conditional GAN (cGAN), további irányított adatgenerálást tesznek lehetővé. A cGAN-oknál a generátornak és a diszkriminátornak nemcsak az eredeti bemenetre kell reagálnia, hanem további címkékre is, amelyek meghatározzák, hogy milyen típusú adatot kell létrehozni vagy felismerni (Goodfellow et al., 2014) .

2.3.3. Feltételes Generatív Ellenséges Hálózatok (cGAN)

A generatív ellenséges hálózatok előnye, hogy csak a gradiens visszaterjesztést használják a gradiensek kiszámításához, nincs szükség a tanulás során inferenciára. Egy nem kondicionált generatív modell esetén nem tudjuk befolyásolni a generált adatokat. Azonban, ha a modellt kiegészítjük és egyéb információval látjuk el, vagyis kondicionáljuk, lehetőség nyílik a generációs folyamat irányítására. Az ilyen kondicionálás történhet például osztálycímkék segítségével (Mirza & Osindero, 2014).

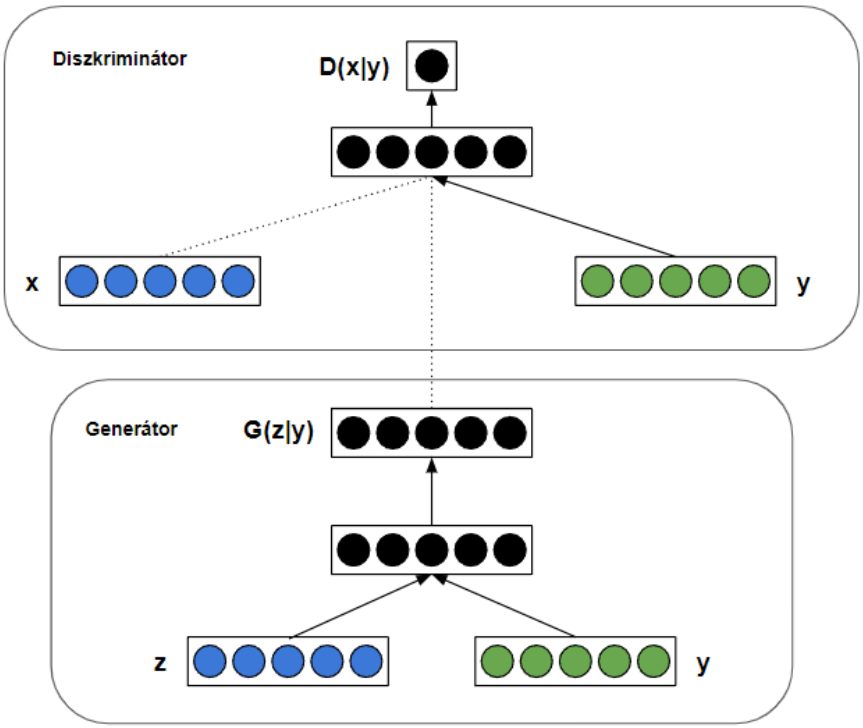
A generatív ellenséges hálózatok kibővíthetők egy kondicionált modellre, ha mind a generátor, mind a diszkriminátor egy extra információra, y-ra (label) van kondicionálva. Az y bármilyen hozzáadott információ lehet, ilyenek például az osztálycímkék. A kondicionálás megvalósításához az y-t további bemeneti rétegként kell betáplálni mind a generátorba, mind a diszkriminátorba. (Mirza & Osindero, 2014).

A generátorban a korábbi bemeneti zaj, $p(z)$, és az y kombinálódik egy közös rejtett reprezentációban. A diszkriminátorban az x, vagyis a generátor által generált generált adat, valamint az y együtt kerül feldolgozásra, hogy a diszkriminátor képes legyen megkülönböztetni a valódi és a generált adatokat a megadott címkék (y) figyelembevételével (Mirza & Osindero, 2014).

A feltételes generatív ellenséges hálózatok kétjátékos minimax játékának célfüggvénye a következő:

$$\min_G \max_D V(D,G) = \mathbb{E}_{X\sim P_{data}(x)}[\log D(x|y)] + \mathbb{E}_{z\sim P_z(z)}[\log(1 - D(G(z|y)))]$$

9. ábra. Az alábbi ábra egy egyszerű feltételes ellenséges hálózat szerkezetét illusztrálja



Forrás: saját szerkesztés (Mirza & Osindero (2014), Conditional generative adversarial nets című művét felhasználva)

2.3.4. Generatív modellek teljesítményértékelése

2.3.4.1. Inception Score (IS)

Az Inception Score-t a generált képek minőségének és változatosságának mérésére fejlesztették ki. Az IS kiszámításához is az Inception-v3 nevű mély neurális hálózat használatos, amelyet eredetileg az ImageNet adatbázison való osztályozási feladatokra terveztek. Az Inception-v3 egy RGB képekre optimalizált, 1000 osztályos osztályozó modell, a kimenete pedig egy valószínűségi vektor, amely minden osztályhoz valószínűséget rendel (Barratt & Sharma, 2018).

Az IS képlete: $IS = exp\left(E_{x\sim p_g} D_{KL}\big(p(y|x) \parallel p(y)\big)\right)$.

Magyarázat:

- $x \sim p_g$: a generált képek mintái,
- $p(y|x)$: a hálózat által adott valószínűségi eloszlás a kategóriákra,
- $p(y)$: az osztályeloszlás a generált képeken,

- D_{KL}: a Kullback-Leibler divergencia, amely a két eloszlás eltérését méri.

A magas IS érték jó minőségű képeket jelez, amelyek osztályspecifikusak és változatosak. Az alacsony IS érték gyenge minőséget, alacsony diverzitást vagy bizonytalan osztályozhatóságot jelez.

2.3.4.2. Fréchet Inception Distance (FID)

A FID az IS-hez hasonlóan az Inception-v3 modellt használja, de teljesen más megközelítéssel, nem a képek osztályozásával dolgozik, hanem a modell egy belső rétegéből származó jellemzővektorokat használ (ez egy sokdimenziós vektor, amely a kép magas szintű jellemzőit tartalmazza, például formák, textúrák, mintázatok). A FID a valós képek és a generált képek jellemzővektorainak eloszlását hasonlítja össze (átlaguk és kovariancia mátrixuk segítségével). Ez azt méri, hogy a generált képek jellemzői mennyire hasonlítanak a valós képek jellemzőire. Mivel a FID nem az osztályozásra, hanem a képek általános mintázataira és jellemzőire koncentrál, kevésbé érzékeny arra, hogy a képek pontosan illeszkedjenek az Inception-v3 által ismert kategóriákhoz (Heusel, 2017).

Az FID kiszámításához a generált és a valós adatokat átfuttattam az Inception-v3 modellen, ezáltal megkaptam a vizuálisan releváns jellemzőket. A FID kiszámításának képlete:

$$\text{FID} = d^2 = \left|\left|\mu_1 - \mu_2\right|\right|^2 + Tr\left(C_1 + C_2 - 2\sqrt{C_1 C_2}\right)$$

Magyarázat:

- $\mu_1 - \mu_2$: a valós és generált jellemzők átlagértékei.
- C_1, C_2: a valós és generált jellemzők kovarianciamátrixai.
- Tr: a lineáris algebra trace műveletét jelöli, például a mátrix főátlója mentén lévő elemek összegét.

Az alacsony FID érték azt jelzi, hogy a generált képek eloszlása közel áll a valós adatokéhoz, így a képek minősége és diverzitása magas. A 0 és 10 közötti FID értékek általában rendkívül jó generált képeket jelentenek, amelyek szinte megkülönböztethetetlenek a valós képektől. A 20 körüli értékek jó generálási teljesítményt jeleznek, az 50-nél nagyobb értékek pedig azt mutatják, hogy a generált képek jelentősen eltérnek a valósoktól, ami alacsony generálási minőségre utal (Heusel, 2017).

