Eötvös Loránd Tudományegyetem
Természettudományi Kar

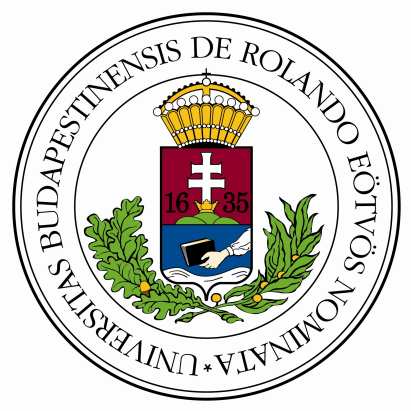
Diffúziós modellek és alkalmazásaik

Szakdolgozat

Szemán Dávid
Matematika BSc
Matematikai elemző szakirány

Témavezető:

Lukács András
Számítógéptudományi Tanszék



Budapest
2023

Köszönetnyilvánítás

Ezen sorokkal szeretnék köszönetet mondani mindazoknak, akik hozzájárultak ahhoz, hogy a szakdolgozatom elkészülhessen. Elsősorban köszönettel tartozom Lukács András Tanár Úrnak, aki elkötelezetten segített és támogatott engem a folyamat minden szakaszában. Az iránymutatásai nélkül nem sikerült volna belevetnem magam a mélytanulás és a diffúzió izgalmas világába. Külön köszönettel tartozom a családomnak és a barátaimnak, akik a támogatásukkal és biztatásukkal mindig a segítségemre voltak, amikor szükségem volt rá.

Tartalomjegyzék

1. Bevezetés

A szakdolgozat középpontjában a diffúziós modellek állnak, melyek eredeti értelmezései alapvető fizikai és matematikai folyamatok. A diffúzió során egy anyag, információ vagy jelenség terjed egy térbeli vagy időbeli gradiens mentén, az anyagrészecskék vagy információrészletek véletlenszerű mozgása révén.
A diffúziós folyamatok modellezése az ősrobbanás elméletével indult, amikor a korai univerzum anyagainak terjedését és eloszlását vizsgálta, és ezen keresztül felmerült a diffúzió jelenségével kapcsolatos első gondolatok a kozmológiában. A 20. század elején különböző tudományterületeken matematikai modellek kezdtek kialakulni, amelyek leírták a diffúziót, ezzel lehetőséget teremtve a tudományágak közötti kiterjesztésre. A kémia területén az 1930-as évektől kezdve növekvő érdeklődés övezte a diffúziót, különösen a kémiai reakciók és gázok közötti anyagtranszport terén. A biológiai kutatások során a diffúziós modellek hozzájárultak a megértéshez, hogy különböző molekulák miként terjednek el sejtekben és szövetekben. Az utóbbi évtizedek fejlett matematikai módszerei és a számítástechnika fejlődése révén lehetőség nyílt bonyolultabb diffúziós modellek felfedezésére, amelyek nagyobb pontosságot és alkalmazhatóságot biztosítanak a különböző tudományterületeken.

A diffúzió matematikai alapjai néhány alapvető elvben összpontosulnak. Az első alapelv a részecskék véletlenszerű mozgása. A diffúziós folyamat során a részecskék véletlenszerűen mozognak a térben. Ezen ugrások során a részecskék olyan irányokba és sebességekkel mozognak, amelyek nem jelezhetők előre. A második alapelv, amely szerint a részecskék a diffúziós folyamat során a magasabb koncentrációjú területekről az alacsonyabb koncentrációjú területekre mozognak. Ezen mozgások következtében a koncentrációs különbségek fokozatosan csökkennek, ami hosszabb távon az anyagok egyenletes eloszlásához vezet a rendszerben. A harmadik alapelv a Fick-törvény, amely leírja az anyag diffúzióját. Ez a törvény alapján a diffúziós sebesség a koncentrációgrádienssel van összefüggésben, és meghatározza, hogy az anyag milyen gyorsan terjed el a térben.

$$J = -D \cdot \nabla C \tag{1}$$

A Fick-törvény 1 egyenletében a J jelöli a diffúziós fluxust, vagyis az anyagrészecskék sűrűségváltozásának sebességét egy adott területen, a D az anyag diffúziós együtthatója, amely az anyag jellegétől függ és a ∇C a koncentráció-

grádienset fejezi ki, vagyis azt, hogy hogyan változik az anyag koncentrációja egy adott térben. A Fick-törvény azt mutatja be, hogy az anyag részecskéi azonos irányban mozognak a koncentrációgrádiens irányában, és a diffúziós sebesség nagysága a gradiens erősségével és az anyag diffúziós tulajdonságaival arányos. Az egyszerű Fick-törvény közvetlenül alkalmazható homogén közegben, de összetett körülmények esetén módosítható.

A mély tanulás fejlődése új lehetőségeket nyitott meg a diffúzió alkalmazásában, mint a DALL-E és a MidJourney, amelyek tovább bővítik a diffúziós modellek alkalmazásának területét. Ezek a rendszerek kreatív művészi alkotások terén és tartalomgenerálásban használják ki a diffúziós modellek előnyeit.

A szakdolgozat célkitűzése, hogy részletes és alapos bemutatást nyújtson a diffúziós modellek világáról a gépi tanulásban, feltárva ezek matematikai alapelveit és alkalmazásait a generatív képalkotó folyamatok területén. A generatív diffúziós modellek célja nem csupán az adathalmaz újraalkotása, hanem annak olyan mély megértése, amely lehetővé teszi új, eddig nem látott adatok létrehozását. Ebben az összefüggésben a modellnek olyan részletességgel kell megismernie az adathalmazt, hogy képes legyen a benne rejlő struktúrák, mintázatok és összefüggések általánosított értelmezésére. A generatív modellezés nem csupán technológiai fejlődést hozott, hanem lehetőséget teremt a kreatív alkotásra, az innovációra, és elősegíti az ismeretek továbbfejlesztését.

A dolgozat struktúrája három fő részből áll. A második fejezet részletesen bemutatja az Autoencoder és az U-Net működését kiemelve ezek kulcsszerepét a generatív modellezésben, miközben egyúttal elmélyül a valószínűségszámítás és a kapcsolódó fogalmak iránti megértés kialakításában. A harmadik fejezet a diffúziós modellek koncepcióját és matematikai alapelveit részletezi, teremtve egy közvetlen kapcsolatot a generatív folyamatok és a matematika logikája között. Ez a rész kiterjed a DDPM, ARDM, DDIM modellekre és a Score-Based módszerre, melyek nagy előrelépéseket hoztak a generatív modellezés terén. A negyedik fejezetben a gyakorlatba ágyazva bemutatásra kerül a DDPM modellek alkalmazása valóságos adathalmazokon, programkódok segítségével MNIST és CIFAR-10 adathalmazon. A programkódok és az azokhoz kapcsolódó eredmények megtalálhatók a következő GitHub oldalon https://github.com/SzemanDavid. A két alkalmazás segít megvilágítani, hogy hogyan működnek a diffúziós modellek valóságos környezetben.

Ez a szakdolgozat egy átfogó képet nyújt a generatív modellezés egyes kulcsfontosságú aspektusairól és alkalmazásairól. A diffúziós modellek világa összefonódik a tudomány, a matematika és a gyakorlati alkalmazások széles skálájával, és ezen dolgozat kifejezett szándéka, hogy ezt a sokszínű területet megközelíthetővé tegye.

2. Mély tanulás alapjai

A mély tanulás alapelvei a mesterséges neurális hálózatokon alapulnak. Ezek a hálózatok inspirációt merítenek az emberi agy működéséből. Olyan matematikai elveken nyugszanak, amelyek lehetővé teszik a tanulást és a következtetéseket. A mesterséges neurális hálózatok számítási egységekből állnak, amelyeket neuronoknak nevezünk. A neuronok kapcsolatokon keresztül kommunikálnak egymással, hasonlóan az emberi agyban található idegsejtekhez. Egy neurális hálózatban a neuronok súlyozott bemeneteket kapnak és a súlyok határozzák meg, hogy mennyire fontos egy adott bemenet az adott neuron számára. A tanulási folyamat során a hálózatoknak be kell állítnunk a súlyokat az optimális eredmény eléréséhez. Ehhez a gradiens számítás segítségével meghatározzák, hogy milyen irányba kell módosítani a súlyokat a veszteség függvény minimalizálása érdekében.

A **Gauss-eloszlás** használata zajosítás céljából optimális a mély tanuló modellek tanításában. Matematikailag jól leírható, mivel a valódi világban sok természetes és emberi folyamat során előforduló kisebb változásokat szimulál. Egyik jól ismert jellemzője, hogy a zaj értékei közötti eloszlás megközelítőleg szimmetrikus, ami azt jelenti, hogy a pozitív és negatív zajértékek egyaránt előfordulhatnak. Emellett a Gauss-eloszlás varianciája, intenzitása vagy terjedése ellenőrizhető, így alkalmazása rugalmas és szabályozható. A mély tanulásban a Gaussi-eloszlásból származó zajt gyakran használják a tanítási folyamat során a hálózatok regularizálására és a túltanulás elleni védekezésre. A bemeneti adatokhoz vagy a rejtett rétegekhez hozzáadott zaj segíthet abban, hogy a modellek általánosabban alkalmazhatók legyenek, nem csupán a tanító adathalmazokra, hanem más, zajosabb adatokra is. Ezenkívül a Gauss-eloszlás segíthet a hálózatok konvergenciájának elősegítésében a tanítási folyamat során. A Gauss-eloszlás sűrűségfüggvénye a következő képlet szerint írható le. Legyen

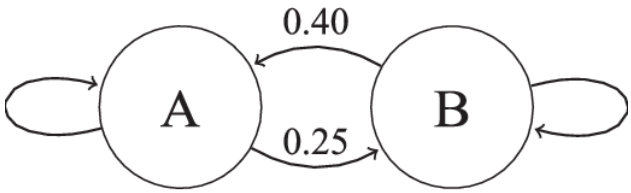
$$p(x) = \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \tag{2}$$

ahol:

- μ a normális eloszlás középértéke,
- σ a szórás.

Az alkalmazásokban a zaj generálásához rendszerint véletlen generátorokat használhatók amelyek alapján a fenti képlet paramétereit véletlenszerűen generálódnak. Az így generált zaj hozzáadható bemeneti adatokhoz vagy alkalmazható különböző statisztikai feladatokban a valószínűségek modellezésére.

A **Markov-lánc** egy valószínűségi modell, amely a jövőbeli események valószínűségét az előző események alapján jósolja meg. Az elnevezés az orosz matematikus, Andrei Markov nevéhez köthető, aki a 20. század elején foglalkozott ezzel a témával. A Markov-láncnak nincs memóriája, azaz a jövőbeli állapot csak az aktuális állapottól függ, ez a Markov-tulajdonság. Ez azt jelenti, hogy a múltbeli események nincsenek közvetlen hatással a jövőre, csak az aktuális állapot számít. A Markov-láncot állapotok és állapotváltozó valószínűségek jellemzik. Az állapotok folytonos vagy diszkrét értékeket vehetnek fel, és az állapotváltozó valószínűségek meghatározzák, hogy az adott állapot milyen valószínűséggel változik egy másik állapotba az idő múlásával. A folytonos állapotterű Markov-lánc egy olyan modell, amely a rendszer állapotát folytonos változóként kezeli, például idő, tér vagy egyéb kontinuális skála mentén.
Nagy szerepük van a mélytanulásban. A Markov-lánc alkalmazása lehetővé teszi a zajos adatok folyamatosan történő tisztítását és az információk szép lassan történő rekonstrukcióját. Ez a folyamat segít a generatív modelleknek az adatok pontosabb reprodukálásában és az előrejelzési feladatokban.



1. ábra. Egy Markov-lánc, amely 2 állapotból áll A és B. A két állapot között vektorok vannak a példában, amelyek az állapotváltozó valószínűségeket jelölik. Értékük azt fejezi ki, hogy mennyi a valószínűsége annak, hogy az adott állapotból a másikba vagy önmagukba lépnek a következő időpillanatban. [ábra forrása]

A **Kullback-Leibler (KL) divergencia** egy mérőszám vagy távolságmetrika, amelyet a valószínűségi eloszlások közötti különbség vagy hasonlóság

mérésére használnak. A D_{KL} divergencia azt mutatja meg, mennyire tér el egy valószínűségi eloszlás egy referencia eloszlástól. Matematikailag két valószínűségi eloszlás $P(x)$ és $Q(X)$ KL-divergenciája a következőképpen számítható ki folytonos esetben

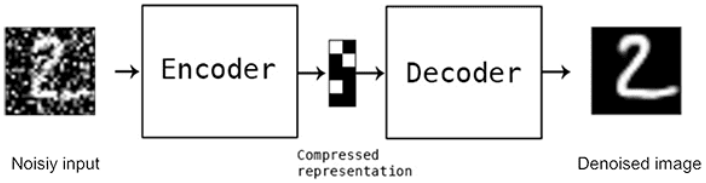
$$D_{KL}(P||Q) = \int_{-\infty}^{\infty} P(x) \cdot \log\left(\frac{P(x)}{Q(x)}\right) dx, \tag{3}$$

ahol $P(x)$ az adatokat vagy mért valószínűségi eloszlást jelképezi és $Q(x)$ eloszlás egy elméletet, modellt vagy P-nek egy közelítését, ebben az esetben a KL-divergencia azt méri, hogy mennyire tér el $Q(x)$ a $P(x)$-től. Gyakran használják generatív modellek értékelésére, ahol a cél az, hogy meghatározzuk, mennyire felel meg egy modell a valós eloszlásnak. Az alacsony KL-divergencia azt jelzi, hogy a generált eloszlás közel van a valóságos eloszláshoz, vagyis a generatív modell jól teljesít. Magasabb KL-divergencia esetén a generált eloszlás és a valóságos eloszlás közötti különbség nagy, ami arra utal, hogy a modellnek további fejlesztésre van szüksége.

