Szintetikus adatgenerálás – az adathozzáférés szent grálja?

DÉVAI GERGELY, RECSE ÁKOS
Ericsson Magyarország, ELTE
{gergely.devai; akos.recse}@ericsson.com

Kulcsszavak: mesterséges intelligencia, gépi tanulás, adathozzáférés, szintetikus adat

A mesterséges intelligenciát, gépi tanulást alkalmazó projektek számára szükséges tanítóadatok sokszor nem hozzáférhetőek az adattudósok számára. Különösen fontos probléma ez személyes vagy üzletileg érzékeny adatok esetén. Ilyen jellegű problémákra jelenthet megoldást a szintetikus adatgenerálás. Ez a technológia automatizált módon tanít generatív modelleket az eredeti adatokon, majd a modell segítségével az eredeti adatok statisztikai jellemzőit és egyéb mintázatait hordozó, szintetikus adathalmazt generál. Mindezt úgy, hogy az eredeti adat egyes rekordjairól a generált adat nem „szivárogtat ki" információt. Elemzésünkben áttekintjük a szintetikus adatgenerálás felhasználási területeit, valamint a módszer elméleti alapjait. Bemutatjuk négy kereskedelmi és hat szabad szoftvereszközzel kapcsolatos tapasztalatainkat és mérési eredményeinket.

1. Bevezetés

A mesterséges intelligencia és gépi tanulás térhódításával olyan eszköz került a vállalatok kezébe, aminek segítségével könnyebben megérthetik, felmérhetik és megjósolhatják a felhasználók igényeit. Az ehhez szükséges adatok sokszor azonban nem vagy csak részben hozzáférhetőek, például, ha egészségügyi vagy személyes adatokról van szó, melyet az Általános Adatvédelmi Rendelet (GDPR) miatt nem lehet az MI-alkalmazást fejlesztő szakemberek, cégek számára elérhetővé tenni. Ilyen jellegű problémákra jelenthet megoldást a szintetikus adatgenerálás.

Kétféle megközelítés terjedt el új adatok előállítására. Egyrészt készíthetünk szimulátorokat, melyekkel megpróbáljuk minél pontosabban leírni az adott rendszer viselkedését, hogy aztán új adatokat tudjuk generálni velük. Ebben az esetben a valós folyamatok modelljét egy szakértő állítja elő, programozza le. Másrészt az eredeti, érzékeny adatokkal taníthatunk generatív modelleket, amelyek automatikusan felismerik a bemenő adatok statisztikai tulajdonságait és jellemző mintázatait, majd ez alapján képesek olyan új adatpontokat generálni, amik minél több tulajdonságban hasonlítanak az eredeti adathoz. Cél azonban, hogy a generált adathalmaz alapján ne lehessen az eredeti adat egyes rekordjaira vonatkozó megállapításokat tenni. A két lehetőség közül ezúttal az utóbbival foglalkozunk.

Cikkünkben azt vizsgáljuk, hogy a jelenleg elérhető szintetikus adatgenerátorok képesek-e olyan minőségű új adatot előállítani, hogy a tisztán ilyen adatokon tanított gépi tanulási algoritmusok pontossága nem marad el lényegesen az eredeti adatokon tanított algoritmusok teljesítményétől, továbbá mérésekkel mutatjuk be, hogy a szintetikus adatok mennyire megkülönböztethetőek vagy éppen hasonlóak az eredeti adathoz.

A 2. szakaszban a szintetikus adatok felhasználási területeivel, a 3. szakaszban a módszereivel és eszköztárával foglalkozunk. A 4. szakasz vizsgálataink módszertanát, az 5. az eredményeit ismerteti. Az összefoglalás a 6. szakaszban található.

2. Használati esetek

Ahogy azt a bevezetőben is érintettük, jogi, üzleti vagy adatvédelmi megfontolásból magas szintű adatvédelmi szabályokat és folyamatokat alkalmazhatnak a vállalatok, melyek sok esetben nem teszik lehetővé a felhasználói adatok széleskörű megosztását. Szintetikus adatgenerátor használatával azonban úgy juthatnak értékes adathoz a fejlesztők, hogy nem kell a valós felhasználói adatokkal dolgozni. Természetesen ez a módszer is körültekintő használatot igényel, hiszen egy túltanított adatgeneráló algoritmus előállíthat olyan adatot, ami megfeleltethető az eredeti adathalmaz valamely rekordjának, vagy akár azonos azzal. Ennek elkerülése érdekében a generált adat „hasznosságán" túl annak „biztonságosságát" is vizsgálni kell.