3. Képek elemzése cGAN modellel

3.1. Probléma bemutatása

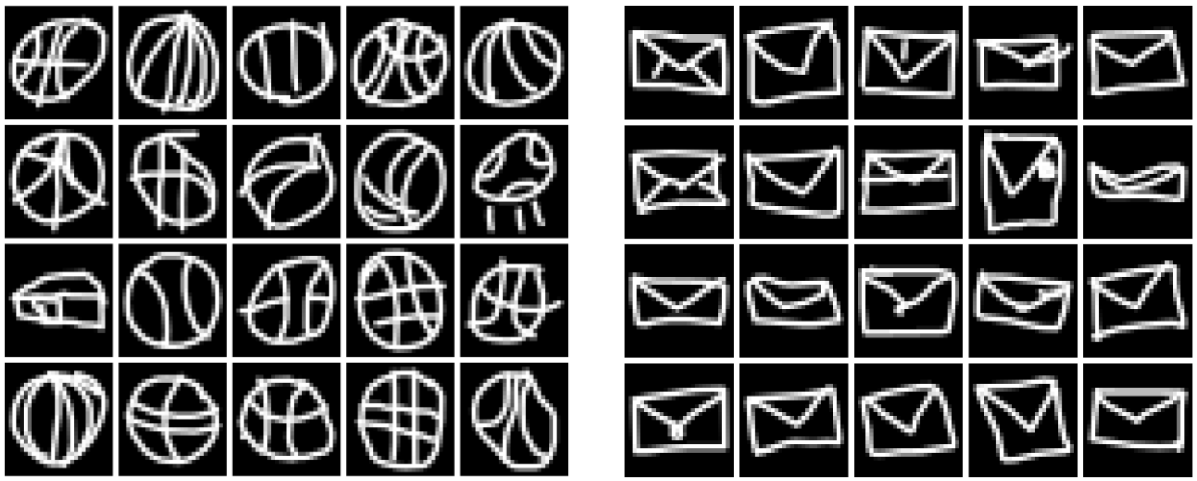
Az esettanulmányom célja megválaszolni a feltett hipotéziseimet, vagyis bebizonyítani, hogy lehetséges olyan osztályspecifikus képeket generálni egy feltételes generatív ellenséges hálózat (cGAN) segítségével, amelyek vizuális minőségükben, részletgazdagságukban és sokszínűségükben közel állnak a valós, ember által készített képekhez. A kutatás során a cGAN modell működését és teljesítményét vizsgálom, különös figyelmet fordítva a generált képek osztályspecifikus jellemzőire, valamint az általuk elért vizuális minőségre.

Ebben a fejezetben bemutatom a cGAN architektúráját és annak működési elvét, beleértve az alapvető építőelemeket, a generátor és a diszkriminátor hálózatokat, valamint azok együttműködését a tanulás során. Az alkalmazott technikák és algoritmusok részletes ismertetésével feltárom a modell tanításának folyamatait, beleértve a hiperparaméterek optimalizálását, az adatfeldolgozást és a tanulás stabilitásának fenntartásához szükséges megközelítéseket. A kutatás alapjául a Google által készített Quick, Draw! adathalmaz szolgált, amely egyedülálló lehetőséget biztosított egyszerű, kézzel készített rajzok tanulmányozására. A generatív és kiértékelő modelleket saját fejlesztésű kódokkal valósítottam meg, amely lehetővé tette a modell testreszabását és hatékony tanítását. A Quick, Draw! adathalmaz egy nyíltan elérhető, szabadon felhasználható adathalmaz, amelyet a Google hozott létre az emberi rajzolási minták elemzésére és modellezésére. Az adathalmaz több mint 50 millió kézzel készített rajzot tartalmaz 345 kategóriában, ezek lehetnek például geometriai alakzatok (kör, háromszög) vagy hétköznapi tárgyak (autó, kamera, labda) formájában.

A rajzok elérhetők több kiterjesztésben is, amelyeket a google előkészített és megtisztított. Az egyszerűsített rajzok „.ndjson”, „.bin” és „.npy” formátumban szerepelnek, én ezek közül a .npy formátum használata mellett döntöttem. A „.npy” formátum tömb alapú adatok tárolására szolgál. Minden rajz 28 x 28 pixeles bináris kép formájában van tárolva, ahol a képpontok intenzitása -1 (fehér) és 1 (fekete) között változik. Ez az adattípus a képek kompakt méretű tárolásának köszönhetően kiválóan alkalmas gépi tanulási algoritmusokkal végzett kísérletekre, mivel gyorsan és költséghatékonyan lehet kezelni, nem igényel sok erőforrást az ilyen típusú fájlok írása és olvasása.

A neurális hálók, főleg a generatív neurális hálók tanítása rendkívül sok erőforrást és időt igénylő folyamat. Annak ellenére, hogy a tanító adataimként szolgáló képek alacsony felbontásúak, ez a relatív kis méret is hatalmas mennyiségű erőforrást igényel. A számítási költségek és tanítási idő csökkentése érdekében a modellemmel egy időben egyszerre csak 2 osztályt dolgozok fel és generálok. A kutatás során a kosárlabda és a boríték osztályokat választottam elemzésem alapjául. A tanító adathalmazból vett példaképek osztályok szerint:

10. ábra. A tanító adatokból ábrázolt képek, osztályonként



Forrás: saját szerkesztés

3.2. Az adathalmaz előkészítése

Az adatok megfelelő formázása és előkészítése a generatív modellek egyik legfontosabb lépése. Az adatok hatékony betöltése és a memóriakihasználtság miatt egy olyan adatkészlet-implementációt használtam, amely nem csak lehetővé teszi a tanító adathalmazból a képek egyszerű betöltését és kezelését, de a megfelelő osztálycímkék hozzárendelését is megvalósítja. A képek dimenzióját a modell úgy alakítja át, hogy minden egyes képet külön tárol egy mátrixként. Betölti a képadatokat a különböző osztályokhoz a kiválasztott tanító fájlokból, majd normalizja azokat. A képadatok normalizálása több szempontból is fontos az neurális hálózatok tanításában.

A normalizált adatok segítenek az optimalizálónak gyorsabban és stabilabban konvergálni, mivel a bemenetek értékei kisebb tartományban mozognak. Ha az adatok értékei túl nagyok vagy eltérő mértékűek, az optimalizálási folyamat instabillá válhat, és nehezebb lesz a megfelelő minimumot megtalálni. A normalizálás az aktivációs függvények (pl. tanh, sigmoid) optimális működéséhez is hozzájárul. Például a tanh függvény értékei [-1, 1] között

mozognak, így a bemenetek ugyanezen tartományba hozása hatékonyabb tanulást eredményez. Ha a bemenetek eltérő skálán vannak, az aktivációs függvény kimenete torzulhat, és a gradiens alapú tanulás nehezebbé válik.

Ha az adatok értékei túl nagyok, az neurális háló súlyainak frissítése során a gradiens értékek hajlamosak lehetnek a hirtelen növekedésre, ami a háló instabilitását okozhatja. Ha az értékek túl kicsik, az az aktivációs függvény szaturációjához vezethet, ami a gradiens eltűnését eredményezheti. Mindkettő jelentősen hátráltatja a tanulási folyamatot. A hálózat rétegeinek kezdeti súlyeloszlását az adatok skálája is befolyásolja. Ha a bemeneti adatok normalizáltak, a kezdeti súlyeloszlás jobban illeszkedik az adatokhoz, ami hatékonyabb tanulást eredményez.

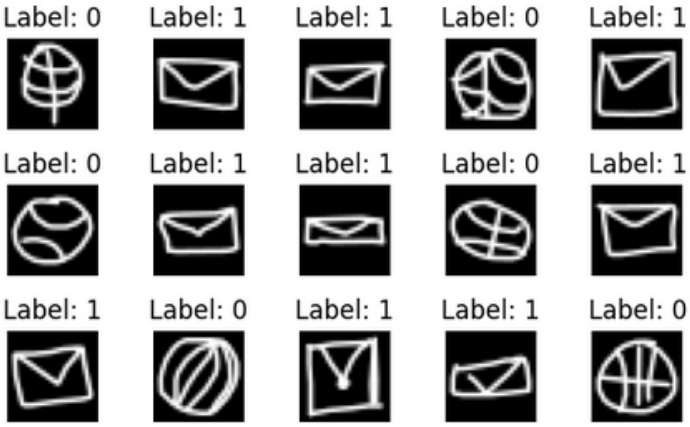
A különböző forrásokból származó képadatok jellemzői (pl. fényesség, kontraszt) eltérhetnek. Bár az én adataimmal ez nem fordulhat elő, mivel ilyen szempontból kivétel nélkül egységesek, a normalizálás biztosítja, hogy a hálózat minden adatot egységes módon dolgozzon fel, ami jobb általánosító képességet eredményez.

A hálózat ezt követően minden osztályhoz egyedi numerikus címkét rendel. A 0. osztályba fog tartozni az elsőnek beolvasott fájl, vagyis a kosárlabda fájl, az 1. osztályba pedig a boríték fájl. Az összes címkét és a képeket egyetlen tömbbe konkatenálja, amely segítségével egységesen tudja kezelni a címkéket a képekkel. A képek dimenzióját egy csatorna dimenzióval bővíti, mivel a modell konvolúciós rétegei ezt megkövetelik. Ez a dimenzió a szürkeárnyalatos képek csatornáját képviseli.