2.1. Autoencoder

Az Autoencoderek, melyeket Geoffrey Hinton vezetett be először 1986-ban, kulcsfontosságú mély tanulási architektúrák, amelyek megszerezték helyüket a gépi tanulás világában.
A következő fejezetben a [6] forrás szolgált alapul. Az Autoencoderek olyan hálózatok amelyek hatékony eszközök az adatok rejtett szerkezetének és fontos jellemzőinek automatikus megtanulására anélkül, hogy szükség lenne felügyelt tanításra vagy az adatok címkéinek előzetes ismeretére. Alapvetően két részből állnak: az encoderből, amely az adatokat egy alacsony dimenziójú vektorban reprezentálja, és a decoderből, amely visszaalakítja ezt a kódolt reprezentációt az eredeti adatok formájába. Ez a szerkezet lehetővé teszi az autoencoderek számára, hogy saját magukat képezzék tanítás során, és az adatokban rejlő mintázatokat, jellemzőket és információkat automatikusan feltárják. Számos alkalmazási területen bizonyultak sikeresnek. Az adattömörítés terén például az Autoencoderek hatékonyan csökkentik az adatok dimenzióit, minimalizálva ezzel az adatok tárolásához vagy továbbításához szükséges erőforrásokat. A generatív Autoencoderek és a variációs Autoencoderek lehetővé teszik új adatok generálását, amelyek hasonlóak az eredeti

adatokhoz. Széles körben felhasználhatók a képek, hangok vagy más tartalmak generálására. Emellett a zajszűrő Autoencoderek hatékonyak zajos adatok tisztítására, mivel a tanítás során megtanulják a zajt az adatokból eltávolítani. Az Autoencoderek sokféle adattípussal is kompatibilisek. A konvolúciós Autoencoderek például kiválóan alkalmazhatók képadatok esetén, mivel képesek megőrizni a képek térbeli szerkezetét és részleteit. Az Autoencoderek tehát nemcsak a mélytanulás alapjait képezik, hanem számos alkalmazási területen hatalmas lehetőségeket kínálnak az adatfeldolgozás és a gépi tanulás terén is.



2. ábra. Zajszűrő Autoencoder: az ábrán szemléltetve látható, ahogy az autoencoder rekonstruálja az eredeti adatokat, minimalizálva a zajokat és kiemelve az eredeti kép fontos jellemzőit. [ábra forrása]

A zajszűrő Autoencoder olyan megközelítés, amely alkalmazható a zajos adatok megtisztítására és az információk kiemelésére. Ebben az architektúrában az encoder fontos szerepet játszik, mivel a zajos bemeneti adatokat kap meg. Az encodernek a feladata, hogy ezen zajos adatokat úgy alakítsa át, hogy egy olyan rejtett, látens reprezentációt hozzon létre, amely kevésbé zajos és tartalmazza az adatok valódi, lényeges jellemzőit. Ez a kód tehát magában foglalja azokat a mintázatokat és információkat, amelyek valóban fontosak a bemeneti adatok szempontjából, miközben minimalizálja a zaj jelenlétét. A decoder rész ezután a kódot visszaalakítja az eredeti adatok formájába. Ennek eredményeképpen a zajok eltávolításra kerülnek, és a kimeneti adatok tisztábbak és közelebb állnak az eredeti zajmentes változathoz. Az ilyen architektúra széles körben alkalmazható például képfeldolgozásban, ahol a zajszűrő Autoencoder segíthet az életlen vagy zajos képek javításában. Az eljárás alkalmazható továbbá hangfelismerésben és más szenzoros adatok tisztításában is. A módszer hozzájárulhat az adatok jobb minőségű és megbízhatóbb elemzéséhez és feldolgozásához, ami különösen fontos az olyan alkalmazásokban, ahol az adatok minősége kritikus szerepet játszik.

A felépítése a következőképpen néz ki a [7] forrás alapján. Legyen $P(X)$ az adat által generált eloszlás, véletlen X mintából. Legyen C egy meghatározott corruption folyamat, amely sztochasztikusan egy X-t $\widetilde{X}$-re képez le a $C(\widetilde{X}|X)$ feltételes eloszlás segítségével. Ez a folyamatot zajként használható a bemeneti adatok zajosítására a tanulási folyamat során. A zajszűrő Autoencoder tanítási adata olyan (X,$\widetilde{X}$) párok halmaza, ahol $X \sim P(X)$ és $\widetilde{X} \sim C(\widetilde{X}|X)$. Az Autoencoder úgy van tanítva, hogy X-t jósolja meg $\widetilde{X}$ alapján egy megtanult $P_\theta(X|\widetilde{X})$ feltételes eloszlás segítségével, egy θ által indexelt eloszláscsaládból választva. A tanítási folyamat úgy fogalmazható meg, hogy az X becslését tanulja meg adott $\widetilde{X}$ alapján, regularizált maximum likelihood alapján, azaz az általános teljesítmény amit a tanítás minimalizálni próbál a

$$L(\theta) = -\mathbb{E}[\log P_\theta(X|\widetilde{X})], \tag{4}$$

ahol a várható érték a közös adatgeneráló eloszláson van számolva

$$P(X, \widetilde{X}) = P(X)C(\widetilde{X}|X). \tag{5}$$

2.2. U-Net

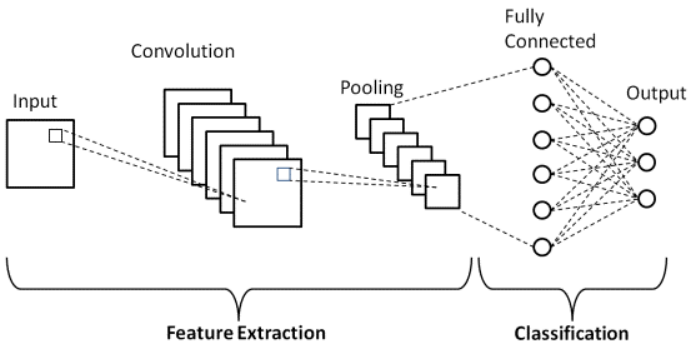
Az U-Net hálózatot először Olaf Ronneberger, Philipp Fischer és Thomas Brox mutatta be 2015-ben egy cikkben [8], amely alapján a következő fejezet íródott.
Az U-Net egy hatékony konvolúciós neurális hálózat (CNN) architektúra, amelynek a felfedezése jelentős áttörést hozott a szegmentációs feladatok terén a gépi tanulásban.
A konvolúciós rétegek matematikai működése a következőképpen írható le

$$Y_{i+1} = f(W_i Y_i + b_i), \tag{6}$$

ahol Y_i a bemeneti jellemző tér, W_i a konvolúciós szűrőmátrix, b_i a torzítás, azaz a réteg eltolása és f az aktivációs függvény, ami az egyes neuronok kimeneti aktivitását szabályozzák a neurális hálózatkban.
A dekódoló részében gyakran találunk transzponált konvolúciós rétegeket, amelyek a következőképpen működnek

$$Y_{i+1} = f(W_i^T Y_i + b_i), \tag{7}$$

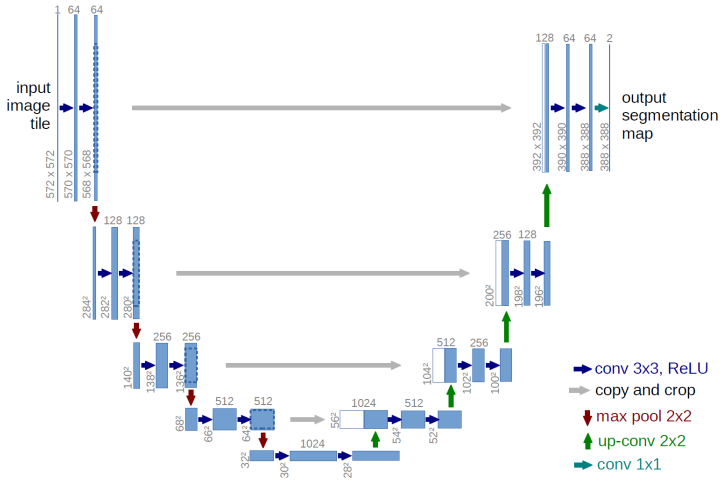
ahol W_i^T a transzponált szűrőmátrix.



3. ábra. Konvolúciós neurális hálózat (CNN): Az eredeti képet a modell bemenetként (*input*) fogadja el. A konvolúciós rétegek (*convolution*) kis szűrőkkel dolgoznak, amelyeket az egész képen mozgatnak, felismerve alacsony szintű jellemzőket, mint például élek vagy szögek. A *pooling* rétegek csökkentik a térbeli dimenziókat, maximumkiválasztással kis ablakokon belül, csökkentve a paraméterek számát és növelve a hatékonyságot. A teljesen összekapcsolt rétegek (*fully connected layers*) magasabb szintű jellemzőket dolgoznak fel. A kimeneti (*output*) rétegek adják a végső eredményt, mint például osztályozás vagy regresszió eredménye. [ábra forrása]

Az U-Net hálózatban az U alakú elrendezését az architektúrájában két rész jellemzi: a kódoló és a dekódoló szoros együttműködése, valamint a reziduális kapcsolatok a két rész között. Az U-Net architektúra kódoló része elsősorban a kép absztrakt reprezentációinak kinyerésére szolgál. Az először érkező bemeneti képet a konvolúciós rétegek sorozatán keresztül fokozatosan mélyebb szintű absztrakt reprezentációvá alakítja. Ennek során a térbeli információ részben elveszhet, azonban a tartalmi absztrakció folyamatosan növekszik. A kódoló rész végeredménye az úgynevezett látens tér, amely a bemenet mélyebb, gazdagabb reprezentációját tartalmazza. A dekódoló rész a mélyebb jellemzőket alakítja vissza eredeti térbeli dimenziójukba a transzponált konvolúciós rétegek segítségével. Ez a rész nagymértékben hozzájárul a szegmentációs feladatokhoz, mivel rekonstruálja a mélyebb rétegekben elvesztett térbeli információt. Emellett a reziduális összekötés is segít az információ továbbításában a kódoló és dekódoló között, így a hálózat könnyebben képes összekapcsolni a finom részleteket a kódolási folyamatból a dekódolási folyamatba.
Az U-Net forradalmi megközelítése az, hogy egyesíti az erőteljes szegmen-

tációs képességeket a konvolúciós hálózatok mélységével és az újrakapcsoló rétegekkel, amelyek lehetővé teszik az adatok helyreállítását a reprezentációk során elveszett részletekkel együtt. Ez az architektúra nemcsak a biomedicinális képfeldolgozásban talált széles körű alkalmazást, hanem olyan területeken is, mint az önvezető járművek környezetfelismerése, objektumdetektálás, orvosi képfeldolgozás és sok más terület a gépi tanulásban. Az U-Net egyike azon architektúráknak, amelyek megnyitották az utat a mély tanulás alkalmazásainak széles köréhez a gyakorlatban.



4. ábra. U-Net: az ábra bemutatja az U alakú szerkezetet, amely magában foglalja a bal oldali kódoló és a jobb oldali dekódó részeket. A kódoló rész progresszíven csökkenti a bemeneti képek méretét és növeli a jellemzők számát. A dekódó rész a kódoló részhez hasonlóan lépcsőzetesen növeli a látenstér méretét, miközben visszaállítja az eredeti kép méreteket. [ábra forrása]

A mély neurális hálózatok hatékony tanításához elengedhetetlen az adatnövelés alkalmazása, különösen akkor, amikor a rendelkezésre álló tanító minták száma korlátozott. A mikroszkópos képek elemzése során különösen fontos az eltolás- és forgásinvariancia, valamint a deformációk és szürkeértékváltozások ellenállósága. A rugalmas deformációk alkalmazása a tanító mintákra kulcsfontosságú a szegmentáló hálózat hatékony kiképzéséhez, különösen akkor, amikor csak néhány kép áll rendelkezésre. Ezeket a deformációkat olyan módon hozzuk létre, hogy véletlenszerű eltolási vektorokat alkalmazunk

egy 3x3-as rácsban, és a vektorok eltolásait egy Gauss-eloszlásból mintavételezzük. Ez a módszer lehetővé teszi a hálózat számára, hogy megtanulja az invariancia tulajdonságokat és a robosztusságot, anélkül, hogy sok képre lenne szükség. A kiképzés során alkalmazott kihagyási rétegek tovább növelik az adatok változatosságát és az általánosíthatóságot. Az így tanított hálózat képes lesz hatékonyan szegmentálni és elemezni képeket.