A fent említett eseten túlmutatóan egyéb környezetben is használhatunk szintetikus adatgeneráló algoritmusokat. Az egyik ilyen lehetőség az adathalmaz-bővítés: amennyiben kevés az elérhető adat egy pontos gépi tanulási algoritmus tanításához, lehetőség van olyan új adatpontokat generálni, melyekkel kiegészíthetjük az eredeti adathalmazt. Ez a módszer különösen akkor ígéretes, ha a generált adatokhoz egy szakértő plusz tudást ad hozzá, például szűrőfeltételek megadása segítségével.

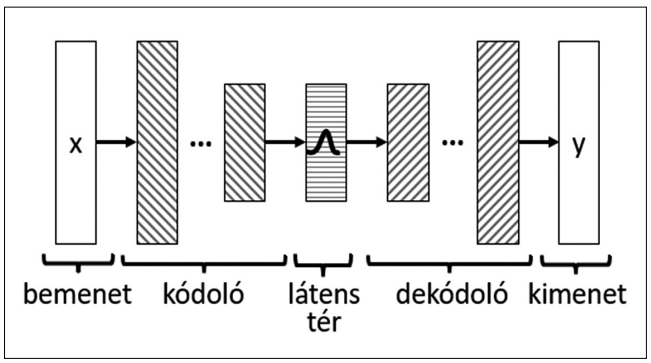
Példa erre a képekkel dolgozó modellek tanítóhalmazának bővítése egyszerű műveletekkel (forgatás, kivágás, skálázás stb.), melyek a képekhez rendelt címkét helybenhagyják [1]. Itt a szakértői tudás a megfelelő transz-

formációk kiválasztásában jelenik meg. Egy másik példa a ritka eseményekkel foglalkozó modellek (pl. anomália-felismerés) esete. Mivel ezekből a valós adathalmazban kevés szerepel, a tanítás hatékonysága érdekében a múltbeli ritka vagy szélsőséges eseményekből kiindulva új, hasonló adatpontokat kell generálni.

3. Módszerek és eszközök

Adatgenerálásra számos modellarchitektúra használható. A teljesség igénye nélkül ezek közül mutat be néhányat ez a szakasz. Kitérünk továbbá az általunk vizsgált kereskedelmi és szabad szoftvereszközökre.

A VAE (Variational Autoencoder) egy neuronháló-architektúra *(1. ábra)*, amelyben a bemenő vektorokat (x) a kódoló egy látens térbe transzformálja. Ez utóbbi a VAE esetén eloszlások kombinációja. A dekódoló a látens tér pontjait képezi le a bemenettel megegyező alakú kimenetté (y). A komponenseket úgy tanítják, hogy a be- és kimenet eltérése minimális legyen.



1. ábra A VAE neuronháló-architektúra

A VAE úgy használható adatgenerálásra, hogy a látens tér eloszlásaiból mintavételezünk, majd ezeket a dekódoló adatvektorokká alakítja.

A kopulák olyan matematikai objektumok, amelyeket többváltozós eloszlások modellezésére használhatunk. A többváltozós eloszlást két komponensre, az egyes változók marginális eloszlására és az egyes változók közötti összefüggésre bontjuk. A kopula feladata ez utóbbi modellezése. Adatgenerálás az érintett eloszlásokból történő mintavételezéssel történik.

A GAN- (Generative Adversarial Network) architektúra *(2. ábra)* esetén a Generátor neuronháló véletlen zajból, mint stimulusból állít elő adatvektorokat. A Diszkriminátor egy másik neuronháló, ami vagy egy valódi, vagy egy generált adatvektort kap inputként, és a feladata azt eldönteni, hogy milyen inputot kapott. A Generátor és Diszkriminátor modellek tehát „egymás ellen játszanak”: minél valószerűbb a generátor kimenete, annál nehezebb a Diszkriminátor dolga. Szintetikus adat előállítására a Generátor használható a tanítás befejeztével.

A GAN-architektúrának számos változata van, például feltételes és adatbázis-generáláshoz használt (cGAN, CTGAN) [6], nembináris diszkriminátorral rendelkező (WGAN) [14], vagy differenciális adatvédelemhez használt (DPGAN) [13] stb.