Ezután az adatokat kötegekre bontja. Egy köteg 64 képet tartalmaz a kódomban, ezzel kisebb méretekben, csoportokban tudja feldolgozni a képeket. A feldaraboláson kívül az adatokat véletlenszerűen meg is keveri minden epoch elején, amely azért fontos, hogy a modell a tanulás során ne tanuljon meg fix sorrendet.

Ezek után ellenőriztem, hogy a betöltés és a címkék hozzárendelése megfelelően ment-e végre. Ez azért fontos, mert ha nem tökéletes a címkék hozzárendelése, a modell nem tud megfelelően tanulni. Ahogy a képen látható, a 0. osztály a kosárlabdát ábrázoló képek, az 1. osztály pedig a borítékot, tehát címkék a megfelelő osztályú képekkel lettek összepárosítva.

11. ábra. Adatok betöltése és címkék hozzárendelésének ellenőrzése



Forrás: saját szerkesztés

3.3. Generatív modell architektúra kialakítása

Generátor

A generatív modell architektúra következő eleme a Generator nevű osztályt. Ez a generátor egy neurális hálózat, amely képes lesz véletlen zajból és címkézett információból új képeket előállítani. A Generátorban először egy label embedding réteg szerepel, ez egy beágyazási réteg, amely az osztálycímkéket (integer értékeket) sűrű, numerikus reprezentációvá alakítja. Ez segít, hogy a generátor jobban megértse az osztályok közötti különbségeket.

A Generátor 5 rétegből áll, az első rétegben a bemeneti zajt és az osztálycímkéket nagyobb méretűre skálázza. Aktivációs függvénynek ReLu-t alkalmaz, és BatchNorm2d-t használ a stabilabb tanulási folyamat érdekében. A második rétegben tovább növeli a térbeli dimenziót 14x14-re, miközben csökkenti a csatornák számát 128-ra. A harmadik rétegben az adat mérete már eléri a 28x28-as célméretet, a csatornák számra pedig 64-re tovább csökken. A negyedik réteg egy finomhangoló réteg, amely 32 csatornát állít elő az eredeti térbeli dimenzió megtartása mellett. A végső réteg egyetlen csatornát (szürkeárnyalatos kép) generál. A Tanh aktivációs függvényt alkalmazva az eredményt a [-1, 1] közötti tartományba hozza.

A modellben előre áramoltatáskor az osztálycímkéket átalakítja a label_embedding segítségével, majd lebegőpontos értékekké alakítja, a bemeneti zajt és a címkék reprezentációját pedig összefűzi a dimenziók mentén. Az összefűzött vektort átalakítja olyan formátummá, amelyet a transzponált konvolúciós rétegek inputként várnak. Az input áthalad a generátor modelljén, amely végül egy 1 x 28 x 28 dimenziójú képet ad vissza.

Diszkriminátor

A diszkriminátor modell a generált képek és címkék osztályozására való. Ez a modell két bemenettel rendelkezik, a generált vagy valós (tanító adathalmazból származó) kép, valamint az ahhoz tartozó osztálycímke. A diszkriminátor célja annak az eldöntése, hogy a bemeneti kép (és a hozzá tartozó címke) valós-e vagy generált.

A generátor architektúrájához hasonlóan elsőnek itt is egy label_embedding beágyazási réteget definiálok, amely a címkéket egy térbeli reprezentációvá alakítja, ahol minden címke vektorként jelenik meg. Ez biztosítja, hogy a címke információja közvetlenül integrálható legyen a képadatokkal.

Ezt követi egy konvolúciós hálózat, amely a képeket többszörös konvolúciós rétegen keresztül dolgozza fel, hogy kiemelje a fontos jellemzőket. A hálózat első rétege 64 darab szűrőt használ, hogy csökkentse a kép méretét az eredeti felére, és egy szűkített, de információdús jellemzőhalmazt hozzon létre.

A második réteg további 128 szűrőt alkalmaz, hogy a képet kisebb méretűvé alakítsa. A jellemzők így egyre inkább absztraktabbá válnak. Mindkét réteg LeakyReLU aktivációs függvényt használ, amely segít a negatív értékek megőrzésében, és kötegek szerint normalizált rétegeket, amelyek stabilizálják a tanulási folyamatot. A következő réteg egy teljesen összekapcsolt réteg. Ez egy lineáris réteg, a képadatok és az osztálycímkék kombinációjával dönt arról, hogy a bemeneti kép valós-e vagy generált. Az utolsó aktiváció egy Sigmoid, amely valószínűségi kimenetet ad, vagyis kimenete 0 vagy 1 (valós vagy nem valós a kép).

A diszkriminátor forward metódusa a modell adatfeldolgozási logikáját határozza meg. A képek a konvolúciós rétegeken áthaladva egy síkba tömörített jellemzővektorrá alakulnak. A címkék hasonlóan, a beágyazási réteg által feldolgozott jellemzővektorrá alakulnak. A feldolgozott képjellemzők és osztálycímkék összefűzésre kerülnek, majd a kombinált vektor a teljesen összekapcsolt rétegen áthaladva egy bináris döntést eredményez: valós (1) vagy hamis (0).

3.4. Generatív modell tanítása

A tanítási folyamatot a hiperparaméterek definiálásával kezdtem, megadtam a bemeneti zaj dimenzióját, valamint az osztályok számát. Inicializáltam a generátor és a diszkriminátor modelleket, amelyek példányosításra kerültek a “bemeneti zaj” és “osztályok száma” hiperparaméterekkel.

Mindkettő modellhez külön-külön optimalizálót definiáltam. A generátor az Adam optimalizálót használja, 0,0002-es tanulási rátával. Az Adam optimalizáló rugalmassága miatt ideális GAN modellekhez. A 0,0002-es tanulási ráta egy elterjedt inicializálási érték, amely teszteléseim során megfelelőnek bizonyult az én feladatomhoz is.

A diszkriminátor szintén az Adam optimalizálót használja, de a diszkriminátorhoz kisebb tanulási rátát választottam (0,0001). Ez lehetővé teszi, hogy a diszkriminátor ne tanuljon túl gyorsan, illetve ne legyen túl erős a generátorhoz képest. A modell kezdeti tesztelési fázisában ez a hiperparaméter módosítás áttörő sikereket hozott a modell stabilitására való tekintettel.

Radford "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks (2015)" című művében mindkét optimalizáló betas paraméterei 0,5 és 0,999. Azért választottam ezeket az értékeket, mert az említett tudományos kísérletben hibátlanul működtek, és jó sikereket értek el vele. Ezek az értékek segítenek a mozgóátlag stabilizálásában, ami különösen fontos az olyan modelleknél, ahol a tanítási folyamat instabil lehet.

A Bináris Kereszt-Entrópia veszteség (BCELoss) a generátor és a diszkriminátor közötti ellentétes célokat segíti modellezni. A diszkriminátor arra törekszik, hogy helyesen osztályozza a képeket valósként vagy hamisként, míg a generátor célja, hogy olyan hamis képeket készítsen, amelyeket a diszkriminátor valósként azonosít.

3.4.1. Teljesítménymonitorozás

A tanulási folyamat nyomon követése nagyon fontos, mivel így lehet megbizonyosodni, hogy a generátor és a diszkriminátor is megfelelően fejlődött. Ennek érdekében négy listát inicializáltam:

- A generátor veszteségeinek tárolására: a generátor célja A G_loss minimalizálása, amely akkor valósul meg, ha a generátor képes olyan meggyőző képeket létrehozni, amelyeket a diszkriminátor nem tud hamisnak ítélni. Ha ez az érték nagyon magas, az arra utal, hogy a diszkriminátor mindig helyesen osztályozza a generált képeket, azaz a generátor nem fejlődik. Ha nagyon alacsony, az azt jelentheti, hogy a generátor olyan jó hamis képeket készít, hogy a diszkriminátor nem tudja megkülönböztetni őket a valós képektől.