3. Diffúziós modellek

A diffúzió és a mély tanulás kapcsolata egyre növekvő figyelmet kap a gépi tanulás világában. A diffúziós alapú mély tanulási módszerek olyan technikákká váltak, amelyek a gépi látás számos területén kiemelkednek, beleértve a képgenerálást, szegmenetációt, objektumfelismerést és képinterpolációt. A diffúzió lényege a kép zajosítása, és segít a bonyolult eloszlásokat lépésről lépésre egyszerűbb eloszlássá alakítani. Ezt követően megpróbáljuk visszafordítani ezt a folyamatot, de sajnos a fordított folyamat eredménye az eredeti adateloszlástól függ, így a visszafordított folyamatot becsülnünk kell. Ennek eredményeképpen a diffúziós modellek képesek magas minőségű, valósághű adatok generálására. Ezek a modellek különböző alkalmazásokban bizonyították hatékonyságukat. A képgenerálás terén a diffúziós modellek lehetővé teszik valósághű fotók és képek generálását, amelyek gyakorlatilag megkülönböztethetetlenek az ember által készített képektől. Emellett alkalmazhatók hanggenerálásban, ahol magas minőségű zenei darabok vagy beszédhangok létrehozását teszik lehetővé.

A diffúziós modellek látens modellek. A látens elnevezés a modell által kódolt rejtett információkra utal, amelyek nem közvetlenül megfigyelhetők, de az adatok generálásához vagy értelmezéséhez kulcsfontosságúak. Legyenek $x_1, \ldots, x_T$ látensek, ugyanolyan dimenzióval mint az adatok $x_0 \sim q(x_0)$. Ekkor a modell

$$p_\theta = \int p_\theta(x_{0:T}) dx_{1:T} \tag{8}$$

alakban adható meg. Az $x_{0:T}$ egy időtartományra vonatkozó változót jelöl, ahol 0 a kezdeti időpontot, T pedig a végpontot reprezentálja. Tehát a $p_\theta(x_{0:T})$ azt fejezi ki, hogy a valószínűségi eloszlás a T időpillanatig terjedő összes x változót tartalmazza a modellparaméterek szerint.

3.1. Denoising Diffusion Probabilistic Model (DDPM)

A DDPM egy olyan keretrendszer, amelynek célja a kezdeti, zajos adatok fokozatosan történő tisztítása, mindezt egy valószínűségi modell segítségével. Ez azt jelenti, hogy a modell nem csak a zajtalanításra összpontosít, hanem a zajok fokozatos kiküszöbölése során felépíti az adatok valószínűségi eloszlását. A DDPM ezen tulajdonsága lehetővé teszi számára, hogy nagy mértékű adaptációt és rugalmasságot nyújtson különböző adattípusok és alkalmazá-

sok esetén, beleértve a képfeldolgozást, generatív modellezést és zajtalanítást is. A modell leírásához a [1] forrás szolgált alapul.
Az **előre folyamat** egy olyan Markov-lánc, amely lépésről lépésre hozzáad Gauss-eloszlásból származó zajt az adathoz. Ez a folyamat az x_0 bemeneti adaton kezdődik, és fokozatosan adja hozzá a zajt a modell paraméterei által meghatározott rendben. A zaj hozzáadási ütemterv legyen $\beta_1, \ldots, \beta_t$. Az egyes lépések során a zajszint általában nő, vagyis a kezdeti lépésekben a zaj kisebb, majd fokozatosan növekszik. Ez azért fontos, mert lehetővé teszi az adatok fokozatos felszabadítását, hogy az eredeti információk egyre jobban előkerüljenek a zajból. Az ütemterv egy másik fontos szerepet is betölt, úgy kell beállítani, hogy az eloszlása megközelítőleg normális legyen. Ez fontos szerepet játszik a diffúziós folyamatban, biztosítva, hogy a zajosított adatok végül egy megközelítően normális eloszláshoz konvergáljanak, ezáltal elősegítve az eredeti információk hatékonyabb visszanyerését. Az előre folyamat célja az adatok diffúziós alapú transzformációja, melynek végpontja az eredeti adateloszlás. Ezt követően lehetőség van arra, hogy a zajjal terhelt adatokból visszafelé járjuk vissza a diffúziós folyamatot, és elérjük az eredeti, zajmentes adatokat. A diffúziós modelleket más látens modellektől az különbözteti meg, hogy az előre folyamatuk vagy más néven diffúziós folyamatuk ($q(x_{1:T}|x_0)$) egy Markov-lánc.

$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1}), \tag{9}$$

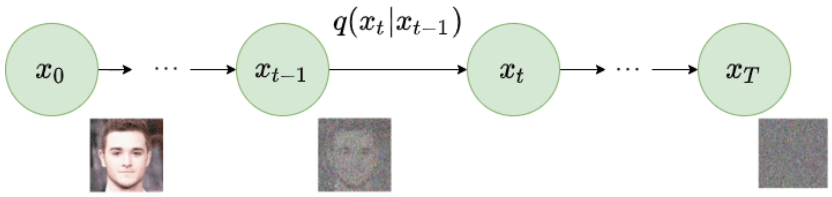
ahol

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t} \cdot x_{t-1}, \beta_t \cdot \mathbf{I}). \tag{10}$$

A $q(x_{1:T}|x_0)$ állítás szerint a q folyamatot ismételten alkalmazzuk az időlépések során 1-től T-ig.
Az adatokat először zajosítjuk a q-val, miközben a β-kat úgy állítjuk be, hogy T lépés után megközelítőleg normális eloszlást érjünk el. A reprezentáció folyamán lehetőség van zárt formátum használatára a mintavételezés során bármely időpillanatban. A reprezentáció során használhatunk egy zárt formát a mintavételezéshez bármely időpillanatban. Definiáljuk $\alpha_t = 1 - \beta_t$, $\bar{\alpha_t} = \prod_{s=0}^{t} \alpha_s$ és $\varepsilon_0, \ldots, \varepsilon_{t-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$. Ekkor

$$
\begin{aligned}
x_t &= \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\varepsilon_{t-1} \\
&= \sqrt{\alpha_t}x_{t-2} + \sqrt{1-\alpha_t}\varepsilon_{t-2} \\
&= \dots \\
&= \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}}\varepsilon_0.
\end{aligned} \tag{11}
$$

Az x_t adat előállításához a következő eloszlást használhatjuk: $x_t \sim q(x_t|x_0) = \mathcal{N} = (x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I})$.



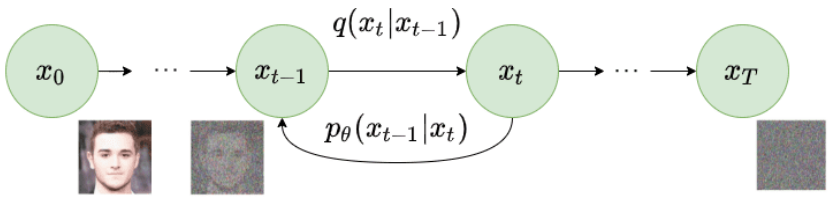
5. ábra. Az előre diffúziós folyamat ábráján látható, hogy a rendszer időbeli fejlődése során zaj hozzáadódik a diffúziós folyamathoz Markov-lánc segítségével. [ábra forrása]

A **vissza folyamat** ($p_\theta(x_{0:T})$) a diffúziós modellek alkalmazásában fontos szerepet játszik a zajjal terhelt adatok visszaalakításában az eredeti, zajtól mentes változatokká. Ebben a kontextusban a vissza folyamat célja az, hogy olyan modelleket hozzunk létre, amelyek megtalálják az eredeti adatokat a zajjal terhelt változatokból. A kifejezés egy olyan modellhez köthető, amelyben összefüggő adatokat generálunk Markov-lánc segítségével. Az átmenetek valószínűségeit tanított modell paraméterek irányítják, és ezek a valószínűségek Gauss-eloszlásúak. Az előrefelé folyamat és a visszafelé folyamat együtt alkotja a diffúziós modellt. A vissza folyamat kezdőpontja $p(x_T) = \mathcal{N}(x_T; \mathbf{0}, \mathbf{I})$. Legyen

$$
p_\theta(x_{0:T}) = p(x_T) \cdot \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t), \tag{12}
$$

ahol

$$
p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)). \tag{13}
$$



6. ábra. Az ábrán szemléltetve látható, hogy a kezdeti állapotból kiindulva a rendszer időbeli fejlődése során a diffúziós folyamatok révén a bemenet egyre inkább zajos lesz. A DDPM ezt követően vissza diffúzióval dolgozik, azaz a zajosított állapotokból visszavezeti a rendszert a kezdeti állapot felé Markov-lánc segítségével. [ábra forrása]

A tanítási folyamat az általános gyakorlat alapján zajlik. A modell a negatív **log-likelihood** függvény optimalizálásával tanul. Ez segít a modellnek alkalmazkodni az adatokhoz, valamint a generatív folyamat paramétereinek megtanulásához. Az optimális negatív log-likelihood korlát célja a modell által adott és a valós adatok közötti különbség minimalizálása, ezáltal lehetővé téve a modell számára, hogy pontosabb generatív folyamatot modellezzen.

$$\begin{aligned} \mathbb{E}[-\log p_\theta(x_0)] \leq \mathbb{E}_q \left[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)} \right] = \\ = \mathbb{E}_q \left[-\log p(x_T) - \sum_{t \geq 1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})} \right] =: L, \end{aligned} \tag{14}$$

ahol

$$\log p(x) = \sum_{t=1}^{T} \log p(x_t|x_{<t}) \tag{15}$$

Az effektív tanítás kulcsa a varianciaminimalizálás és a SGD kombinálásában rejlik. A varianciaminimalizálás lényege, hogy olyan módszereket alkalmazunk, amelyek minimalizálják a negatív log-likelihoodot, ehhez a Variation Upper Bound-t használjuk. Ezzel csökkentve a zajt és a változatosságot a tanulási folyamatban. Ennek eredményeképpen a tanítás stabilabbá és hatékonyabbá válik, mivel az optimalizáció során a gradiensek megbízhatóbbak lesznek. A sztochasztikus gradiens csökkentés pedig lehetővé teszi, hogy a

modellt az algoritmusban batchek felhasználásával több adatponttal tanítsuk anélkül, hogy az összes adatpontot egyszerre feldolgoznánk, ami nagymértékben gyorsítja a tanulási folyamatot. Ezen módszerek kombinációjával a gépi tanulásban alkalmazott diffúziós modellek hatékonyan képesek optimalizálni a negatív log-likelihood-et, és így megtanulni a valószínűségi eloszlásokat, amelyek alapján az adatok generálódnak. Ezek alapján a 14 egyenletet újraírva a következő modellt kapjuk

$$\mathbb{E}_q \left[L_T + L_{t-1} + L_0 \right], \tag{16}$$

ahol

$$\begin{aligned} L_T &= D_{KL}(q(x_T|x_0)||p(x_T)) \\ L_{t-1} &= \sum_{t=2}^{T} D_{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)) \\ L_0 &= -\log p_\theta(x_0|x_1) \end{aligned}$$

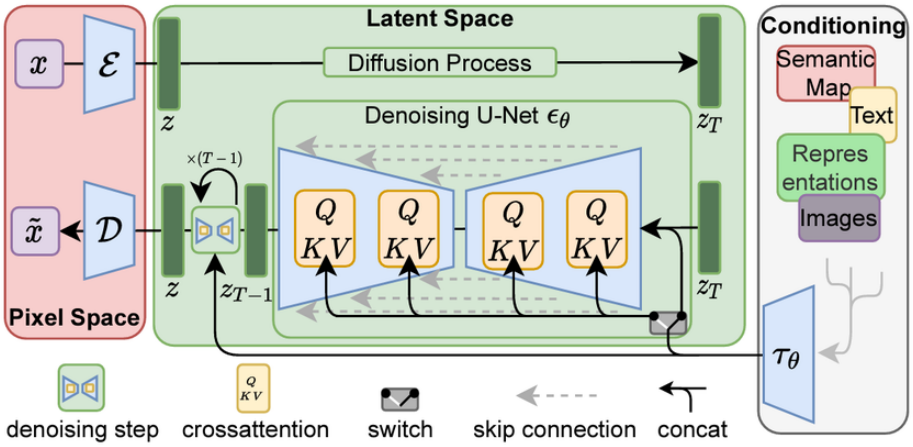
3.2. Látens diffúziós modell (LDM)

Az alábbi szakasz a [5]-ös forrásból származó elméletek és adatok alapján készült.