A fenti módszereknek számos kereskedelmi és szabad szoftveres implementációja létezik. Ezek közül kutatásunk során a következőket vizsgáltuk:

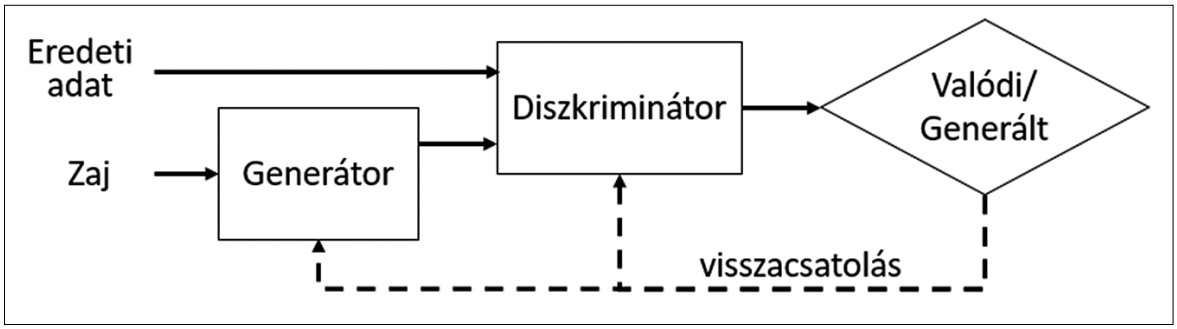
- DeltaPy (szabad, nyílt forráskódú) [10]
- Gretel (kereskedelmi) [7]
- Gretel nyílt forráskódú változata [8]
- Mostly.AI (kereskedelmi) [2]
- SDV (szabad, nyílt forráskódú) [5]
- Statice (kereskedelmi) [4]
- Synthpop (szabad, nyílt forráskódú) [11,12]
- Syndata (szabad, nyílt forráskódú) [15]
- YData (kereskedelmi) [3]
- YData nyílt forráskódú változata [9]

4. Módszertan

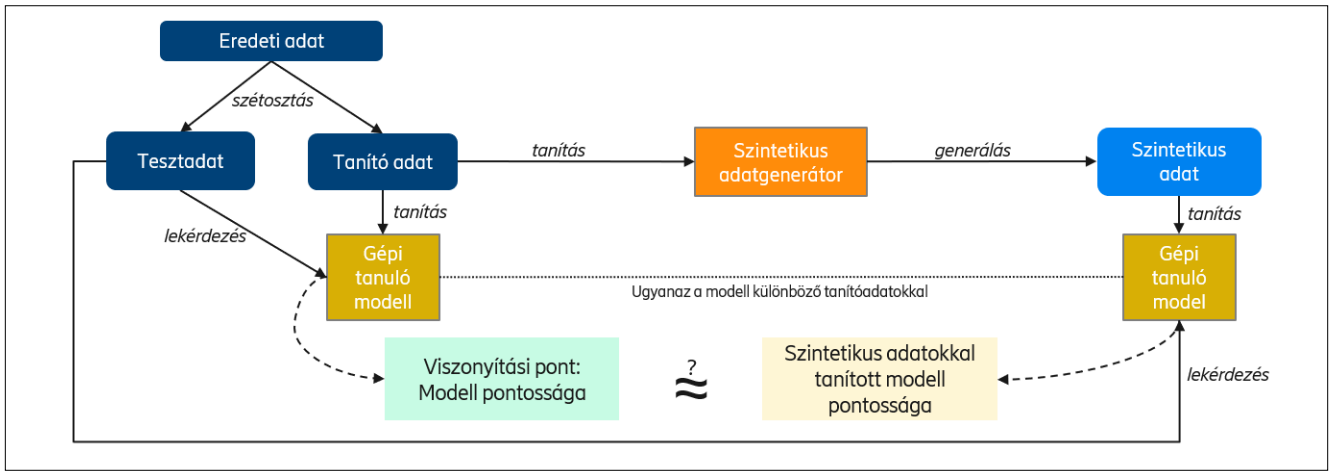
A kísérletekhez olyan adathalmazra volt szükség, amelyen mobilhálózatok szempontjából releváns gépi tanuló algoritmusokat lehet tanítani. Egy laborkísérlet mérési eredményeit használtuk: az adatokban szerepeltek a mobiltelefon rádiós környezetére vonatkozó információk, a maghálózatban mért adatforgalmi jellemzők, valamint egy videókonferencia-alkalmazás által feljegyzett adatok: a hang és videó minőségét leíró mérőszámok. Ez utóbbiak általában nem állnak a hálózatüzemeltetők rendelkezésre, ezért ezek becslését választottuk a kísérletünkben szereplő gépi tanuló modell feladatául.