- A diszkriminátor veszteségeinek tárolására: a diszkriminátor célja ennek a minimalizálása, amely akkor valósul meg, ha a diszkriminátor hatékonyan felismeri a különbséget a valós és a hamis képek között. Ha a diszkriminátor túl jól teljesít (vesztesége közelít a 0-hoz), az azt jelzi, hogy a generátor által készített képek gyengék, és a generátor nem tud megtanulni jobbat készíteni.
- A diszkriminátor valós pontosságára: a diszkriminátor pontosságát méri a valós képek esetében, vagyis ez annak az aránya, hogy a diszkriminátor hány valós képet sorol be helyesen valósnak. Egy jó diszkriminátor esetén a valós pontosság a kezdetben magas, mivel a valós képeket könnyebb felismerni. Ahogy a generátor fejlődik, a diszkriminátornak egyre nehezebb lesz különbséget tennie a valós és hamis képek között, így ez az érték ideális esetben csökken a tanulás során.
- A diszkriminátor hamis pontosságára: a diszkriminátor pontosságát méri a generált képek elutasításakor, vagyis ez annak az aránya, hogy a diszkriminátor hány generált (hamis) képet sorol be helyesen hamisnak. A tanítás elején a diszkriminátor könnyen felismeri a generált képeket, így a hamis pontosság a tanítás kezdeti fázisában magas. Ahogy a generátor kezd jobb képeket készíteni, a diszkriminátor egyre gyakrabban téved, így ez az érték ideális esetben csökken a tanulás során.

Egy kondicionális ellenséges hálózat tanulása során fontos ezeknek az értékeknek a figyelése, csak úgy lehet jó eredményeket elérni, ha fenntartjuk a generátor és diszkriminátor közötti versenyt, amely úgy van egyensúlyozva, hogy mindkettő hálózat fejlődni tudjon, és ne legyen egyik se túlnyomóan erős. Ezeknek az adatoknak az elmentésével a tanulási folyamat után tudom ábrázolni, hogy a megfigyelt értékek hogyan változtak a tanítás során.

3.4.2. Tanítás

Kezdeti paraméterek meghatározása

A modellt ideális esetben érdemes lehet akár 100 epoch-on keresztül is tanítani, azonban a megfelelő erőforrás hiánya korlátozta lehetőségeimet, így 30 epoch hosszúságot választottam, ahol minden epoch a teljes adathalmaz egy teljes körű feldolgozását jelenti.

Diszkriminátor címke-simítása

A modell kezdeti fázisaiban sokszor előfordult a modell összeomlása, amely a generatív ellenséges hálózatok egyik leggyakoribb hibája, amely akkor következik be, amikor a generátor

nem képes változatos mintákat előállítani, hanem egy vagy néhány hasonló mintát ismétel meg. Ez azt jelenti, hogy a generátor "összeomlik", egy vagy néhány adatpéldányt generál újra és újra, ahelyett, hogy az adatok teljes eloszlását megtanulná. Habár a generátor azért generálja ezt az egy adatpéldányt, mert megfelelően becsapja a diszkriminátort, értelemszerűen ez nem jó megoldás. Ennek a kiküszöbölésére vezettem be címke-simítást, amelynek célja, hogy a diszkriminátor ne váljon túl magabiztossá. A valódi képekhez tartozó címkéket, amely eredetileg csak az 1-es értéket vehetik fel, [0.9, 1.0] tartományba helyezi, míg a hamis képekhez tartozó címkéket, amelyek 0-t vehettek fel, a [0.0, 0.1] intervallumba. Ez a technika segít megelőzni a diszkriminátor "tökéletes" tanulását, ami megakadályozhatja a generátor hatékony fejlődését.

Diszkriminátor tanítása

A kód a diszkriminátor tanítása során a generátor bemenetéhez véletlenszerű zajt hoz létre. A generátor a zaj és a valós címkék alapján képeket állít elő, a diszkriminátor először a valós képeket és azok címkéit, majd a generált képeket és azok címkéit osztályozza. A valós és hamis képek esetén külön veszteséget számol, amelyeket átlagolva frissítiem a diszkriminátor súlyait.

Generátor tanítása

A generátor célja, hogy olyan képeket hozzon létre, amelyek a diszkriminátor szerint valósnak tűnnek. A generátor vesztesége azt méri, mennyire sikerült olyan képeket előállítani, amelyeket a diszkriminátor valódiként osztályoz. A generátor súlyai a veszteség deriváltja alapján frissülnek.

Tanítás ellenőrzése

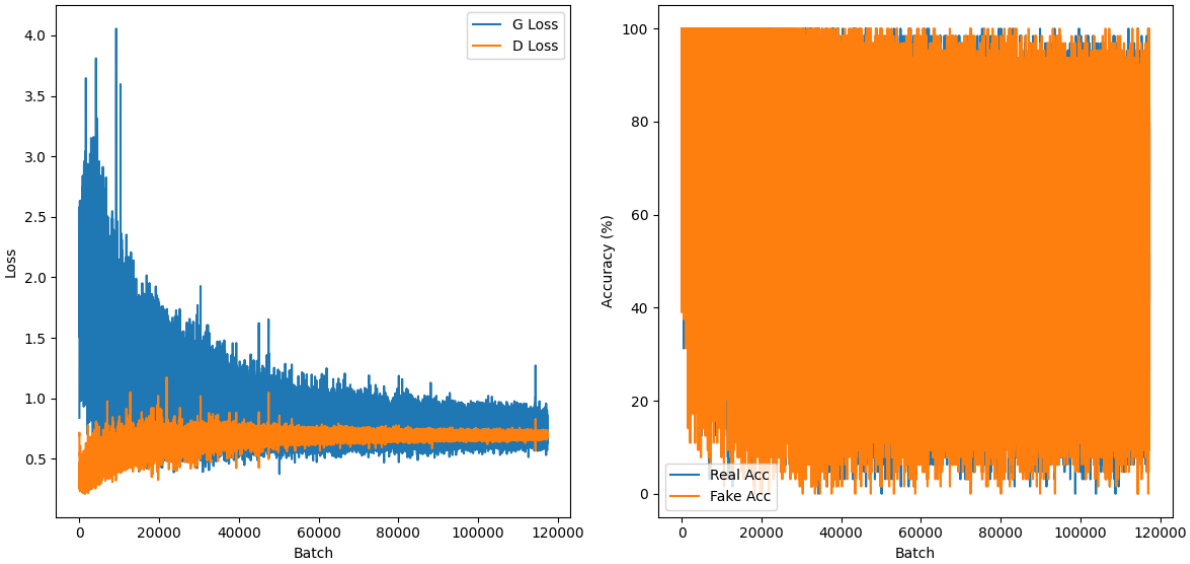
A modell minden 10.000 iteráció után megjelenít generált képeket, a tanítás fejlődésének ellenőrzése érdekében. Ez a lépés segít ellenőrizni, hogy a generált képek javulnak-e az epoch-ok során. Ezen kívül minden epoch végén egy-egy képet generál az összes címkéhez tartozóan. Ez segít átfogó képet kapni arról, hogy a generátor mennyire sikeresen tanul minden osztályból. Az iterációk során 500 kötegenként kiírja az aktuális veszteségeket és pontosságokat. Ez hasznos a tanulási folyamat monitorozásához, és lehetőséget ad a hiperparaméterek időben történő módosítására. A tanítási folyamatoknál sokszor előfordult, hogy a hasonló ellenőrzési technikák értékes időt spóroltak meg, mivel, ha észreveszem, hogy rosszul tanul a modell van lehetőségem javítani és újraindítani a tanítási folyamatot.

4. Eredmények és elemzés

4.1. A generálás eredményének bemutatása

A tanítás befejezése után az egyik fontos lépés az eredmények vizualizálása, hogy jobban megértsük a modell teljesítményét és a folyamat során bekövetkezett változásokat. A nyers veszteségek és pontosságok az iterációk során gyakran tartalmaznak kisebb-nagyobb ingadozásokat vagy zajokat, amelyek nehezen értelmezhetővé tehetik az ábrákat. Ez a zaj különösen jellemző a generatív modellek tanításakor, mivel a generátor és a diszkriminátor folyamatosan egymáshoz alkalmazkodik.

12. ábra. A modell teljesítményének monitorozása simítás nélkül

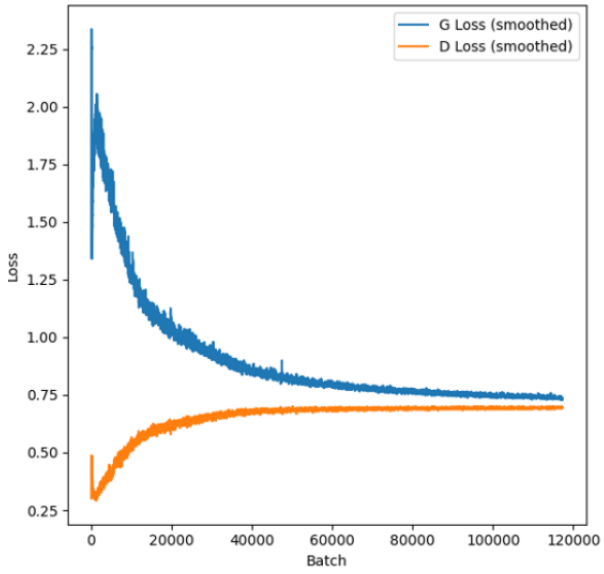


Forrás: saját szerkesztés

Egy egyszerű diagramon ezek az ingadozások elfedhetik a fontos trendeket, például azt, hogy a veszteségek hosszú távon csökkennek-e, vagy hogy a pontosság valóban konvergál-e. Ennek javítása érdekében mozgóátlaggal simított görbe segítségével kiemeltem az általános trendeket, amely megkönnyíti a változások követését.