A látens modellek egy egyforma súlyú Autoencoderként vannak reprezentálva $\varepsilon_\theta(x_t, t)$, amik arra vannak tanítva, hogy előrejelezzék a bemenet zajmentes változatát.

$$L_{DM} = \mathbb{E}_{x,\varepsilon,t} \left[||\varepsilon - \varepsilon_\theta(x_t, t)||_2^2 \right], \tag{17}$$

ahol $\varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$ és $t = \{1, \ldots, T\}$. Fontos megjegyezni, hogy az ε_θ egy előre tanított encoder és decoder párost foglal magában, amelyek felelősek az ε zaj hozzáadási folyamat modelljéért és az inverz folyamatért, mely a zajmentes bemenet előrejelzéséért felel.
A látens reprezentációk generatív modellezése közben kifejlesztett tömörítő modell úgy épül fel, hogy egy $\mathcal{E}$ (Encoder) egy $\mathcal{D}$ (Decoder), valamint egy U-Net részekből áll. Ezek a modellek új szintre emelik a látens tér fogalmát, létrehozva egy hatékony, alacsony dimenziójú teret, ahol a magas frekvenciájú, néha észrevehetetlen részletek kerülnek megfigyelésre. Ennek köszönhetően, összehasonlítva a hagyományosan magas dimenziójú térrel, a látens tér egy olyan környezetet teremt, amely különösen alkalmas a valószínűség alapú



7. ábra. A látens diffúziós modell ábrájáról látható, hogy a bemeneti kép először áthalad egy encoderen, majd elindul a diffúziós folyamat, amely kialakítja a látens tér struktúráját. A következő lépésben egy zajszűrő U-Net kerül alkalmazásra a látens térbeli reprezentáción, amely eltávolítja a zajokat, ami a diffúzió során felmerülhet. Az U-Net után a látens térben a conditioning szakaszban egy olyan folyamat történik, amikor a tanítás során egy adott információt vagy jellemzőt hozzárendelünk a modell bemenetéhez annak érdekében, hogy tanuljon az adott környezeti vagy kontextuális feltételekről. Utolsó lépésben a feldolgozott látens tér a decoderre irányul, ahol a kimeneti kép kialakul. [ábra forrása]

generatív modellek számára. Ebben a térben a generatív modellek képesek finomhangolni, kiemelve az adatok lényeges részeit. Emellett az alacsony dimenziójú látens tér lehetővé teszi a modellek számára, hogy hatékonyabban tanuljanak, miközben csökkentik a számításigényt, ami az informatikai erőforrások hatékonyabb felhasználását eredményezi. Ezáltal a generatív modellek pontosabban modellezhetik és reprodukálhatják az adatokat, figyelembe véve azok rejtett tartalmát, és mindezt alacsonyabb dimenziójú térben tehetik meg. Ez a fejlett módszer nemcsak a látens reprezentációk terén hoz létre új lehetőségeket, hanem elősegíti a hatékonyabb és tartalmasabb generatív modellek fejlődését, amelyek széles körben alkalmazhatók.
A modell fejlesztésének során kiemelkedő hangsúlyt helyezünk az alapvető U-Net struktúrájának kialakítására, amelyet elsősorban két dimenziós konvolúciós rétegek alkotnak. Ezen építőelemek kifinomult alkalmazása lehetővé

teszi, hogy a modell a képek szempontjából releváns részletek felismerésére összpontosítson. A két dimenziós konvolúciós rétegekkel és a súlyozott kötéssel támogatott modellünk így jobban megfelel a gyakorlati alkalmazásoknak, különösen a képalkotás és észlelés terén, ahol a hangsúly a látens térben rejlő lényeges részletekre helyeződik.

$$L_{LDM} = \mathbb{E}_{\varepsilon(x),\varepsilon,t}\left[||\varepsilon - \varepsilon_\theta(z_t, t)||_2^2\right] \tag{18}$$

A modell gerince az $\varepsilon_\theta(\circ, t)$, ami egy idő együttes U-Net. Ez a kialakítás lehetővé teszi a modell számára, hogy időben változó információkat integráljon a látens reprezentációkba. Az előre folyamat állandó, ami azt jelenti, hogy a z_t látens változó előállítható az ε alkalmazásával a tanítási folyamat és a diffúziós lépések során. A látens térben lévő z_t értékei olyan rejtett jellemzőket és mintákat kódolnak, amelyek alapján a modell képes a generatív folyamatokat időbeli kontextusban értelmezni. A $p(z)$ eloszlásból származó minták dekódolása során a $\mathcal{D}$ modell egyetlen áthaladással képes visszafejteni ezeket a látens reprezentációkat a képtérbe. Ez a megközelítés nemcsak hatékony, hanem lehetővé teszi a modell számára, hogy időfüggő információkat tanuljon meg és hasznosítson a generatív folyamat során.

3.3. Denoising Diffusion Implicit Model (DDIM)

A Denoising Diffusion Implicit Model, azaz Zajszűrő Diffúziós Implicit Modell és a Denoising Diffusion Probabilistic Model szoros kapcsolatban állnak, a két modell tanítása folyamata megegyezik és mindkettő közvetlenül hozzájárul a Látens Diffúziós Modellhez. Mindhárom modell azon alapul, hogy a diffúziós folyamatokat alkalmazzák képek látens térbeli reprezentációjának kialakításához, de eltérő módszereket alkalmaznak. A DDIM és a DDPM közötti különbség az alkalmazott generatív struktúrában és a zajmodell megközelítésében rejlik. A DDPM explicit módon modellezi a zajos képek valószínűségi eloszlását és használja a diffúziós folyamatokat a zajos változatok generálásához. Ezzel szemben a DDIM implicit módon alkalmazza a diffúziós folyamatokat egy generatív hálózaton keresztül, ami segít zajos képek előállításában. A LDM a diffúziós folyamatokat használja a látens tér struktúrájának kialakításához, és mindkét módszer struktúráját integrálja a modelljébe.
A továbbiakban a bekezdés során a [2] forrásban található koncepciókat és eredményeket veszem alapul.

A modell lényege az implicit tanulás, ahol a zajszűrés folyamatát egy látszólag véletlen zajjal végezzük, és a modell tanulja meg, hogy az adatok hogyan néznek majd ki a zaj hozzáadása előtt és után. Az implicit tanulás azt jelenti, hogy a modell valójában nem generál explicit valószínűségi eloszlást az adatokra, hanem közvetett módon tanulja meg azokat. A DDIM tehát az előrefelé diffúziós folyamat és a vissza folyamat használatával képezi le az adatokat a zajjal terhelt és zajtól mentes állapotok között, ami lehetővé teszi a zaj eltávolítását a generált adatokból. Az x_{t-1} mintát generáljuk az x_t adat alapján a 12 egyeneletből. Legyen

$$x_{t-1} = \sqrt{\alpha_{t-1}} \underbrace{\left(\frac{x_t - \sqrt{1-\alpha_t}\varepsilon_\theta(x_t)}{\sqrt{\alpha_t}} \right)}_{\text{előrejelzett } x_0} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \varepsilon_\theta(x_t)}_{x_t \text{ iránya}} + \\ + \underbrace{\sigma_t \varepsilon_t}_{\text{véletlen zaj}} , \tag{19}$$

ahol $\varepsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ standard Gauss-eloszlásból származó zaj és $\alpha_0 := 1$. Amikor $\sigma_t = \sqrt{\frac{1-\alpha_{t-1}}{1-\alpha_t}} \cdot \sqrt{\frac{1-\alpha_t}{\alpha_{t-1}}}$ minden t-re, akkor az előre folyamat Markov-tulajdonságú és a generatív folyamat DDPM lesz. Egy speciális esete amikor $\sigma_t = 0$ minden t-re akkor az előre folyamat determinisztikus lesz adott x_{t-1} és x_0-ra, kivéve $t = 1$-t és ε_t véletlen zaj együtthatója 0 lesz. Ez a modell egy implicit valószínűségi modell, ahol a minták látens változókból vannak generálva, ezt nevezzük zajszűrő diffúziós implicit modellnek.

A DDIM előnyei számos területen megnyilvánulnak. A DDIM alkalmazható képjavítási feladatokban, például zajos képek zajmentesítésére. A modell képes a zajos képek restaurálására, javítva a képek minőségét és részleteit. Ennek gyakorlati alkalmazásai sokfélék lehetnek, ideértve az orvosi képalkotást, a fotóretusálást és az eszközellenőrzést. A DDIM segítségével javítható a diagnosztikai pontosság és az eszközök hatékonysága. A generatív modellek területén is kiemelkedően teljesít. A DDIM segítségével lehetőség nyílik olyan tartalmak létrehozására, amelyek hűen tükrözik a valóságot, miközben kreatívak és újszerűek. Ez a tulajdonság különösen fontos azoknak, akik művészetben, tervezésben és tartalomgenerálásban tevékenykednek, mivel lehetővé tesz valódi, inspiráló és kreatív tartalmak előállítását.

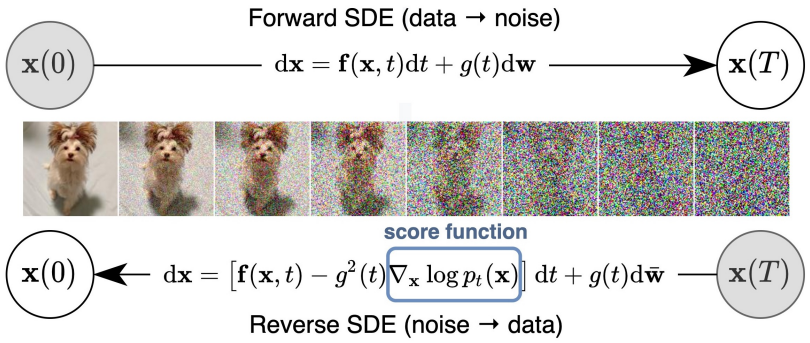
3.4. Score Based Generative Model (SBGM)

A Denoising Diffusion Probabilistic Model és a Denoising Diffusion Implicit Model diffúziós folyamatot alkalmaznak az adatok generálásához. A DDPM expliciten modellezi a zajosított adatok valószínűségi eloszlását, míg a DDIM impliciten generálja a zajosított adatokat. A Látens Diffúziós Modell hasonlóképpen diffúziós folyamatot használ, de a látens térben. A Score Based Generatív Modellekkel való kapcsolat abban rejlik, hogy minden megközelítésben diffúziós folyamatot alkalmaz a generatív folyamatok irányításához. A SBGM által számolt gradiensek hatékonyan használhatóak az adatok generálásának finomhangolására, míg a diffúziós folyamat segít a látens térbeli reprezentációk és az eredeti adatok közötti kapcsolat megteremtésében.
A [9] forrás által nyújtott nézőpontokat és kutatási eredményeket használva készült el ez a bekezdés.
A Score Based Generative Model olyan generatív modell kategóriája, amelyek újfajta megközelítést alkalmaznak az adatgenerálásban. Ezek a modellek a generálási folyamat során nem közvetlenül az adatokat hozzák létre, hanem egy adott adathoz számolnak ki egy pontszámot vagy gradienst. Ezen pontszámok alapján hozzák létre az új adatokat, figyelembe véve a gradienseket a generálás folyamatában. Egyik érdekessége az, hogy ezek a modellek tanulás útján képesek finomhangolni magukat. Ez azt jelenti, hogy a modellek a gradiensekből folyamatosan javítják a generált tartalmak minőségét, és képesek az adathalmazuk sajátosságaihoz alkalmazkodni. A modellek nem függnek előzetesen rögzített mintáktól vagy sablonoktól, ezért képesek valósághű és változatos adatok generálására. Számos területen alkalmazható, például képgenerálásban, nyelvi modellalkotásban és zenegenerálásban.

$$dx = f(x,t)\,dt + g(t)\,dw \tag{20}$$

A 20 egyenlet leírja, hogy az előre folyamatban lépésről lépésre viszi át az x adatok eloszlását egy normális eloszlásra zaj segítségével, egy sztochasztikus differenciál egyenlettel. Az adatok egy olyan x_t folyamatot alkotnak ahol a w_t a Brown-mozgás és x_0 a kezdeti bemeneti adateloszlás amihez iteratívan adja hozzá a zajt. Az időbeli változás egy lineáris csökkenési és egy véletlen zaj növekedési komponensből áll. A differenciál egyenlet megoldása egy Ornstein-Uhlenbeck folyamat, amelyet a következőképpen lehet reprezentálni

$$x_t = e^{-t}x_0 + \sqrt{1 - e^{-2t}}z, \text{ ahol } z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{21}$$



8. ábra. Az ábrán látható Score-Based Generative Model folyamat szemléltetése. A kezdeti értékből kiindulva az előrefelé haladó diffúziós folyamat generálja a zajos képet, míg a visszafelé haladó diffúziós folyamat a zajos képből visszaállítja az eredeti kezdeti értéket. Mindkét irányú folyamatot a Stein Score alapján van irányítva. [ábra forrása]

A vissza folyamatban az utolsó időpillanatban generált x_T adatot állítja vissza az eredeti x_0 eloszlásba. A modell meghatározza az x_{T-t} időpontban az x_T visszafejtéséhez szükséges deriváltat és a folyamat inverz módon végrehajtható

$$dx_{T-t} = \{x_{T-t} + 2\nabla \log p_{T-t}(x_{T-t})\}dt + \sqrt{2}dw_t, \tag{22}$$

ahol p_t az x_t valószínűségi sűrűsége, $\nabla \log p_t(x)$ a Stein score függvény, ami azt mutatja meg, hogy a log-likelihood hogyan változik, amikor a modell paramétereit módosítjuk, vagyis az adott időpillanatban az adat valószínűségi sűrűségfüggvényének gradiense.
A mintavételezés során egy numerikus sztochasztikus differenciálegyenlet megoldóját alkalmazzuk. A folyamatban, amikor az x_T-t visszafele időben nézzük, és megfelelően becsüljük meg az $\nabla \log p_t$ függvényt minden időpillanatban, akkor képesek vagyunk az x_0 kezdeti adatokat generálni. Ezt a folyamatot egy időben visszafelé menő sztochasztikus differenciálegyelten keresztül végezzük, amely összeköti a két időpontot. Ez a módszer lehetővé teszi a látens változók értékeinek következő időpontbeli becslését a megfigyelt adatok és gradienseik segítségével, így együttesen modellezve az időbeli folyamatot.