A nyers adatrekordok sorrendezettek voltak és időbélyeget is tartalmaztak. Az első kísérlet arra irányult, hogy mennyiben tudják az egyes szintetikus adatgeneráló eszközök megőrizni egy ilyen adatsor tulajdonságait. Ehhez 6 idősoros tulajdonságot választottunk ki.

A második kísérlethez a nyers adatokat átalakítottuk időbélyeg és sorrendezés nélküli adathalmazzá. Ezen az adathalmazon egy Tensorflowban megvalósított, egysze-



2. ábra A GAN-architektúra



3. ábra A szintetikus adatgenerátorok kiértékelési módszertana

rű neurális hálót tanítottunk, amelynek feladata egy kiválasztott, alkalmazási rétegbeli mutató becslése volt a hálózati mérőszámok alapján.

Az adathalmazt teszt- és tanítóhalmazra bontottuk. A modell az utóbbi alapján tanult, míg az előbbi segítségével kiértékeltük a teljesítményét. Ez látható a *3. ábra bal oldalán*.

A modell tanítását megelőzően az adatok a gépi tanulásnál szokásos előfeldolgozáson mentek keresztül, például számos oszlopra – beleértve a céloszlop értékeit is – logaritmusfüggvényt alkalmaztunk. A modell becsléseinek átlagos hibájára a teszthalmazon mérve 0,2 adódott úgy, hogy a céloszlop logaritmusának értékei egy 4-es szélességű intervallumban szóródnak.

Megjegyzendő, hogy ennek a kísérletnek nem a választott mutató minél pontosabb előrejelzése volt a célja, hanem annak megállapítása, hogy mennyit romlik a modell teljesítménye, amikor valós adatok helyett szintetikus adatokon tanítjuk. Ennek a célnak a 0,2 átlagos becslési hibát adó modell megfelelt, ezért ennek javítására további erőfeszítéseket nem tettünk.

A tanító adatot ezután az egyes szintetikus adatgenerátorok tanítására használtuk, majd az eredeti tanítóhalmazzal azonos méretű szintetikus adathalmazt generáltunk az eszközökkel. Ezeken a szintetikus adatokon tanítottuk a korábban említett Tensorflow-modellt és megvizsgáltuk a kapott átlagos hibákat. Fontos, hogy a kiértékelés itt is az eredeti tesztadatokon történik. Ez látható a *3. ábra jobb oldalán*. Minél közelebb van ezen második modell teljesítménye az eredeti adaton tanított modell teljesítményéhez (azaz minél kevésbé nő az átlagos becslési hiba), annál jobb minőségű a szintetizált adat, illetve az azt létrehozó generátor eszköz.

A szintetikus adathalmazzal szemben azonban nemcsak az a követelmény, hogy minél hűebben megőrizze az eredeti adat statisztikai jellemzőit és a benne rejlő mintázatokat. Ugyanilyen fontos, hogy ne „szivárogtassa ki” az eredeti adat egyes rekordjainak információit, például ne legyen kikövetkeztethető a szintetikus adatból, hogy az eredeti adatban szereplő előfizetők mikor kinek telefonáltak, milyen szolgáltatásokat használtak a mobiljukkal és merre jártak. Egyes generátoreszközök automatikusan előállítanak egy dokumentumot a szintetizált adathalmazról, amelyben ezt az aspektust is elemzik, különféle „támadásokat” szimulálva.

Ezeken a generált dokumentumokon túl egy saját metrikával is vizsgáltuk, mennyire biztonságosak az egyes szintetikus adathalmazok. Megmértük, hogy milyen mértékben hasonló az eredeti adat teszt- és tanítóhalmaza egymáshoz („önhasonlóság”), majd azt is, hogy a szintetikus adat mennyire hasonlít az eredeti tanítóhalmazhoz („szintetikus adat hasonlósága az eredetihez”). Ha a szintetikus adat jobban hasonlít az eredetihez, mint amennyire az eredeti adat önhasonló, az arra utal, hogy a generátor eszköz nem valódi szintézist végez, hanem részben vagy egészben másolja az eredeti rekordokat.

5. Eredmények

A kereskedelmi szoftverekkel kapcsolatos eredmények bizalmasak, ezért az ebben a szakaszban a vizsgált négy eszközre SZ1-SZ4-ként hivatkozunk, véletlenszerű hozzárendeléssel.