Az alábbi diagram bemutatja a generátor (G Loss) és a diszkriminátor (D Loss) veszteségeinek alakulását a feldolgozott kötegek előrehaladásával.

13. ábra. A generátor és a diszkriminátor veszteségeinek alakulása a tanulás alatt



Forrás: saját szerkesztés

Ez az ábra segít megérteni, hogyan változtak a veszteségek a tanítás folyamán. Az X-tengely az iterációk számát, az Y-tengely pedig a veszteség értéket mutatja. A grafikonból megállapítható, hogy a modellek megfelelően tanultak.

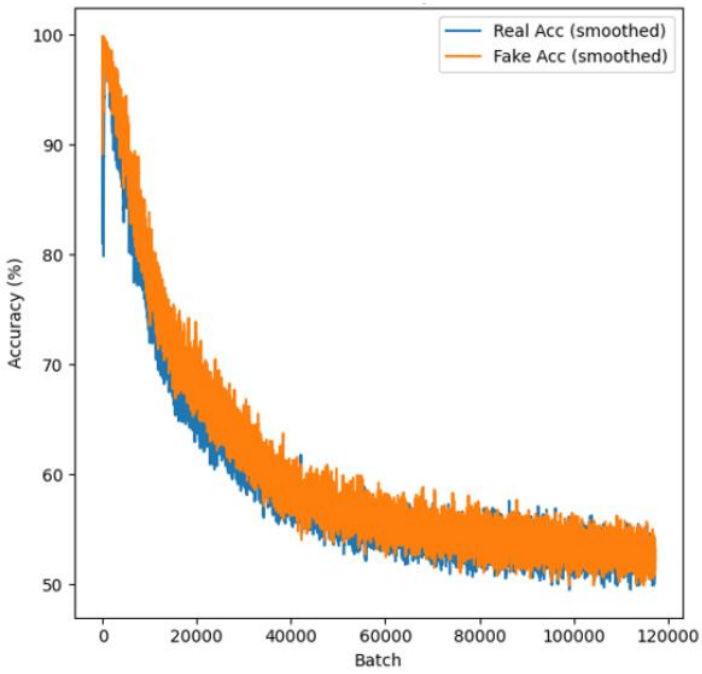
A G Loss először magas értékeket vesz fel, amely teljesen normális, hiszen a diszkriminátor kezdetben könnyedén felismeri melyik kép valódi és melyik hamis. Láthatjuk, hogy ez az érték folyamatosan csökken, és végül a 0,75-ös értékhez konvergál. A 0,75-ös érték arra utal, hogy a generátor képes a diszkriminátor döntéseit megnehezíteni, de nem tudja teljesen becsapni azt. Ez azt jelenti, hogy a generált képek elég valósághűek, és a diszkriminátor nem talál túl könnyen hibát.

A D Loss először alacsony értéket vesz fel, amely szintén várható volt, mivel a generátor kezdeti gyengeségének köszönhetően könnyedén osztályozza helyesen a valódi és hamis képeket. Ahogyan a generátor fejlődik, láthatjuk, hogy értéke elkezd növekedni, és szintén a 0,75-ös értékhez konvergál. A 0,75-ös érték itt azt jelzi, hogy a diszkriminátor nem válik túl erőssé, és még mindig képes helyes döntéseket hozni.

Amikor mindkét veszteségérték 0,75-höz konvergál, és a generált képek jó minőségűek és változatosak, akkor a tanítás sikeresnek tekinthető. Ez azt jelenti, hogy a generátor és a diszkriminátor közötti verseny kiegyensúlyozott volt, és mindkét modell optimálisan tudott fejlődni.

Az következő diagramon a diszkriminátor valós és hamis pontossági arányainak változását ábrázoltam a tanulási folyamat során.

14. ábra. A diszkriminátor valós és hamis pontossági arányai ábrázolva



Forrás: saját szerkesztés

Jól látszik, hogy kezdetben mindkettő érték magas. A tanítás kezdetén a diszkriminátor könnyen felismeri a valódi képeket, mivel a generátor még nem képes valósághű mintákat létrehozni. Ezért a diszkriminátor szinte mindig helyesen azonosítja a valódi képeket, ami magas pontosságot eredményez. Ilyenkor a diszkriminátor nagyon hatékonyan felismeri a generált (hamis) képeket is, mert azok még messze vannak a valóságostól. Ez szintén magas pontosságot eredményez.

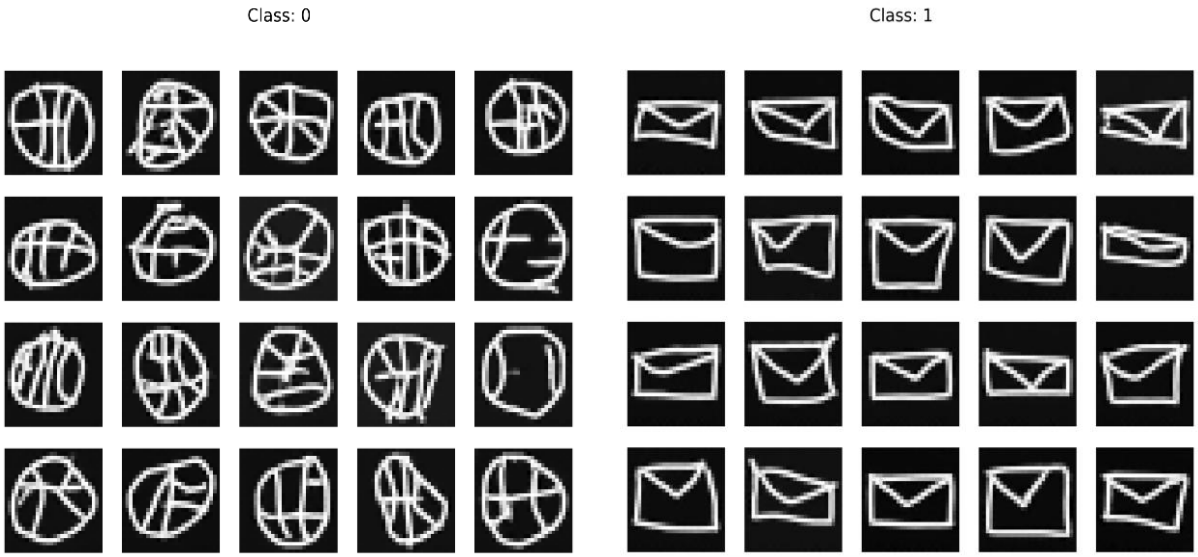
Ahogy a tanítás előrehalad, a generátor egyre valósághűbb képeket hoz létre. Ez megnehezíti a diszkriminátor számára a valódi és hamis képek megkülönböztetését. Egy jól működő GAN esetén ideális egyensúlyban a diszkriminátor számára a generált képek megkülönböztethetetlenné válnak a valódiaktól.

Mindkettő az 50%-os pontossági értékhez konvergál. Ha a diszkriminátor az összes képet 50%-os valószínűséggel minősíti valódinak vagy hamisnak, az azt jelenti, hogy nem képes dönteni, mivel a generátor által létrehozott minták a valós adatokhoz nagyon hasonlóak, így kénytelen “tippelni”, hogy valós vagy hamis képpel van-e dolga. Ez az ideális eredménye a generatív ellenséges hálózat tanításának.

4.2. A generált képek manuális vizsgálata, vizuális megítélése

A korábban is említett kosárlabda, illetve boríték osztályokból való generálás során kapott képek:

15. ábra. A két osztályból generált képek



Forrás: saját szerkesztés

A cGAN modell által generált képek kiváló eredményeket mutatnak mind a kosárlabda, mind a boríték osztályok esetében. A képek jól tükrözik az osztályspecifikus tulajdonságokat: a kosárlabda rajzokon felismerhető a kerek forma és a tipikus barázdák, míg a borítékok egyszerű, de jellegzetes szögletes formái és vonalai egyértelműen az adott osztályra utalnak.