3.5. Mixture Gaussian Denoising Diffusion Model (MG-DDM)

A Mixture Gaussian Denoising Diffusion Model egy olyan modell, amely az előbb említett DDPM, DDIM, LDM és SBGM fontos elemeit integrálja, továbbfejlesztve azt. Ez a modell kiterjeszti a zajszűrő diffúziós folyamatot azzal, hogy több Gaus-függvényt alkalmaz a zajosító folyamatokra. Ezáltal a MG-DDM egyfajta keverékmodell, amely lehetővé teszi a bemeneti adatok komplex zajainak modellezését. Ötvözi a diffúziós modellek általános szerkezetét a zajszűrő folyamatokkal, miközben bevezeti a Gauss-függvények keverékét a zajosítások variabilitásának és összetettségének modellezésére. Ennek eredményeként a modell képes kezelni olyan adatokat, amelyek esetében a zaj nem feltétlenül követ egyetlen determinisztikus folyamatot, hanem több különböző variánsból származik.
A továbbiakban található információkat a [11]-es forrásból merítettem. A Vegyes Gaussi Zajszűrő Diffúziós Modell egy olyan komplex adatmodellezési eszköz, amely a statisztikai változók eloszlását a Gauss-függvények kombinációjával írja le, miközben zajszűrést is végez. Ezek a modellek az adatok sokféle tulajdonságát képesek reprezentálni, olyan változatos adathalmazoknál, ahol több különböző eloszlás és zaj is szerepet játszik. A Gauss-görbék keverékével ezek a modellek rugalmasan leképezik az adatok valószínűségi eloszlását, különböző varianciájú és eltolású görbékkel, miközben a zajt is csökkentik. Ezáltal a modellek segítenek a különféle statisztikai tulajdonságokkal rendelkező adatok komplex reprezentációjában, ami széles körű alkalmazást tesz lehetővé, különösen az olyan területeken, ahol az adatok összetettek és sokrétűek.
A 10 egyenlet átírható a következő alakra

$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\varepsilon_t, \tag{23}$$

ahol ε_t a Gauss-eloszlából származó zaj. Ha ehhez minden t lépésben hozzáadunk egy vegyes zaj eloszlást akkor általánosítható a következőképpen

$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}(\sum_{i=0}^{C} z_i\varepsilon_t^i), \tag{24}$$

ahol z_i véletlen bináris változó és C a változók száma. Legyen p_i egy valószínűség, hogy mekkora az esélye annak, hogy $z_i = 1$ az i-edik Gauss változó esetén.

$C = 2$ esetet vizsgálva olyan vegyes eloszlásokat kapunk, amelyeknek a várható értékük 0 és két Gauss-eloszlásból állnak, ugyanazzal a varianciával $(\phi_t)^2$ és valószínűséggel $p = 0.5$. Legyen

$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}(b\varepsilon_t^1 + (1-b)\varepsilon_t^2), \tag{25}$$

ahol $\varepsilon_t^1 \sim \mathcal{N}(m_t^1, (\phi_t)^2), \varepsilon_t^2 \sim \mathcal{N}(m_t^2, (\phi_t)^2)$ és $b \sim Bernoulli(p)$.

$$\begin{aligned} m_t^1 &= \sqrt{\frac{1-(\phi_t)^2}{p(1-p) + \frac{p^3}{1-p} + 2p^2}} \\ m_t^2 &= -\frac{p}{1-p}m_t^1 \end{aligned} \tag{26}$$

A hozzáadott zajt minden t lépésben újraparaméterezzük a következőképpen $\sqrt{\beta_t}X_t$ ahol $X_t = b\varepsilon_t^1 + (1-b)\varepsilon_t^2$. Mivel X_t várható értéke 0 és varianciája 1 ($\mathbb{E}[X_t] = 0, \mathbb{V}[X_t] = 1$) ezért minden lépésben a hozzáadott érték $\sqrt{\beta_t}X_t$ várható értéke 0 lesz és varianciája β_t.
Adott x_0 kezdeti mintából x_t zárt formája a következőképpen néz ki

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}N_t, \tag{27}$$

ahol $N_t \sim \mathcal{M}((\phi_t)^2)$ és ϕ_t egy hiperparaméter, amely a modell varianciáját jelöli minden t időpontban a folyamat során.
Ez a modell különösen hasznos lehet az adatok zajosításának szimulálására, valamint annak feltérképezésére, hogy az eredeti adatok hogyan transzformálódnak a diffúziós folyamat során. Az MG-DDM használata lehetővé teszi az adatok vizsgálatát és a különböző zajok hatásainak szimulálását, ami fontos lehet bizonyos folyamatok modellezése során, például a rendszerhibák, hibák becslése vagy a jel-zaj arány elemzése során.

3.6. Autoregressive Denoising Diffusion Model (ARDM)

Az Autóregresszíós Zajszűrő Diffúziós Modell és más generatív modellek, például a DDPM vagy a LDM, közötti kapcsolatok megértéséhez érdemes a generatív modellezés néhány alapvető aspektusát és az egyes modellek jellemzőit szemügyre venni.
Az autóregresszíós diffúziós modell egy olyan eljárás, amely az autoregresszió és a diffúzió koncepcióit kombinálja. Az autoregresszió azt jelenti, hogy a modell képes a jövőbeli állapotait a jelenlegi állapotok alapján becsülni és előre

jelezni. Ezzel szemben a diffúziós folyamatok olyan matematikai modelleket jelölnek, amelyek leírják egy rendszer véletlenszerű változásait az időben. Az összekapcsoláslényege az, hogy az ARDM felhasználja a DDPM és az LDM által kifejlesztett diffúziós folyamatokat a zajosítás és zaj eltávolítás során. Az autóregressziós diffúziós modellek ez a két elv kombinációjával rendkívül hatékonyan képesek modellezni és előre jelezni olyan folyamatokat, amelyeknél az időbeli függőségek és a véletlenszerű változások egyaránt fontosak. A modellek további előnye, hogy lehetőséget nyújtanak a rendszer komplexitásának és a folyamatok bonyolultságának kezelésére. A modellek finomhangolhatók és alkalmazhatók olyan kihívó feladatokra is, ahol a hagyományos statisztikai modellek vagy más gépi tanulási módszerek kevésbé lehetnek hatékonyak.
Ezáltal az autóregressziós diffúziós modellek széles körű alkalmazási lehetőségekkel rendelkeznek, például pénzügyi piacokon, ahol az árak és a kereslet napról napra változnak, a modellek segíthetnek a trendek azonosításában és a jövőbeli fejlemények becslésében. Emellett az egészségügyi területen is hasznosak lehetnek, például a járványterjedés modellezésében, ahol az időbeli változások és a véletlenszerű események befolyásolják a betegség terjedését és dinamikáját.
A következő bekezdés tartalmát a [12]-es forrás alapján dolgoztam ki.
A 15 log-likelihood egyenlet átírható a következő formába, ahol a random sorrendet jelölje $\sigma \in S_T$, S_T jelöli az összes permutációt $\{1, \ldots, T\}$ között és $\mathcal{U}(\circ)$ az uniform, azaz egyenletes eloszlást.

$$\begin{aligned}
\log p(x) &\geq \mathbb{E}_{\sigma \sim \mathcal{U}(S_T)} \sum_{t=1}^{T} \log p(x_{\sigma(t)} | x_{\sigma(<t)}) \\
&= \mathbb{E}_{\sigma \sim \mathcal{U}(S_T)} T \cdot \mathbb{E}_{t \sim \mathcal{U}(1,\ldots,T)} \log p(x_{\sigma(t)} | x_{\sigma(<t)}) \\
&= T \cdot \mathbb{E}_{t \sim \mathcal{U}(1,\ldots,T)} \cdot \mathcal{L}_t, \qquad (28)
\end{aligned}$$

ahol $\mathcal{L}$ reprezentálja a likelihood komponenst:

$$\mathcal{L}_t = \frac{1}{T-t+1} \mathbb{E}_{\sigma \sim \mathcal{U}(S_T)} \sum_{k \in \sigma(\geq t)} \log p(x_k | x_{\sigma(<t)}).$$

A modell eloszlás logaritmusának paraméterezése azon célból történik, hogy minden $k \in \sigma(\geq t)$ esetén, tetszőleges σ permutáció és t időpont mellett pontosan meghatározzuk a valószínűségi eloszlást. Az egyenként alkalmazott, különálló neurális hálózatok használata minden σ és t esetén rendkívül költséges lenne egy nagy méretű modell esetén. Ennek megkerülése érdekében

egyetlen neurális hálózatot alkalmazunk, amelyet megosztunk. Ezt a megosztást a bemeneten alkalmazott maszkolással érjük el, ahol a változókat egy Boole-függvénnyel maszkoljuk, ami az adott σ és t permutáció és időpillanat alapján határozza meg, hogy mely változókat kell figyelembe venni.

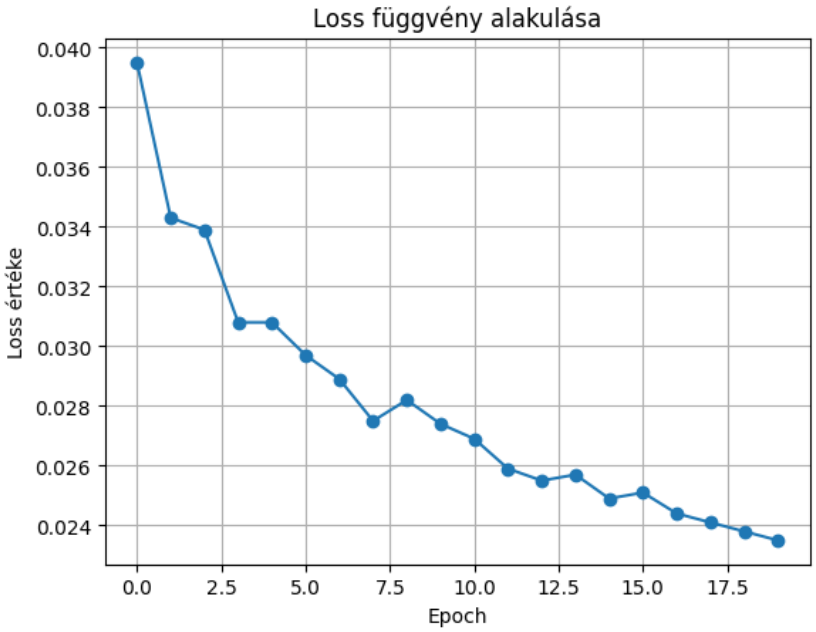
4. Alkalmazás

Ebben a fejezetben a diffúziós modellek alkalmazásainak bemutatására összpontosítok, kiemelve a Denoising Diffusion Probabilistic Model (DDPM) alkalmazását a MNIST és CIFAR-10 adathalmazon. A DDPM egy olyan generatív modell, amely diffúziós folyamatokat alkalmaz a zajosított adatok előállítására és ezekből a zajos adatokból való visszaállításra. A fejezetben kitérek arra, hogyan alkalmaztam a DDPM-t a feladatokra, hogyan ért el eredményeket a zajos és zajmentes képek generálásában, és milyen előnyöket nyújtott az MNIST és CIFAR-10 adathalmazokon történő alkalmazás során. Emellett bemutatom a modellezés során felmerülő kihívásokat és azokra nyújtott megoldásokat, valamint értékelem a modell teljesítményét az adott alkalmazási területeken.
A programkódok és a mérések eredményei a [GitHub oldalamon] találhatóak.