Az első kísérletben idősoros adatokból kiindulva szerettünk volna új adatsorokat generálni, azonban erre egyik eszköz sem volt képes az elvárt minőségben. Az SZ2-szoftver tudott bizonyos metrikák alapján hasonló adatsort előállítani, mint a bemeneti halmaz, azonban a további vizsgálatok során kiderült, hogy bizonyos adatoszlopokat egy az egyben emelt át onnan, amely kizárja a szintetikus adatok használhatóságát az általunk érdekesnek vélt esetekben. A többi kereskedelmi eszköz a 4. szakaszban említett 6 idősoros tulajdonság közül csupán 2-4 tulajdonságot tudott megőrizni a szintetizált adatban. Hogy megértsük az idősoros adatok generálásának határait, készítettünk egy egyszerű, szinuszhullámot leíró adathalmazt, amit bemeneti adatként alkalmaztunk. Az SZ1 és SZ2 üzleti szoftverek képesek voltak lekövetni a mintázatot, és azt alapul véve új adatot tudtak generálni. A többi program esetében nem volt sikeres ez a kísérlet sem.

A 4. szakaszban ismertetett módszertant alkalmazva megmértük, hogy az egyes eszközök által generált szin-

tetikus adathalmazokkal tanított gépi tanulási algoritmus milyen hibával képes megbecsülni a céloszlop értékeit a mérés elején leválasztott teszthalmazon. Ennek eredményei az *1. és 2. táblázatban* láthatóak.

Összesen négyféle neurális hálón alapuló modellt használtunk különböző számú szintekkel és neuronokkal, ezek 1-4-ig vannak számozva a táblázatban. A táblázatokban az első sor az eredeti adatokon tanított és validált modell átlagos hibáját adja meg. A következő sorok az egyes szoftverek által előállított szintetikus adatokon tanított modell átlagos hibáját, valamint a referenciaértékhez képest számolt hibanövekedést írják le. A kereskedelmi szoftverek közül az SZ1 és SZ2 produkálta a legkisebb átlagos hibát a méréseink során. Az SZ2-program esetében azonban egyes kiugró értékeket ki kellett szűrnünk a teszthalmazból. Az SZ1-program adatain végzett mérés átlagosan 31%-kal mutatott rosszabb eredményt, mint a referenciamérés, míg az SZ2 esetén ez 13,75%. Jól látható, hogy nagy eltérés lehet a mérésekben a modell felépítésétől függően (SZ1 a legjobb: 18%, SZ1 a legrosszabb: 39%).

A nyílt forráskódú programcsomagok közül a Synthpop-szoftver ért el a kereskedelmi szoftverekhez hasonló eredményt. A többi ingyenes eszköz által generált adatok használatakor a modellek legalább 187%-kal rosszabbul teljesítettek, mint a referenciamérések során, a legrosszabb esetben ez a szám elérte a 385%-ot. A 2. táblázatban nem szereplő nyílt forráskódú eszközökkel a kísérlet sikertelen volt.

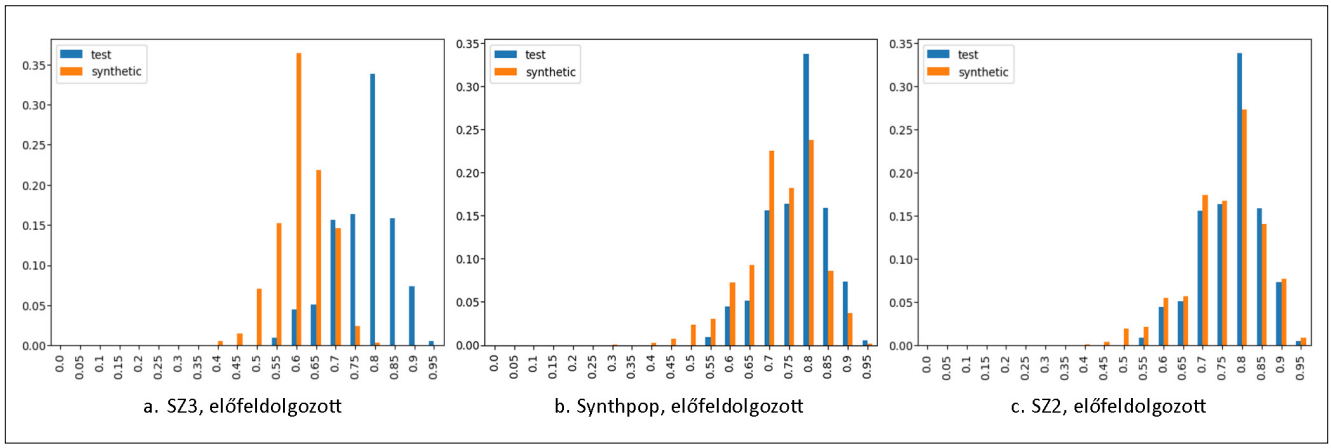
Egy másik kísérletben az eredeti és a szintetikus adatok közti hasonlóság azon dimenzióját szerettük volna leírni objektív metrikával, amely szerint az új adatpontokból nem tudunk visszakövetkeztetni az eredeti adatpontokra. Ennek mérésére a 4. szakaszban leírtaknak megfelelően hasonlósági mérőszámot alkalmaztunk egyrészt az eredeti adaton belül, másrészt az eredeti és a gene-