A generált képek minősége figyelemre méltó, hiszen azok nagyban hasonlítanak a tanító adatokra, ugyanakkor egyedi variációkat is tartalmaznak, így nem csupán másolatok, hanem valóban változatos mintákat lehet megfigyelni. Az azonos osztályon belüli képek között érzékelhető a sokszínűség, ami a modell által megtanult reprezentációk gazdagságát jelzi és arra utal, hogy nem volt modell összeomlás a tanítás során. Mindezek alapján a generálás sikeresnek tekinthető, hiszen a képek nemcsak esztétikusak, hanem jól alkalmazhatók az osztályspecifikus tulajdonságok felismerésére is.

A kosárlabda osztálynál megfigyelhető pár pontatlanabb rajz, amelyet első ránézésre nehéz lenne eldönteni mi is valójában. Ez azzal magyarázható, hogy mivel a tanító adatokat emberek alkották, így a részletgazdagabb osztályok képei, amiket nehezebb lerajzolni több pontatlan rajzot tartalmaznak, amelyeket a modell ugyanúgy megtanul, mint a tökéletes

alkotásokat. Egy borítékot lényegesen egyszerűbb lerajzolni, ezért kevesebb rosszul sikerült rajz szerepel a tanító adatok között, amelynek köszönhetően szebb eredményeket tud generálni a modell.

4.3. Generatív modell értékelése a választott mutatókkal

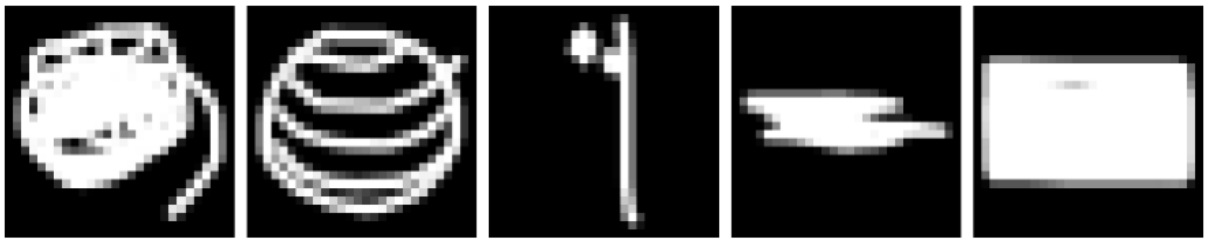
Vizuális megítélés alapján tehát jól néznek ki a képek, sikeres volt a modell tanítása. A modell értékeléséhez azonban különféle módszereket alkalmaztam, amelyek megmutatják, számszerűsítik hogyan teljesít a modell.

4.3.1. Anomália detekció

Első lépésnek anomália detekciót használva megvizsgáltam a tanító adatokat. Kíváncsi voltam, hogy milyen arányban fordulnak elő rosszul elkészített rajzok az adatokban. Ezzel megkaptam, hogy hány olyan van a tanító adatokban, amelyek jelentősen eltérnek a többitől, adott osztályon belül. A képek egy előre betanított Inception-v3 nevű neurális hálón mennek keresztül. Ez a hálózat nem a képek nyers pixeleit elemzi, hanem egy magasabb szintű, 2048 dimenziós jellemzővektort készít róluk, amely összefoglalja a képek fontos tulajdonságait (például formák, textúrák, mintázatok). Mivel a generatív hálózatok képesek modellezni egy adott adathalmaz valószínűség-eloszlását, ezek a modellek megtanulják, hogyan néznek ki a "normális" példák egy adott kontextusban vagy osztályban, így képesek kiemelni azokat az adatokat, amelyek eltérnek ettől az eloszlástól, vagyis anomáliák. A tanító adathalmaz eloszlását az átlagvektor és a kovariancia-mátrix határozza meg, melyek egy többváltozós normál eloszlást alkotnak, leírva a „normális” képeket. Az anomália detekció számításoknál minden esetben 1%-os küszöbértékkel dolgoztam.

Az emberek által rajzolt képeket vizsgálva, a kosárlabda osztályban 907 anomáliát találtam, amely a teljes rendelkezésre álló kosárlabdákat ábrázoló adathalmaz 133 793 képének csupán a 0,68%-a. Pár anomália megjelenítve a kosárlabda osztályból:

16. ábra. A tanító adatokban szereplő kosárlabda anomáliákból vett példák



Forrás: saját szerkesztés

Szintén az emberek által rajzolt képeket vizsgálva, a boríték rajzok között 1597 anomália szerepel, amely az összes 134 863 borítékot ábrázoló kép 1,18%-a. Pár anomália megjelenítve ebből az osztályból:

17. ábra. A tanító adatokban szereplő boríték anomáliákból vett példák

Forrás: saját szerkesztés

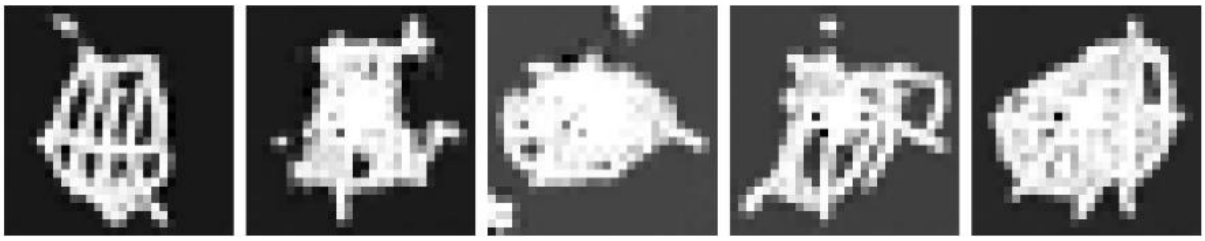
Láthatjuk, hogy a korábban említettekkel ellentétben nem a kosárlabda osztályban szerepel több anomália, hanem a boríték osztályban, annak ellenére, hogy összetettebb, részletgazdagabb és nehezebb lerajzolni egy kosárlabdát. Ennek oka valószínűleg az, hogy az envelope (boríték) mint angol szó kevesebb ember számára ismert, mint a basketball (kosárlabda). Mivel a Quick, Draw! egy rajzolós játék, valószínűleg a fiatalabb generációk körében népszerűbb, a mai digitális korszakban pedig a digitális kommunikáció elterjedésével a borítékok használata kevésbé ismert, a felhasználók egy része nem ismeri a szó jelentését, ezért nem tudja mit kell lerajzolni.

Adattisztítási lépésként az anomália detekcióval kiszűrt képeket akár el is lehetne távolítani a tanító adatok közül, azonban ezt azért nem tettem meg, hogy ne csak jól sikerült rajzokon tanuljon a modell, így a valós adatokat jobban reprezentálja a modell.

Ezek után megvizsgáltam a cGAN modell segítségével generált 10 000 képet osztályonként és megkerestem ezekben az anomáliákat. Minden generált képre egy log-likelihood pontszámot számol a modell, ami megmutatja, mennyire illeszkedik az a tanító adathalmazhoz.

A kosárlabda osztályból generált 10 000 kép közül 65 darabot azonosított anomáliaként a modell, amely 0,65%-nak felel meg. Ez az érték nagyon közel áll a tanító adatokban szereplő anomáliák arányához, így elmondható, hogy a modell jól reprezentálja a tanító adatok valós eloszlását. Pár anomália megjelenítve a generált kosárlabdákból:

18. ábra. A generált kosárlabda képekből vett anomáliák



Forrás: saját szerkesztés

A boríték osztályból szintén 10 000 képet generáltam, ebben 486 darab anomália szerepel. Ez 4,86%, amely több, mint négyszerese a tanító adatokban lévő anomáliák arányához képest. Ez azért lehet, mert az egyszerűbb felépítésű képek osztályában az adatok kevésbé komplexek, ami azt eredményezheti, hogy a modell hajlamosabb a túláltalánosításra, vagy a nem jellemző mintázatok megtanulására. Ez a generált képek között magasabb anomália arányt okozhat. A kosárlabda osztály képei többféle jellemzővel rendelkeznek, amelyek jobban differenciálják az anomáliákat, ezáltal megkönnyítik a modell számára az adatok helyes eloszlásának megtanulását. Ez nem mondható el a boríték osztályról, ezért nagyobb arányban fordulnak elő anomáliák. Pár anomália megjelenítve a generált borítékokból:

19. ábra. A generált boríték képekből vett anomáliák

Forrás: saját szerkesztés

A generatív modell eredményeit a már korábban is említett Inception Score (IS) és Fréchet Inception Distance (FID) mutatókkal értékeltem, amelyek számszerűsítik az eredmények minőségét. Az IS és az FID két elterjedt mérőszám, amelyeket generatív modellek teljesítményének értékelésére használnak.