4.1. MNIST DDPM

A programkód elkészítéséhez az implementáció során a GitHubon található https://github.com/TeaPearce/ConditionalDiffusionMNIST/tree/main programkódot vettem alapul.
MNIST kód GitHub oldala
A program kód egy PyTorch-alapú implementációt mutat be a DDPM tanítására és mintavételezésére. A programot a Google által szolgáltatott Colaboratory interaktív környezetben, NVIDIA Tesla T4 GPU grafikus kártyával futtattam. A modell az MNIST képek tanítására van beállítva és a generált képek összehasonlítására az eredetihez képest FID metrikával.
Ez a kód egy mély generatív modellt definiál, amely a U-Net architektúrát és egy diffúziós zajszűrő valószínűségi modellt kombinál. A modell célja valósághű képek generálása. Nézzük meg a kódot különböző komponensekre bontva.
Az U-Net háló a kódban implementálva van, kiegészítve reziduális konvolúciós blokkokkal *ResidualConvBlock*, amelyek segítenek a hálózat mélyítésében és az információ hatékonyabb átvitelében. A lefelé mintavételező rétegek *UnetDown* a képek jelentős részét szűkítik, míg a felfelé mintavételező rétegek *UnetUp* segítik a részletes információk visszaállítását.
A *ContextUnet* modell a fő architektúrája, amely figyelembe veszi a kontextuális információkat a képek generálásához. Ez azt jelenti, hogy a modell

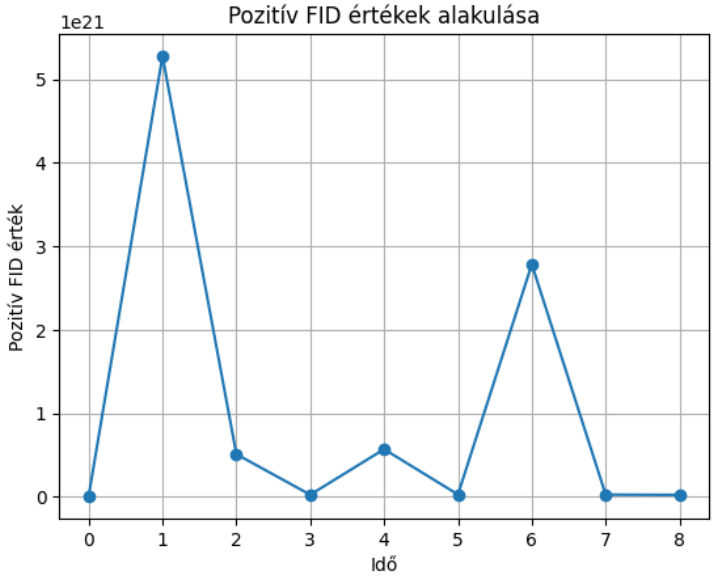
nem csupán a jelenlegi képet veszi figyelembe, hanem környezeti jellemzőket, összefüggéseket is bevon a folyamatba. Ennek eredményeként a generált képek tartalmazhatnak olyan információkat, amelyek a környezetükből erednek, lehetőséget teremtve így a valósághűbb és kontextusban gazdagabb képek előállítására. A *ContextUnet* így a kontextus fontosságát hangsúlyozza a generatív folyamat során.
A kódban DDPM egy olyan valószínűségi generatív modell, amely a U-Net architektúrát alkalmazza a generatív folyamat során. A DDPM osztály a modell két fő funkcionalitását kezeli: tanítást és mintavételezést. A tanítási részben a modellt egy rekonstrukciós veszteség alapján tanul, ahol az átlagos négyzetes hiba mértékét minimalizálják, ez a *loss* függvény. A rekonstrukciós veszteség azt mutatja, mennyire tér el a modell által generált kimenet az eredeti bemenettől.



9. ábra. MNIST esetén a loss függvény értékei

Ezen kívül a tanítás során alkalmazza a diffúziós folyamatot időben elhalványító ütemezési tervet is a tanulási folyamat finomhangolására. A *ddpm schedules* függvény által előre kiszámolt ütemezési terv segítségével a modell finomhangolja az időbeli elhalványítást, ami javíthatja a generált képek

minőségét és diverzitását. A DDPM modell egyesíti a diffúziós folyamatot és az U-Net architektúrát, és a két paradigma kombinációja lehetővé teszi a modellnek, hogy hatékonyan generáljon képeket a tanulási folyamat során. A kódban található w egy változó, amely a generatív irányítás erősségét jelöli a mintavételezés során. Azt szabályozza, hogy mennyire irányítsa a generatív modell a mintavételezést. Ha a w értéke 0, akkor nincs irányítás, ha 0.5 akkor a tanulás során a modell részben támaszkodik a generatív irányításra a zajos adatok javítása érdekében, ha $w = 2$ akkor a modell nagyobb mértékben támaszkodik az irányításra.



10. ábra. MNIST esetén a FID alakulása

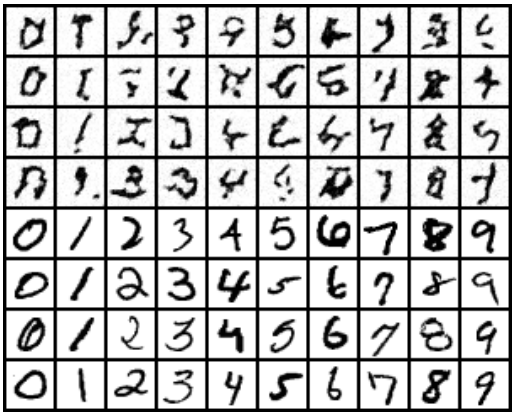
A Frechet Inception Distance (FID) az egyik elterjedt metrika, amelyet a generált képek minőségének és sokszínűségének értékelésére alkalmaznak, és összehasonlítja ezeket a valós képek minőségével. A Frechet-távolság ebben az értelmezésben a térbeli eloszlások közötti hasonlóságot méri. Ez a távolság kiterjeszti a klasszikus euklideszi távolság definícióját, kifejezetten két valószínűségi eloszlás közötti összehasonlításra alkalmazva. Legyen

$$FID(P, Q) = ||\mu_p - \mu_q||_2^2 + Tr(\Sigma_p + \Sigma_q - 2 \cdot \sqrt{\Sigma_p \Sigma_q}), \tag{29}$$

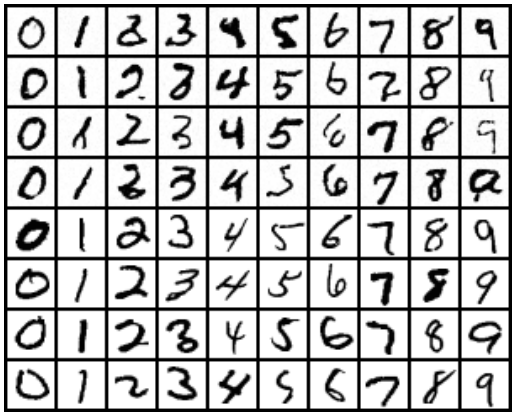
ahol a μ a perceptron kimeneteinek átlagai, a Σ a perceptron kimeneteinek kovarianciái és a Tr a mátrix nyomát jelöli. Ebben a kontextusban az aktivációk eloszlásait a kimeneti rétegben az ImageNeten előtanított Inception modell utolsó előtti rétegének aktivációinak eloszlásával van összehasonlítva. Az utolsó előtti réteg aktivációi a modell bemeneti képekre adott válaszainak statisztikai tulajdonságait reprezentálják
Az eredeti kódban nem szerepelt a FID metrika mérése, azonban azt észrevettem, hogy ez a metrika jelentős mértékben hozzájárulhat a generatív folyamat teljesítményének objektív értékeléséhez. Ezért döntöttem úgy, hogy implementálom a metrikát a kódba, annak érdekében, hogy további mélységben elemezhessem és pontosan meghatározhassam a generált képek minőségét. A mérést a generatív folyamat mindegyik epoch-jában végrehajtottam, lehetővé téve a FID metrika időbeli alakulásának nyomon követését. Ezáltal a fejlesztések, változások és finomhangolások hatásait értékelhettem, ami segített a generatív modell teljesítményének folyamatos optimalizálásában és fejlesztésében.
A *calculate fid* függvény a generált képek eloszlását és a valós képek eloszlását veszi alapul. Először számol egy diszkriminatív modellt mindkét eloszlásra, majd a Frechet-távolságot számolja ki a két eloszlás perceptron kimenetei alapján. Minél kisebb a FID értéke, annál hasonlóbbak a generált és valós képek eloszlásai, és így a generatív modell minősége magasabb. A kódom által kapott eredmények közül van ahol a modell FID számítás közben hibázott és negatív számot adott, ezeket az eredményeket nem vettem figyelembe.

A következő ábrákon bemutatásra kerül kettő kiragadott kép a generatív folyamatról. A megjelenített képek mind $w = 0$ értékkel lettek generálva. Képenként 8 sorból álló számok jelennek meg, amelyek közül a felső 4 sor a modell által generált, az alsó 4 sor pedig a valódi képek.

A modell által generált képek fejlődése jól látható szemmel is, ahogy az idő előre haladtával egyre kifinomultabbá válnak. Az inicializálást követően a generatív folyamat elindul, és a kimeneti képek fokozatosan javulnak és közelítik a valóságos mintákat. Az egyes iterációk során megfigyelhető, ahogy a model tanul az adathalmaz struktúrájáról, és ennek eredményeként a generált tartalom egyre élethűbb formákat ölt. Az időbeli fejlődés nyomán a modell képes finomítani a részleteket és textúrákat, így a generált képek egyre inkább hasonlítanak a valóságos elemekhez. Ezen folyamatok észrevehetően tükrözik



11. ábra. Epoch 0, w=0.0

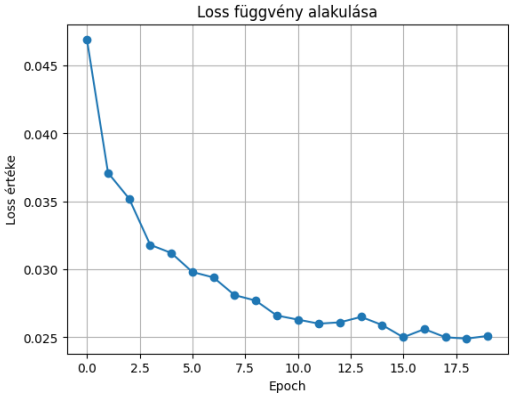


12. ábra. Epoch 19, w=0.0

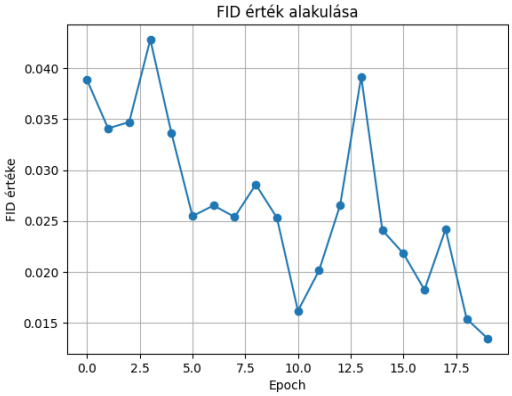
a modell tanulását és alkalmazkodását az adott adathalmaz sajátosságaihoz, létrehozva ezzel egy dinamikus és progresszív generatív folyamatot.

4.1.1. Fejlesztés

A 2. mérés GitHub oldala
A második mérésre azért volt szükség, mert ahogy említettem az előzőben a FID metrikát használva valamilyen számítási probléma lépett fel. Az összehasonlító metrikát továbbfejlesztettem a Wasserstein távolsággal.
A Wasserstein távolság vagy p-normájú távolság, egy metrika, amely méri a két valószínűségi eloszlás közötti különbséget vagy távolságot egy adott norma alapján. Gyakran alkalmazható két eloszlás közötti hasonlóság vagy különbség számszerű kifejezésére. A számítás az eloszlások közötti optimális átemelési terv alapján történik, ahol az átemelési költségek a két eloszlás pontjainak távolságát jelentik. A két eloszlás közötti optimális átemelési terv meghatározza, hogy milyen mértékben és módon kell átmozgatni a valószínűségi tömeget az egyik eloszlásról a másikra annak érdekében, hogy minimalizáljuk az átemelési költséget. A távolság kiszámításához a 2 eloszlásnak egyforma hosszúságúnak kell lennie. Ezt a kódomban úgy oldottam meg, hogy mindkét eloszlást lerövidítettem azonosra. A távolság számítására a *scipy stats* beépített modulját használtam.
A fejlesztett modell loss függvénye illetve a FID metrika eredményei az alábbi diagramokon láthatóak.
Az ábrák elemzése alapján megfigyelhető, hogy a veszteségfüggvény értékei



13. ábra. Loss



14. ábra. FID

az idő előrehaladtával egyenletesen csökkennek 0.045-ről 0.025-re. Az első pár epoch során magasabb értékeket mutat, ami arra utal, hogy a modell ezen időszak alatt tanul és alkalmazkodik az adathalmazhoz. A FID érték is átlagos csökkenést mutat, kezdeti 0.05-ről végül 0.015 alá süllyed. Ez azt sugallja, hogy a generált képek egyre jobban illeszkednek a valós képekhez. Bár a 10. és 13. epoch közötti időszakban megfigyelhető egy kis ingadozás az FID értékben amíg a loss továbbra is csökken, ami a túltanulás egyik jele lehetne, azonban ezután a modell tanulási folyamata helyreállt, és továbbra is hatékonyan javult. Ezáltal a modell tanulási folyamata stabilnak bizonyult, és a generált képek minősége folyamatosan fejlődött, ígéretes eredményeket hozva elő a további fejlesztések szempontjából.