1. táblázat Kereskedelmi adatgenerátorok eredményei

	Időbélyeg nélküli adathalmaz				Előfeldogozott, időbélyeg nélküli adathalmaz			
	gépi tanulási modell:				gépi tanulási modell:			
	1	2	3	4	1	2	3	4
Referencia	0.2247	0.2087	0.2011	0.1951	0.2236	0.1953	0.2062	0.1943
SZ1	0.2767	0.2721	0.2752	0.2675	0.2649	0.2718	0.2666	0.2622
SZ1 hibatöbblet	23%	30%	37%	37%	18%	39%	29%	35%
SZ2	-	-	-	-	0.2426	0.2297	0.2323	0.2254
SZ2 hibatöbblet	-	-	-	-	8%	18%	13%	16%
SZ3	0.2967	0.2952	0.2918	0.2998	0.4517	0.5539	0.4792	0.6196
SZ3 hibatöbblet	32%	41%	45%	54%	102%	184%	132%	219%
SZ4	0.4793	0.5031	0.4804	0.4843	0.4848	0.4843	0.4756	0.488
SZ4 hibatöbblet	113%	141%	139%	148%	117%	148%	131%	151%

	Időbélyeg nélküli adathalmaz				Előfeldogozott, időbélyeg nélküli adathalmaz			
	gépi tanulási modell:				gépi tanulási modell:			
	1	2	3	4	1	2	3	4
Referencia	0.2247	0.2087	0.2011	0.1951	0.2236	0.1953	0.2062	0.1943
Gretel (nyílt)	-	-	-	-	0.6423	0.629	0.595	0.6506
Gretel hibatöbblet	-	-	-	-	187%	222%	189%	235%
Synthpop	0.2578	0.2355	0.254	0.2401	0.2563	0.2412	0.286	0.2446
Synthpop hibatöbblet	15%	13%	26%	23%	15%	24%	39%	26%
SDV	0.6261	0.56764	0.57009	0.58682	0.71699	0.75138	0.54922	0.69036
SDV hibatöbblet	279%	272%	283%	301%	321%	385%	266%	355%

2. táblázat Nyílt forráskódú adatgenerátorok eredményei



4. ábra
Adatpontok hasonlósága az eredeti adatokon belül (kék), valamint az eredeti és szintetikus adatpontok között (narancs)

rált adat között. A mérések során szinte az összes szintetikus adathalmaz megfelelőnek bizonyult, mivel a szintetikus adat nem hasonlított jobban az eredetihez, mint az eredeti adat önmagához. A *4.a. ábra* egy ilyen sikeres mérés eredményét mutatja.

Az SZ3-eszköz által generált szintetikus adatok elemenkénti hasonlósága az eredeti adathoz narancssárgával szerepel az ábrán, és mivel ez balra tolódva található a kékkel jelölt eredeti adathalmazon belüli hasonlósági hisztogramtól, sikeresnek tekinthetjük a mérést. Hasonló ábrákat kaptunk a többi eszköznél is, amelyek nem szerepelnek a fenti ábrák között. A Synthpop esetén *(4.b. ábra)* a hisztogram eltolódása jóval kevésbé szembetűnő, de még elfogadható. Az SZ2-eszköz diagramján pedig látható, hogy a kívánt tulajdonság a nagy hasonlóságú rekordok esetén sérül (*4.c. ábra,* lásd a hisztogramok jobb szélét). Ez az eredmény utalhat arra, hogy a szintetikus adathalmaz bizonyos pontjaiból nagyobb valószínűséggel lehet visszakövetkeztetni az eredeti adatpontokra, mint az eredeti adathalmaz tetszőleges adatpontjaiból a többire.