4.3.2. Inception Score kiszámítása

Az Inception Score kiszámítását először külön végeztem el az osztályokon. A kosárlabda osztályból generált 10 000 kép IS értéke 1,90, a boríték osztály IS értéke pedig 1,88. Mivel a generált képek szemmel láthatóan jól néznek ki, de az IS értéke nem ezt tükrözi, ezért

a hiba feltárása érdekében kiszámítottam az Inception Score-t csak a tanító adatokon is, ezzel tesztelve az Inception Score kompatibilitását az adathalmazzal. Ekkor a kosárlabda osztályra 2,27-et kaptam, a boríték osztályra pedig 2,04-öt. Ezek mind alacsonynak nevezhetők, amely azért lehet, mert az Inception Score azt vizsgálja, hogy a generált képek mennyire "tiszták" és mennyire "változatosak" az Inception-v3 modell osztályozása alapján. Az IS alapja, hogy az Inception-v3 modell milyen magabiztosan sorolja a képeket különböző kategóriákba. Ez azt jelenti, hogy a generált képnek valódinak kell tűnnie a modell számára.

Az Inception-v3 modellt a ILSVRC 2012 adatbázison tanították, ami 1000 különböző kategóriát tartalmaz (például állatok, járművek, hétköznapi tárgyak). Mivel a Quick, Draw! adathalmaz képei nem valódi fényképek, hanem fekete-fehér rajzok, a modell valószínűleg nem tudja ezeket jól felismerni, hiszen ezek nem hasonlítanak azokra a valódi fotókra, amelyeken a modellt tanították. Ez alacsony IS értéket eredményez, mert a modell nem képes magabiztos osztályozást adni ezekre a képekre, így kijelenthető, hogy a modell kiértékelésére kompatibilitási problémák miatt nem alkalmazható az Inception Score.

4.3.3. Fréchet Inception Distance (FID) kiszámítása

Az FID értéket először a modell által generált kosárlabda képeken számoltam ki, amelyre 21,97-es értéket kaptam, majd a boríték osztályra, amely pedig 24,66-os FID értéket ért el. Ezek mind jó értéknek számítanak, tehát a generált képek eloszlása kellően hasonló a tanító adatok eloszlásához. Elmondható, hogy a generálás sikeres volt, eredménye a tanító adatokhoz hasonló eloszlású és minőségű képek.

5. Következtetések

5.1. A feldolgozott téma összefoglalása, hipotézisek megválaszolása

A szakdolgozatban bemutatott kutatás célja az volt, hogy megvizsgáljam a generatív mesterséges intelligencia, különösen a feltételes generatív ellenséges hálózatok (cGAN) képességeit az emberi alkotásokhoz hasonló képek létrehozásában. A kutatás a Google által fejlesztett „Quick, Draw!” adathalmazra összpontosított, amely emberek által, kézzel készített rajzokat tartalmaz. Ezen adathalmaz alapján tanítottam a cGAN modellt, amely segítségével arra törekedtem, hogy új, korábban nem létező rajzokat hozzak létre.

A dolgozat alapos elméleti háttérrel kezdődött, amely bemutatta a mesterséges intelligencia, a gépi, illetve mélytanulás, valamint a generatív modellek alapjait. Ezen belül részletesen ismertettem a neurális hálózatok működését, a cGAN felépítését, és a generatív modellek alkalmazási lehetőségeit. A kutatás esettanulmányi részében a saját készítésű generatív modellre fókuszáltam. Az adathalmaz bemutatása és előkészítése után a modell felépítése és tanítása következett. A modell tanítása során különös figyelmet fordítottam a diszkriminátor és generátor közötti egyensúly fenntartására, valamint a teljesítmény monitorozására.

A generálás eredményei azt mutatták, hogy a modell képes az emberi alkotásokhoz hasonló képeket létrehozni. A generált képek kvalitatív vizsgálata alapján vizuálisan jól kidolgozott rajzok születtek, amelyek az esetek többségében elérik, sőt, sokszor meg is haladják az emberi alkotások szintjét.

Kvantitatív értékelés

A dolgozatban alkalmazott kvantitatív mutatók között szerepelt az Inception Score (IS) és a Fréchet Inception Distance (FID). Az IS alkalmazása során kiderült, hogy ez a mérőszám nem volt kompatibilis a modell által használt rajzokkal. Az IS ugyanis inkább komplexebb, valósághűbb képek vizsgálatára lett kifejlesztve, így az általam használt adathalmaz jellege miatt nem tudta teljes mértékben tükrözni a generált képek valódi minőségét. Ezzel szemben a FID jobb kompatibilitást mutatott az adathalmazzal, és pontosabb képet adott a generált rajzok minőségéről. A FID alapján a generált képek hasonló eloszlással rendelkeznek, mint az eredeti adatok, ami a modell jó generatív teljesítményét igazolja.

Kvalitatív értékelés

Az emberi értékelések során a generált rajzok vizuális minőségét és felismerhetőségét vizsgáltam. A kiváló vizuális minőség a modell tanítási folyamatának sikerét tükrözi. A generált képek között jól megfigyelhető és elkülöníthető a generáláshoz használt két osztály. A kosárlabda osztály képeiben fellelhető sajátos mintázatok a másik osztályban nem szerepelnek, az osztályok képeinek jellemzői között nincs átfedés. Egy véletlen mintavételezéssel kiválasztott képet könnyedén el lehetne dönteni melyik osztályba tartozik.

A dolgozat első hipotézise szerint lehetséges olyan magas minőségű képek (rajzok) generálása a „Quick, Draw!" adathalmazon tanított feltételes generatív ellenséges hálózat (conditional Generative Adversarial Network, cGAN) használatával, amelyek eddig nem léteztek. A kutatás bebizonyította, hogy a feltételes generatív ellenséges hálózatok képesek magas minőségű, emberi alkotásokhoz hasonló szintű képek generálására. A generált képek a hozzájuk rendelt címkék szerinti osztályba tartoztak, az osztályok egyértelműen elkülönültek egymástól. Bár a generált képek egyszerűek voltak, azok a modell tanításához használt adathalmaz jellegzetességeit kiválóan tükrözték. A generált rajzok több esetben meghaladták a várakozásokat, bizonyítva, hogy a cGAN képes, és hatékony eszköz kreatív tartalom előállításában.

A dolgozat második hipotézise azt állította, hogy a generált képek megkülönböztethetetlenek az ember által készített képektől, kvantitatív metrikák, illetve kvalitatív emberi értékelések által egyaránt. Annak ellenére, hogy az anomália detekció elvégzése rávilágított a rosszul generált rajzokra, ezeknek az aránya elenyésző és hasonló volt a tanító adatokban is szereplő rossz képek arányához. A generált képek az esetek túlnyomó többségében megkülönböztethetetlenek voltak az emberi alkotásoktól. A kvantitatív mutatók közül az Inception Score nem volt ideális a generált képek minőségének vizsgálatához, amely a tanító adatok jellege miatt kompatibilitási problémához vezetett és hamis pontatlan értékelést eredményezett. A vizuális megítélés, anomália detekció és a FID mutatók alkalmazásával megállapítottam, hogy a generált képek minősége, változatossága és realizmusa megkülönböztethetetlen az emberi alkotásoktól.

A kutatás rámutatott arra, hogy a generatív modellek, különösen a cGAN, rendkívüli potenciállal rendelkeznek a művészeti és a kreatív iparágak területén. Segítségükkel lehetséges korábban nem létező tartalmakat létrehozni, amely számos új lehetőséget kínál. A Quick,

Draw! adathalmaz egyszerűsége ellenére a modell képes volt értékes és kreatív rajzokat előállítani. Az IS értékelési nehézségei ugyanakkor rávilágítottak arra, hogy a modell értékeléséhez alkalmazott metrikák kiválasztása kritikus fontosságú, különösen az olyan adathalmazok esetében, amelyek sajátos, egyszerűsített jellemzőkkel rendelkeznek.

5.1.1 Javaslatok további kutatásokra a témában

A generatív modellek művészetekben való felhasználása, különösen a képek, zenék és videók generálása továbbra is jelentős mennyiségű kutatást igényel. Érdemes lenne részletesebben vizsgálni, hogyan befolyásolják ezek a modellek az emberi kreativitást, és milyen szerepet játszanak a művészeti alkotások létrehozásában.