4.2. CIFAR-10 DDPM

A programkód elkészítéséhez az implementáció során a GitHubon található https://github.com/pjborowiecki/COMP3547-Deep-Learning/tree/main kódot vettem alapul.
A CIFAR-10 adathalmaz, amelyet használok a modell felépítése során, széles körben alkalmazott adatkészlet, amelyet azért terveztek, hogy teszteljék és értékeljék a mély tanulási modellek teljesítményét. Az adathalmaz tartalmaz 60 000 színes képet tíz különböző kategóriából, ahol minden kategóriában 6000 kép található. A tíz osztály közé tartoznak olyan objektumok, mint repülőgépek, autók, madarak, macskák, szarvasok, kutyák, békák, lovak, hajók és kamionok. A CIFAR-10 egy kihívást jelentő adathalmaz, mivel a képek

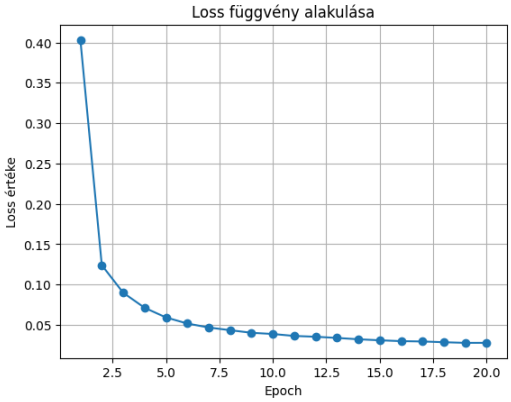
alacsony 32x32 pixel felbontásúak és gyakran zajosak. Az ilyen alacsony felbontás ellenére az adathalmazban szereplő képek változatosak és részletesek. Emellett a képek valós környezetben, természetes helyzetekben történő rögzítését szimulálva a CIFAR-10 ideális alapot kínál az olyan mély tanulási algoritmusok és modellek fejlesztéséhez, amelyek valósághű körülmények között tudnak eredményesen működni. A zaj és a változatosság a képeken tovább növeli a kihívásokat, hozzájárulva a modellek robosztusságának fejlesztéséhez.

A kód egy DDPM implementációt tartalmaz PyTorch keretrendszerben. A mérés célja a generatív modellezés területén való alkalmazás. Az eljárás zajszűrésen keresztül történő tanítással próbálja reprodukálni a bemeneti adathalmaz valószínűségi eloszlását, lehetővé téve a valósághűbb minták generálását. Az implementáció a DDPM struktúráját követi, amely magában foglalja az $Attention, Residual, DownBlock, UpBlock$ osztályokat. Az $EpsilonTheta$ osztály az DDPM fő architektúráját valósítja meg, mely a hiperparaméterekre és a modell rétegeire épül. A *DenoisingDiffusion* osztály a zajszűrést végző algoritmusokat, például a *forward diffusion* és *reverse diffusion* függvényeket tartalmazza.
Az epoch-okon keresztül figyelemmel kísérem a loss függvényeket és a generált képek minőségét. Az eredeti kódban nem szerepelt metrika a generált képek minőségének objektív értékelésére. Ennek hiányában úgy döntöttem, hogy a Frechet Inception Distance (FID) metrikát implementálom a generatív folyamat során keletkező képek minőségének értékelése érdekében. Ez lehetővé tette számomra, hogy objektív és mérhető eredményeket kapjak a generált tartalom minőségéről, hozzáadva ezzel további mélységet az értékeléshez.
A DDPM hatékonyan képes generálni valósághű képeket a zajszűrés segítségével. Azonban az optimalizáció és a tanulási folyamat finomhangolása további kutatást és kísérletezést igényelhet. Az esetleges továbbfejlesztési lehetőségek közé tartozik a hálózat mélységének növelése vagy a hiperparaméterek finomhangolása.

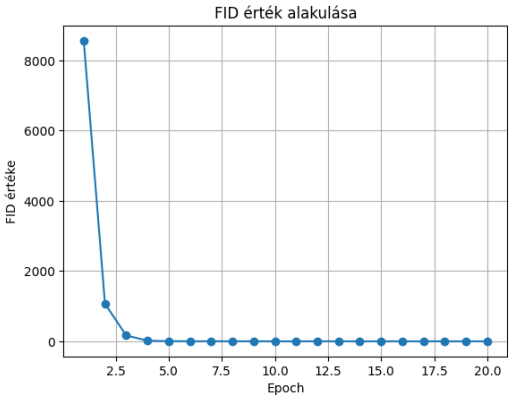
4.2.1. 1. mérés

Az 1. mérés GitHub oldala.
Az első mérést a Google által szolgáltatott Colaboratory interaktív környezetben, NVIDIA Tesla T4 GPU grafikus kártyával futattam, a következő

hiperparaméterekkel. A *Batch size* paraméter meghatározza, hány képet dolgoz fel a modell egyszerre a tanítás során. A *Image size* és a *Channels* beállítások meghatározzák a bemeneti képek méretét és a színcsatornák számát. A *Beta initial* és *Beta final* értékek a DDPM modellben használt beta paraméterek kezdeti és végső értékeit határozzák meg. Ezek a paraméterek befolyásolják az exponenciálisan csökkenő diffúziót a DDPM-ben, és fontos szerepet játszanak a modell tanításában. A *Feature map size* paraméter a hálózat belső rétegeiben alkalmazott karakterisztikus térképek számát szabályozza, míg a *Groups number*, a *Heads number*, és a *Blocks number* a GAU blokkok felépítését és számát befolyásolják. A *Learning rate* meghatározza, milyen mértékben módosítja a tanulási folyamat a hálózat súlyait. Az *Epochs* szám a teljes adathalmazon való átfutások számát határozza meg a tanulás során. Végül a T értéke az időpillanatok számát jelenti a DDPM diffúziós folyamatában, ami meghatározza, hány időlépésen keresztül terjed ki a diffúzió a képek generálása során.



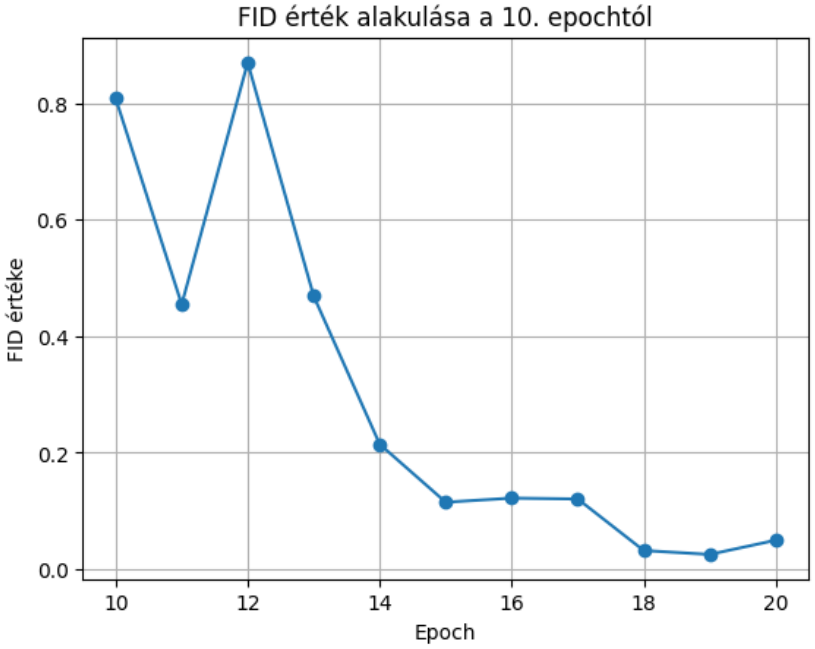
15. ábra. Loss



16. ábra. FID

Az első futtatás során elvégzett 20 epochot követően részletesen elemeztem a loss és a FID értékek alakulását. A loss függvény értékei az epochok során stabilan csökkentek, jelezve a modell hatékony tanulási folyamatát és az alkalmazkodását az adathalmazhoz. A magasabb értékek a tanítási folyamat kezdeti szakaszában megfigyelhetőek, és azt mutatják, hogy a modell tanul és alkalmazkodik az adathalmaz sajátosságaihoz. A FID értékek alakulása szintén figyelemre méltó volt. Kezdetben magasabb értékek jellemzik a FID-t, ami a generált képek és a valós képek közötti különbségeket tükrözi. Ahogy

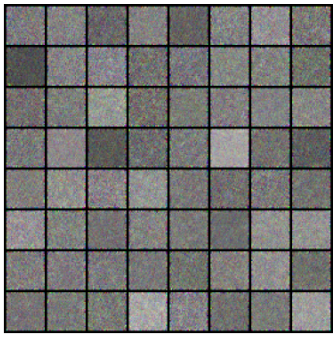
az epochok előrehaladtak, a FID értékek szisztematikusan csökkentek, ami azt sugallja, hogy a modell képes volt olyan generált képeket előállítani, amelyek strukturálisan és tartalmilag közelebb állnak a valóságbeli képekhez.
A FID értékek kiemelkedő csökkenése a 10. epochot követően igen ígéretes eredményt mutat. A kezdeti 8000-s értékről a 10. epochra már 1 alá került. Ez a jelenség azt jelzi, hogy a modell tanulási folyamata a 10. epoch után vált igazán hatékonnyá és eredményessé a generált képek minőségének javítása terén. Ez a szakasz arra utal, hogy a modell képes volt megragadni az összetettebb és magasabb szintű jellemzőket a tanítóadatokban, és ezeket sikeresen visszaadni a generált tartalomban.



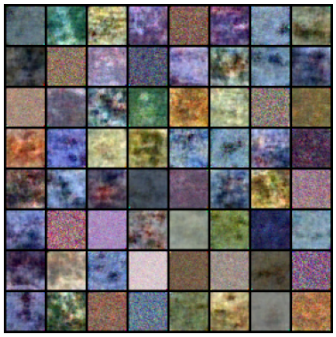
17. ábra. FID érték

Az alábbiakban bemutatásra kerülnek 1-1 epochban generált képek. A bemutatott képek tükrözik a generatív modell tanulási folyamatának haladását. Az első epochokban láthatók a kezdeti lépések, ahol a generált képek még zajosak és kevésbé strukturáltak. Azonban az idő előrehaladtával megfigyelhető, hogy a modell egyre inkább rögzíti és reprodukálja az adathalmaz jellemzőit. A képek részletei fokozatosan tisztulnak, és a zajszint csökken,

ami a tanulási folyamat hatékonyságát és a modell megbízhatóságát jelzi. A későbbi epochokban megfigyelhető, hogy a generált tartalom struktúrája és összetettsége fokozatosan javul, és a képek élesebbé válnak, idővel egyre kevésbé zajosak és hűen kezdik tükrözik az eredeti adathalmazt.



18. ábra. Epoch 1



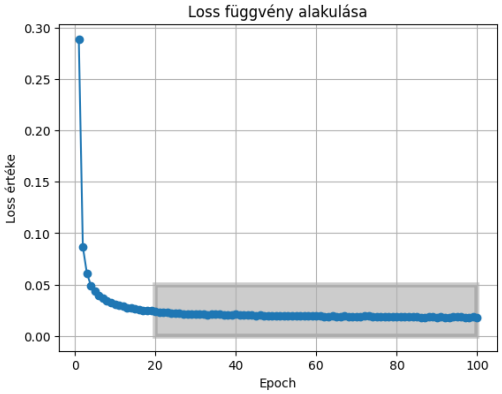
19. ábra. Epoch 10

20. ábra. Epoch 20

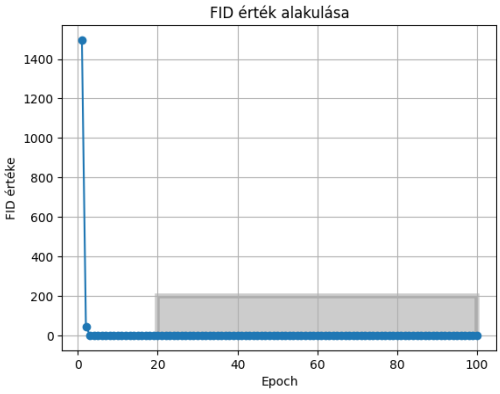
4.2.2. 2. mérés

A 2. mérés GitHub oldala.
A második mérést szintén a Google által szolgáltatott Colaboratory interaktív környezetben futtattam. Ebben a környezetben egy NVIDIA által fejlesztett V100 GPU-t használtam, amely nagy teljesítményének köszönhetően ideális választás volt a magasabb számítási igényű feladathoz. Az előző méréssorozatban összesen 20 epochot futtattam le, míg ebben a feladatban 100 epochig terjedt a tanítás. A magasabb epoch szám jelentős javulást eredményezett mind a képek zajtalanítása, mind pedig a FID értékek tekintetében. Az emelkedett epoch szám lehetővé tette a modell mélyebb tanulását, aminek következtében a képminőség szignifikánsan javult. Az iterációk számának jelentős növelésével párhuzamosan megfigyelhető, hogy a loss értékek is számottevően csökkentek a tanítási folyamat során. Ez az indikátor további bepillantást nyújt abba, hogy a modell mennyire hatékonyan tanult az idő előrehaladtával. A csökkenő loss értékek azt jelzik, hogy a modell jobban illeszkedik a tanító adathalmazhoz, és egyre jobban reprodukálja az eredeti képeket. A fokozott teljesítmény eredményeképpen az alacsonyabb loss értékek összhangban vannak a képek magasabb minőségével és az alacsonyabb

FID értékekkel. Ezáltal az elért eredmények nemcsak a vizuális eredményességet, hanem a modell stabilitását és konvergenciáját is tükrözik.
A következőkben bemutatom tanítási folyamat során kialakult loss és FID értékeket diagramokon keresztül. A diagram segítségével nyomon követhetjük a tanulási folyamat alatt a modell pontosságát és hatékonyságát, valamint észrevehetjük az iterációk számának növekedésével járó javulásokat mind a loss, mind pedig a FID értékek tekintetében.