Végül az eszközök nem funkcionális képességeivel kapcsolatos tapasztalatokat foglaljuk össze:

• **Grafikus felhasználói felület:** A vizsgált nyílt forráskódú szoftverek egyike sem rendelkezik grafikus felhasználói felülettel. Az SZ1 és SZ3 kereskedelmi szoftverek teljes grafikus támogatással érkeznek, míg az SZ2- és SZ4-programok részlegesen vezérelhetők a grafikus felületen keresztül. Utóbbi esetekben néhány funkció csak a parancssoros felületen érhető el.

• **Dokumentáció:** Az SZ1-szoftver esetében teljeskörű dokumentációt kapunk. SZ2 esetében a dokumentáció példákon keresztül mutatja be a program használatát, míg az SZ3- és SZ4-programok használatához a gyártó munkatársainak utasításait követtük többségében. A nyílt forráskódú megoldások közül mindegyik dokumentációval érkezett, melyekben, bár lehetett hibát találni, alapvetően jól leírták a szoftvereket felhasználói szempontból.

• **Használhatóság:** Érdemes megjegyezni, hogy ez a pont inkább informatív jelleggel szerepel, hiszen részben a programok használata során szerzett szubjektív vélemények alapul. Az SZ1-, SZ3- és SVD-szoftvereket találtuk a legkönnyebben és legintuitívabban használhatónak. A Synthpop-szoftver az automatikus típusfelismerés hiányát leszámítva egy letisztult programozói interfészt biztosít. Az SZ2- és a nyílt forráskódú Gretel-programok limitáltan felhasználóbarátok (például olyan egyedi koncepciók alkalmazása, amik nincsenek megfelelően dokumentálva), míg a SZ4-program használata nagyobb nehézségek árán volt csak lehetséges.

• **Konfigurálhatóság:** A vizsgált programok többségében széles skáláját támogatták a beállítási lehetőségeknek. A Synthpop-szoftvernek a Python-alapú csomagját használtuk, azonban az eredeti csomag R-ben íródott és a konfigurációs paraméterek egy jelentős része nem érhető el a Python-alapú verzióban.

• **Robusztusság:** A tesztelt szoftverek általában egy-egy hibától eltekintve stabilan működtek. Az SZ1-program kiemelkedően stabilnak bizonyult. Az SZ2- és SZ4-programok azonban olyan hibákat produkáltak, melyekre nem számít a felhasználó és nincs is kész megoldás az orvoslásukra. (A használhatósági jellemzőkhöz hasonlóan ez a metrika is informatív jelleggel szerepel, mivel a programok használata során szerzett szubjektív tapasztalatokon alapul.)

• **Jelentésgenerálás:** A kereskedelmi szoftverek mindegyike képes felhasználónak szánt (például pdf formátumú) jelentést generálni az előállított adatok statisztikai

3. táblázat
Az adatgenerátorok használhatósága a nem funkcionális kritériumok alapján

Szoftver	Használhatóság
SZ1	6
SZ2	3.5
SZ3	5
SZ4	3
SDV	4
Synthpop	3
Gretel (nyílt)	2.5

jellemzőiről, valamint az eredeti adatokhoz való hasonlóságukról. Ezzel szemben a nyílt forráskódú programok nem nyújtottak ilyen jellegű funkciókat, kivéve a Gretel nyílt változatát, amely képes nagyon limitált jelentéseket generálni.

A *3. táblázatban* a korábban felsorolt nem funkcionális tulajdonságokat egyesével értékelve (0-1) majd azokat összeadva elért pontszámok láthatóak. A magasabb pontszám jobb használhatóságot jelent.

6. Összefoglalás

Cikkünkben bemutattuk, hogy a gépi tanulási algoritmusok fejlesztéséhez gyakran nem érhető el a szükséges adat, amelyre bizonyos esetekben megoldást jelenthet szintetikus adatgenerátorok alkalmazása. Megmutattuk, hogy személyes vagy érzékeny adatok, ritka és kevés adat esetén is érdemes megfontolni ezen eszközök alkalmazását, viszont ezek körültekintő használatot igényelnek. Megmutattuk, hogy milyen módszereken alapulnak a jelenleg elérhető szintetikus adatgeneráló algoritmusok. Bemutattuk az általunk használt mérési környezetet, melyben neurálisháló-alapú algoritmusok tanításának pontosságát értékeltük ki egy valós és különböző eszközök által generált szintetikus adathalmazokon.