A mesterséges intelligencia, különösen a generatív modellek fejlődése számos etikai és társadalmi kérdést vet fel, amelyek további kutatásokat igényelnek. A mesterséges intelligencia által generált művészeti alkotásokkal kapcsolatos jogi kérdések, mint például a szerzői jogok, az alkotói identitás, valamint a mesterséges intelligencia és az emberi alkotó közötti határvonalak kérdése, mind olyan területek, amelyek további kutatást igényelnek. Ezen kívül a generatív modellek által előállított egyre valósághűbb hamis információk terjedésének megelőzésére és kezelésére is fokozottan nagyobb hangsúlyt kell fektetni.

A jövőbeni kutatások egyik kulcsfontosságú aspektusa a generatív modellek továbbfejlesztése. A mélytanulás és a generatív ellenséges hálózatok területén folytatott fejlesztések új lehetőségeket teremthetnek a képgenerálás minőségének javításában, különösen olyan alkalmazásokban, mint a szórakoztatóipar, filmipar, valamint a digitális művészetek és a vizuális kommunikáció.

A generatív modellek teljesítménye nagymértékben függ a tanításhoz alkalmazott adathalmazok minőségétől és változatosságától. További kutatási kérdés, hogy hogyan javíthatók, illetve bővíthetők a tanításra használt adathalmazok, hogy a modellek képesek legyenek jobban kezelni a sokszínű adatokat, és hogy az új modellek miként képesek alkalmazkodni a különböző stílusokhoz, technikákhoz és művészeti hagyományokhoz. Végül, a kutatásoknak érdemes lenne foglalkozniuk a mesterséges intelligencia és az emberi kreativitás határvonalának újraértékelésével. Milyen szerepe van az emberi kreativitásnak a generatív modellek által alkotott művekben? A mesterséges intelligencia valóban képes-e új tartalmat létrehozni, vagy csupán a tanulás és adaptáció eszközeit alkalmazza? Ha egy ember létrehoz egy generatív modellt, a végeredményként kapott mű a modell alkotóját vagy a

generatív modellt illeti? Az ilyen kérdések megválaszolása mélyebb filozófiai és gyakorlati kutatásokat igényel, amelyek képesek lennének új fényt deríteni a mesterséges intelligencia és az emberi alkotás közötti kapcsolatra.

Ezek a kutatási irányok lehetőséget adnak arra, hogy a generatív modellek területe továbbra is új kihívásokkal és kérdésekkel bővüljön, és hogy a mesterséges intelligencia más iparágakba is sikeresen integrálódni tudjon és még szélesebb körben elterjedjen, negatív következmények nélkül.

6. Befejezés

A kutatás rávilágított arra, hogy a generatív mesterséges intelligencia, különösen a feltételes generatív ellenséges hálózatok (cGAN), hatékony eszközei lehetnek új, korábban nem létező vizuális tartalmak előállításának. Az eredmények alátámasztották, hogy a modell képes volt a „Quick, Draw!” adathalmazban található emberi rajzok jellemzőit elsajátítani, és azok alapján új mintákat generálni. Ez a technológia egyértelműen demonstrálta a generatív modellek alkalmasságát egyszerűbb vizuális adathalmazok feldolgozására és a tanult eloszlások valósághű visszaadására.

A hipotézisek vizsgálata során az első hipotézis igazolást nyert, miszerint a cGAN modellek képesek a tanító adathalmaz alapján olyan, magas vizuális minőségű képeket előállítani, amelyek nem csupán az eredeti adathalmaz mintázatait tükrözik. A kvalitatív vizsgálatok alapján a generált rajzokon jól azonosíthatók a képek osztályspecifikus jellemzői. A második hipotézis, amely szerint a generált képek megkülönböztethetetlenek lennének az emberi rajzoktól, szintén igazolást nyert. Bár kvalitatív emberi értékelések alapján a képek többsége az emberi alkotásokkal összevethető minőségű volt, a kvantitatív mutatók közül az Inception Score korlátokba ütközött az adathalmaz egyszerűbb felépítése miatt. Ugyanakkor a Fréchet Inception Distance (FID) eredményei jobb kompatibilitást mutattak a vizsgált adatokkal, és hibátlanul jelezték a generált képek eloszlásának hasonlóságát az eredeti adatokhoz.

Összességében a kutatás bebizonyította, hogy a cGAN modellek alkalmazása nemcsak az egyszerűbb vizuális adatok, hanem az alkotófolyamatok technológiai támogatására is ígéretes megközelítést kínál. Az alkalmazott metrikák és módszerek rámutattak a modellek teljesítményének és a kiértékelési rendszerek fejlesztésének fontosságára. Ezek az eredmények megerősítik, hogy a generatív mesterséges intelligencia a jövőben jelentős szerepet játszhat a kreatív iparágakban, ugyanakkor újabb kihívásokat is teremt, amelyek további figyelmet és kutatást igényelnek.

7. Irodalomjegyzék

7.1. Felhasznált szakirodalmak

Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein GAN. In Proceedings of the International Conference on Machine Learning (ICML). https://arxiv.org/pdf/1701.07875

Barratt, S., & Sharma, R. (2018). A note on the Inception score. *arXiv*. https://arxiv.org/abs/1801.01973

Bernard, M. (2024. April 18). The Role Of Generative AI In Video Game Development. Forbes. https://www.forbes.com/sites/bernardmarr/2024/04/18/the-role-of-generative-ai-in-video-game-development/

Bishop, C. M. (2006). Pattern recognition and machine learning. Springer. https://www.microsoft.com/en-us/research/uploads/prod/2006/01/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf

Fazekas, I. (n.d.). Neurális hálózatok. https://gyires.inf.unideb.hu/GyBITT/19/Neuralis_halozatok_v8.pdf

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial networks. arXiv preprint arXiv:1406.2661. https://doi.org/10.48550/arXiv.1406.2661

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press. http://www.deeplearningbook.org

Haykin, S. (1998). Neural Networks: A Comprehensive Foundation. Prentice Hall.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770–778. https://doi.org/10.1109/CVPR.2016.90

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). GANs trained by a two time-scale update rule converge to a local Nash equilibrium. *arXiv*. https://doi.org/10.48550/arXiv.1706.08500

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. Neural Computation, 14(8), 1771–1800. https://doi.org/10.1162/089976602760128018

Kingma, D. P., & Welling, M. (2014). Auto-encoding variational Bayes. In Proceedings of the International Conference on Learning Representations (ICLR). https://doi.org/10.48550/arXiv.1312.6114

Kitao, A. (2022). Principal component analysis and related methods for investigating the dynamics of biological macromolecules. Journal of Biomolecules, 5(2), Article 21. MDPI. https://doi.org/10.3390/j5020021

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems, 25, 1097–1105. https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444. https://doi.org/10.1038/nature14539

Mehrabi, M., Mohammadkarimi, M., Ardakani, M., & Jing, Y. (2019, Jan. 11). Decision directed channel estimation based on deep neural network k-step predictor for MIMO communications in 5G. *arXiv*. Retrieved from https://export.arxiv.org/pdf/1901.03435

Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784. https://arxiv.org/abs/1411.1784

Murphy, K. P. (2012). Machine learning: A probabilistic perspective. MIT Press.

Ng, A. (n.d.). Machine learning yearning. Self-published. https://gyires.inf.unideb.hu/GyBITT/19/Neuralis_halozatok_v8.pdf

Nielsen, M. A. (2015). Neural networks and deep learning. Determination Press. http://neuralnetworksanddeeplearning.com

Patel, A., & Rama, R. K. (2020). An overview of Boltzmann machine and its special class. https://doi.org/10.13140/RG.2.2.28630.88641

Pratik, R. (2024, October 21). A Complete Guide to Generative AI in Healthcare. Intuz. https://www.intuz.com/generative-ai-in-healthcare-guide

Radford, A., Metz, L., & Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. In Proceedings of the International Conference on Learning Representations (ICLR). https://arxiv.org/pdf/1511.06434

Russell, S. J., & Norvig, P. (2020). Artificial Intelligence: A Modern Approach (4th ed.). Prentice Hall

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. Visual Geometry Group, Department of Engineering Science, University of Oxford. arXiv preprint arXiv:1409.1556. https://arxiv.org/abs/1409.1556

TechStoryteller. (2023, October 28). Fashion Forward: The Influence of Generative AI in the Clothing Industry. Medium. https://medium.com/@chaitechstoryteller/fashion-forward-the-influence-of-generative-ai-in-the-clothing-industry-2499cc6faf04

Wu, J. (2017, May 1). Introduction to convolutional neural networks. LAMDA Group, National Key Lab for Novel Software Technology, Nanjing University. https://cs.nju.edu.cn/wujx/paper/CNN.pdf

Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In European Conference on Computer Vision (ECCV). Springer. https://doi.org/10.1007/978-3-319-10590-1_53

7.2. Ábrajegyzék