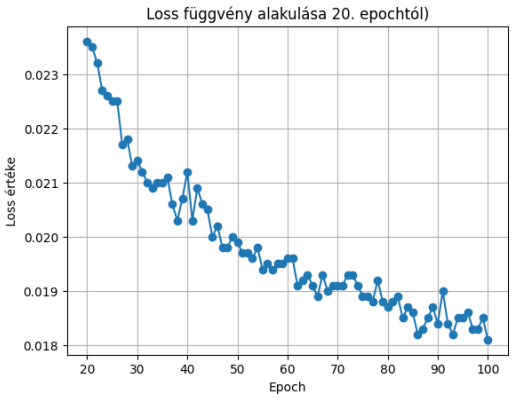
21. ábra. Loss



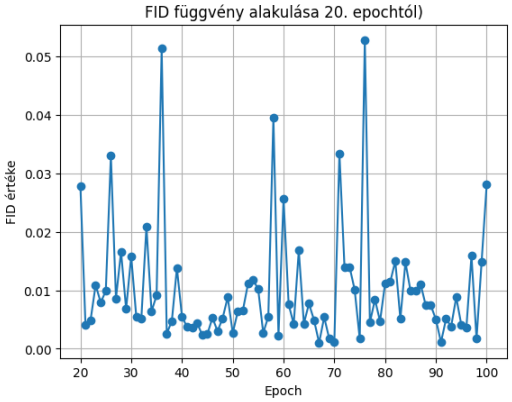
22. ábra. FID

Az ábrákról megfigyelhető, hogy a FID és a loss értékek egyaránt szignifikáns csökkenést mutattak a tanítási folyamat során. A 20. iterációtól kezdődően egyértelműen észrevehető a csökkenés ugrása, jelezve, hogy az ezt követő tanulási szakaszokban bevezetett változtatások és finomhangolások jelentős mértékben befolyásolták a modell teljesítményét. A FID értékek csökkenése arra utal, hogy a generált képek egyre közelebb kerültek az eredeti, valóságból származó képek eloszlásához, mérve ezzel a generált képek és a valóságos képek közötti hasonlóságot. A loss értékek csökkenése pedig azt jelzi, hogy a modell egyre jobban illeszkedik a tanító adathalmazhoz, minimalizálva a két eloszlás közötti különbséget.

A 20. epochtól kezdve jól megfigyelhető, hogy a folyamatosan csökkenő loss érték kifejezetten pozitív dinamikát tükröz a tanítási folyamatban. Ez az értékrendszer arra utal, hogy a modell egyre hatékonyabban illeszkedik a tanító adathalmazhoz. A folyamatos csökkenés azt jelezheti, hogy a modell fokozatosan megtanulja a képekben rejlő komplex összefüggéseket, struktú-



23. ábra. Loss



24. ábra. FID

rákat és jellegzetességeket. A FID értékek stagnálása egy kiemelkedő értéken, kiegészülve időnkénti kiugró értékekkel, érdekes jelenséget mutat a tanulási folyamatban. A FID értékek stabilizálódása azt jelzi, hogy a generált képek és a valóságból származó képek közötti hasonlóságok vagy különbségek egy adott szinten stagnálnak. Ugyanakkor az időnkénti kiugró értékek arra utalhatnak, hogy a modell bizonyos iterációk során nehezen tudja reprodukálni vagy közelíteni a valóságos képek jellegzetességeit. Ez a jelenség számos lehetséges magyarázattal rendelkezhet. A FID értékek stagnálása esetleg azt jelezheti, hogy a modell valamely ponton elért egy olyan képességi szintet, ahol további fejlődés lassú és nehezen észlelhető. Az időnkénti kiugró értékek pedig lehetnek következményei azoknak a helyzeteknek, amikor a modell nehezen kezeli vagy nem teljesen érti a tanító adathalmazt. Ez a jelenség lehetőséget nyújt arra, hogy további mélyreható elemzéseket végezzünk a tanulási folyamat dinamikájáról, és szükség esetén azonosítsuk azokat a konkrét eseteket, ahol a modell nehezen boldogul.

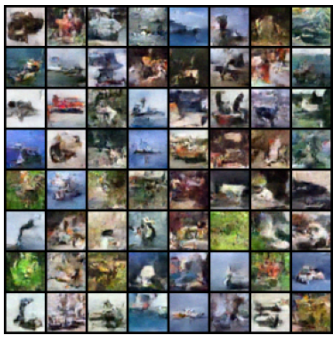
A FID átlagának vizsgálata az egész tanulási folyamat alatt és a 20. epochtól kezdve különösen érdekes perspektívát nyújt a modell fejlődéséről. Ebben a modellben a FID érték az egész tanulási folyamat alatt 15.44 volt. Az átlagos FID érték teljes tartományának áttekintése alapján megfigyelhető, hogy az értékek alapvetően egy tendenciát követnek az idő előrehaladtával. Azonban a 20. epochtól kezdődően az átlag 0.0102-re csökkent. Ez a jelenség azt sugallja, hogy a tanulási folyamat a 20. epochtól kezdve hatékonyan irányított volt és ez a csökkenés kézzelfoghatóan mutatja, hogy a tanítási stratégiák, finomhangolások és egyéb fejlesztések a 20. epochtól kezdve jelentős előre-

lépést eredményeztek a generált képek minőségében. A FID átlagának ilyen mértékű csökkenése arra utal, hogy a modell olyan szintre emelte a képalkotás képességeit, ahol a generált képek már közelítik vagy meghaladják a valóságból származó képek jellegzetességeit.

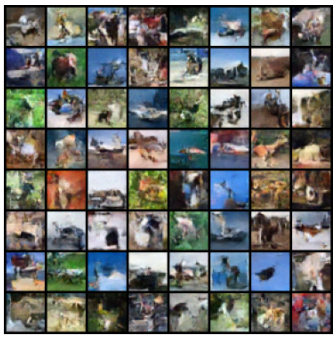
25. ábra. Epoch 20

26. ábra. Epoch 50

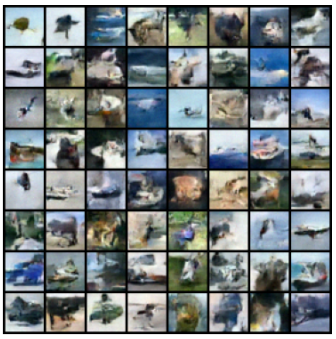
27. ábra. Epoch 67



28. ábra. Epoch 75

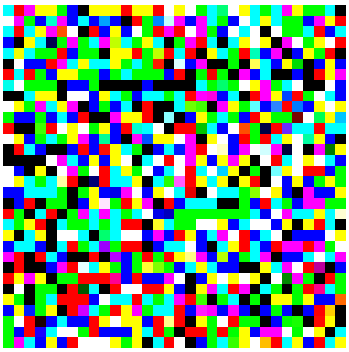


29. ábra. Epoch 90



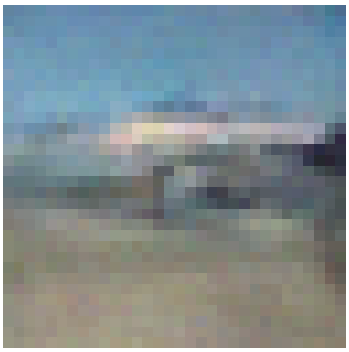
30. ábra. Epoch 100

A következő ábrákon repülőgépek generált képei láthatóak az adathalmazból, és jól szemléltetik a diffúziós folyamatot, amely a zajos kezdeti állapotról fokozatosan egyre tisztább és behatárolhatóbb képeket generál. Az első képeken láthatóak a diffúzió kiindulópontjaként szolgáló zajos képek, amelyeken a repülőgépek alakja még nem kivehető. Ahogy a diffúziós folyamat halad előre az időben, a képek fokozatosan tisztulnak, és a repülőgépek részletei egyre élesebben válnak láthatóvá. Az utolsó képen már jól kivehetőek a repülőgép részletei, és az eredmények olyan képek, amelyek kevésbé zajosak és részletgazdagabb reprezentációt nyújtanak, természetesen a korlátozott 32x32 pixeles tartományon belül, ahol a nagyon részletes és szép képek létrehozása kihívást jelent.



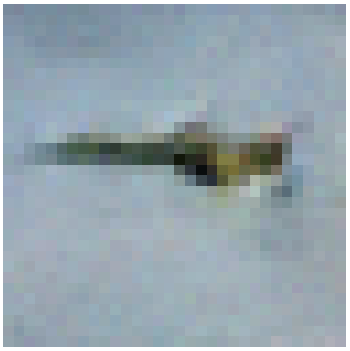
31. ábra. Epoch 1

32. ábra. Epoch 30



33. ábra. Epoch 74

34. ábra. Epoch 81



35. ábra. Epoch 95

36. ábra. Epoch 100

5. Összegzés

Szakdolgozatom témája rendkívül izgalmas és korszerű, hiszen a diffúziós modellek generatív modellezési területen való növekvő népszerűségével foglalkozik.
Az elején a mély tanulás matematikai alapjait és a szükséges valószínűségszámítási hátteret áttekintettem. Ezt követően részletesen kifejtettem több diffúziós modell elméletét és felépítését, valamint a köztük lévő viszonyokat.
A kutatás során átfogóan vizsgáltam a diffúziós modellek alkalmazását a generatív modellezés terén, különös tekintettel a Denoising Diffusion Probabilistic Model-re. Az eredmények egyértelműen alátámasztják, hogy a DDPM nem csupán elméleti értelemben érdekes, hanem gyakorlati alkalmazásaiban is kiemelkedő teljesítményt nyújt a valósághű generatív folyamatokban. Kiemelhető, hogy a további kutatás és modellek fejlesztése nagy ígérettel bír a valósághűbb és részletesebb generatív folyamatok terén.
A dolgozatom célja az volt, hogy hozzájáruljon a generatív modellezés területéhez, bemutatva a diffúziós modellek széleskörű és hatékony felhasználását a valóságos adathalmazokon.

6. Irodalomjegyzék

Hivatkozások

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: Advances in Neural Information Processing Systems 33 (2020), pp. 6840–6851.

[2] Jiaming Song, Chenlin Meng, and Stefano Ermon. “Denoising diffusion implicit models”. In: arXiv preprint arXiv:2010.02502 (2020).

[3] Prafulla Dhariwal and Alexander Nichol. “Diffusion models beat gans on image synthesis”. In: Advances in Neural Information Processing Systems 34 (2021), pp. 8780–8794.

[4] Alex Nichol and Prafulla Dhariwal. Improved Denoising Diffusion Probabilistic Models. 2021. arXiv: 2102.09672 [cs.LG]

[5] Robin Rombach et al. “High-resolution image synthesis with latent diffusion models”. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, pp. 10684–10695.

[6] Sewak, Mohit, Sanjay K. Sahay, and Hemant Rathore. "An overview of deep learning architecture of deep neural networks and autoencoders." Journal of Computational and Theoretical Nanoscience 17.1 (2020): 182-188.

[7] Bengio, Yoshua, et al. "Generalized denoising auto-encoders as generative models." Advances in Neural Information Processing Systems 26 (2013).

[8] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." Medical

Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18. Springer International Publishing, 2015.

[9] Guth, Florentin, et al. "Wavelet score-based generative modeling." Advances in Neural Information Processing Systems 35 (2022): 478-491.

[10] Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." Advances in Neural Information Processing Systems 33 (2020): 6840-6851.

[11] Nachmani, Eliya, Robin San Roman, and Lior Wolf. "Non gaussian denoising diffusion models." arXiv preprint arXiv:2106.07582 (2021)

[12] Hoogeboom, Emiel, et al. "Autoregressive diffusion models." arXiv preprint arXiv:2110.02037 (2021).