Vizsgálatunk során négy kereskedelmi és hat nyílt forráskódú szoftvert használtunk. A kísérletek három nyílt forráskódú eszközzel sikertelenek voltak. Egy kereskedelmi eszközt (SZ2) pedig az adatvédelemmel kapcsolatos aggályok miatt zártunk ki. A maradék eszközök közül a legjobbak (a kereskedelmi SZ1 és a nyílt Synthpop) csupán 15-18%-os hibatöbbletet produkáltak a legkedvezőbb esetben és 23-31%-osat átlagosan. Láttuk azonban, hogy a különböző eszközök között nagy a különbség, illetve egy-egy programon belül is függ az eredmény a vizsgált neurális háló tulajdonságaitól.

Összegyűjtöttük továbbá a programok használatakor tapasztalt, nem funkcionális képességekkel kapcsolatos észrevételeinket. A kereskedelmi szoftverek átlagosan felülmúlták a nyílt forráskódú programokat mind funkcionális, mind nem funkcionális képességek tekintetében. A Synthpop nevű programcsomag kiemelkedett a nyílt forráskódú programok közül, azonban ez az eszköz egyáltalán nem támogatja az idősoros adatokat.

A mérések alapján láthatjuk, hogy a szintetikus adatgenerátorok gyakorlati alkalmazása lehetséges, azonban számítani kell a felmerülő többlethibára. Ez azonban elfogadható, különösen akkor, ha szintetikus adatgenerálás nélkül egyáltalán nem hozzáférhető az eredeti adat az adott projekt számára. Az idősoros adatok generálása kapcsán problémákba ütköztünk: ezen a területen az eszközök további fejlődésére számítunk.

Hivatkozások

[1] Perez, L., & Wang, J. (2017): The effectiveness of data augmentation in image classification using deep learning. arXiv preprint arXiv:1712.04621.
[2] Mostly.AI weboldala: *https://mostly.ai/synthetic-data-platform/*
[3] YData weboldala: *https://ydata.ai/*
[4] Statice weboldala: *https://www.statice.ai/*
[5] Synthetic Data Vault weboldala: *https://sdv.dev/*
[6] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, Kalyan Veeramachaneni: Modeling Tabular Data using Conditional GAN, Advances in Neural Information Processing Systems, Vol. 32, 2019.
[7] Gretel weboldala: *https://gretel.ai/*
[8] Gretel nyílt forráskódú verziója: *https://github.com/gretelai/gretel-synthetics*
[9] YData nyílt forráskódú verziója: *https://ydata.ai/products/synthesizer*
[10] Derek Snow: DeltaPy: Tabular Data Augmentation, 2020. *https://github.com/firmai/deltapy*
[11] Synthpop Python implementáció: *https://github.com/hazy/synthpop*
[12] Synthpop R implementáció: *https://www.synthpop.org.uk/*
[13] Liyang Xie et. al.: Differentially Private Generative Adversarial Network, CoRR, 2018.
[14] Martin Arjovsky et. al.: Wasserstein GAN, arXiv, 2017.
[15] Syndata implementáció: *https://github.com/LLNL/SYNDATA*
(weboldalak elérési dátuma: 2023.02.28.)

A szerzőkről

RECSE ÁKOS 2016-ban mérnökinformatikus szakon BSc-, 2018-ban pedig gazdaságinformatikus szakon MSc-fokozatot szerzett a Budapesti Műszaki és Gazdaságtudományi Egyetemen. 2018 óta az Eötvös Loránd Tudományegyetem Informatikai Doktori Iskolájának hallgatója. 2019-től a European Institute of Innovation & Technology (EIT) doktori iskolájának hallgatója. Doktori tanulmányai során főbb témái az elosztott, felhőalapú infrastruktúrák automatizált felügyelete és erőforrás-menedzselése, valamint a decentralizált adattárolási szolgáltatások vizsgálata. 2022 óta dolgozik az Ericsson Magyarországnál, System Architect pozícióban.

DÉVAI GERGELY jelenleg az Ericsson mobilhálózat-analitikai és hálózatautomatizálási termékeivel, valamint a kutatási eredmények termékesítési lehetőségeivel foglalkozik. Diplomát és doktori fokozatot az Eötvös Loránd Tudományegyetem Informatikai Karán szerzett, ahol azóta is oktat. Számos ipari-egyetemi kutatási projektet vezetett, főként szoftvertechnológiai témákban. Érdeklődése középpontjában az analitikai rendszerek praktikus kihívásai állnak: adathozzáférés, zártláncú automatizálás, az adatfeldolgozás erőforrásigényének csökkentése.