

TDK-dolgozat

Szabó Zoltán

ESZTERHÁZY KÁROLY KATOLIKUS EGYETEM INFORMATIKAI KAR

Szemantikus szegmentáció mély tanulási algoritmusok segítségével

Témavezetők:

Dr. Tajti Tibor

adjunktus

Kovács Ádám

óraadó oktató

Szerző:

Szabó Zoltán

Eger, 2023

Tartalom

Bevezetés.....	4
Gépi tanulási modell létrehozása.....	7
A modell felépítése.....	7
Adat augmentációs modell	9
A modell tanítása	9
Első futási eredmények összehasonlítása	14
Újrafuttatás eredmények összehasonlítása	16
Adat augmentációs eredmények	19
Összefoglalás	23
További kutatási célok.....	23
Záró gondolatok	23
Irodalomjegyzék.....	24

Bevezetés

A mesterséges intelligencia lényeges szegmense a modern tudománynak, amelynek ipari felhasználása számottevő technológiai előrelépést biztosít. Mindenre kiterjedő felhasználhatósága az emberihez hasonló kompetenciák utánzásában rejlik, mint például a tanulás, a tervezés, az érvelés, esetleg a problémamegoldás. Támogatásával a különféle számítógépes rendszerek előjelzéseket tudnak készíteni, esetleg a meglévő adatokban található különböző mintázatok alapján műveleteket tudnak végrehajtani. Fontos, hogy ezek a rendszerek képesek tanulni a hibáikból, ezáltal is növelve a pontosságukat. A mesterséges intelligencia részhalmazának tekintendő a gépi tanulás, amely számítógépeket tanít be adatok emberi szintű elemzésére, továbbá értékelésére.

A tudományos diákköri dolgozatom célkitűzése egy CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) tesztek tartalmazó adathalmaz elemeinek osztályozása neurális hálózat segítségével különféle paraméterek használatával. Végezetül a modellek által kapott eredmények kiértékelésének, ezen kívül azok statisztikai elemzésének bemutatása következik.

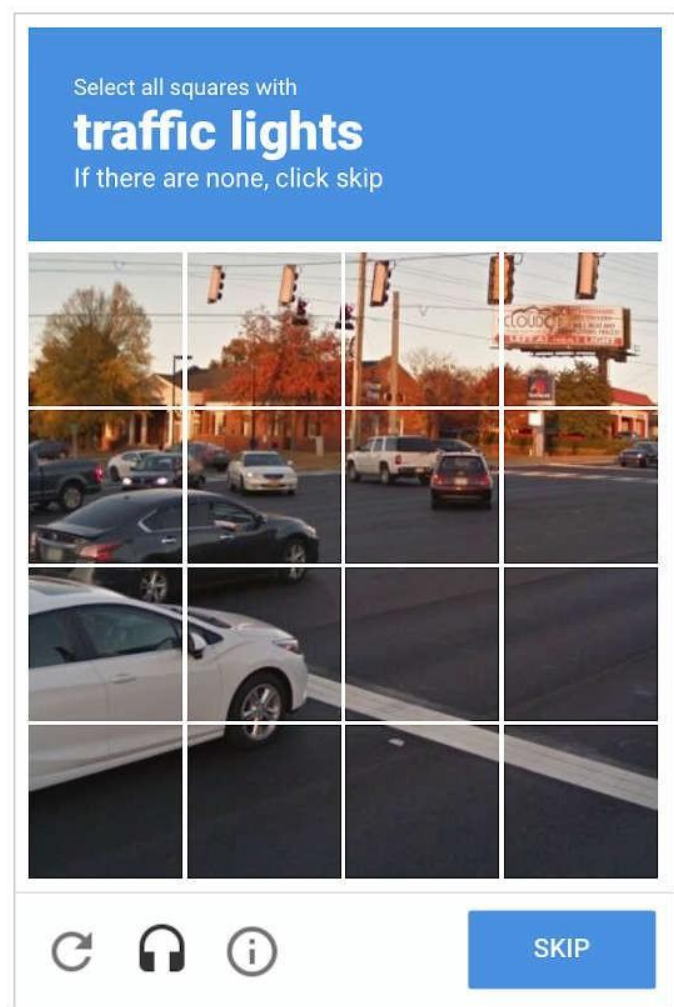
A különféle CAPTCHA tesztek még a mai napig olyan jelentős webhelyek is alkalmazzák, mint a Google, Yahoo, valamint a Microsoft. A tesztek célkitűzése, hogy meg tudják különböztetni a valós felhasználókat a botoktól ezáltal is csökkentve a rosszindulatú internetes támadások számát. Ebből kifolyólag a különféle CAPTCHA tesztek használhatók webszolgáltatások, webhelyek, bejelentkezési adatok, de akár még vezetőtámogató rendszerek védelmére is az eltérő kiberbiztonsági fenyegetésekkel, támadásokkal, továbbá behatolókkal szemben.

A CAPTCHA tesztek jellegüket tekintve többféle csoportba is lehet sorolni, amik a következők:

- szöveg alapú,
- kép alapú,
- hang alapú,
- videó alapú,
- egér mozgásán alapuló,
- és egyéb változatok.

A szöveg alapú CAPTCHA kódokat tartalmazó képek a legelterjedtebb és legegyszerűbb formája a CAPTCHA teszteknek. A szövegek betűkből és számjegyekből állnak össze,

amelyeken különböző módosításokat végeznek el a nehezebb felismerhetőség érdekében. Ilyen módosítás például a karakterek torzítása, elhomályosítása vagy zaj alkalmazása. Azonban mára a technológiai rohamos fejlődése már oda vezetett, hogy egyre elterjedtebb lett a kép alapú CAPTCHA tesztek alkalmazása, hiszen a különböző gépi tanulási algoritmusok egyre nagyobb pontossággal ismerik fel a szöveg alapú CAPTCHA kódokat. A legelterjedtebb kép alapú CAPTCHA teszteknel általában a felhasználóknak kell kiválasztaniuk a megadott képek közül azokat, amelyek tartalmazzák a teszt által kért dolgokat, hogy átmenjenek a biztonsági teszten.



Kép alapú CAPTCHA teszt, ahol közlekedési lámpákat kell felismerni

Továbbá megjelentek a hang alapú CAPTCHA tesztek alkalmazásai is a szöveges CAPTCHA tesztek túlzott sebezhetőségének leküzdésére. A hang alapú CAPTCHA sikeres teljesítése érdekében a felhasználónak meg kell hallgatnia egy előre felvett hangklipet, majd válaszként meg kell adnia az abban hallottakat. Gyakori példa, hogy némi zaj adódik hozzá a lejátszott hanghoz, hogy a botok számára megnehezítse a tesztek megoldását. Azonban a túl sok háttérzaj alkalmazása a felhasználók számára is nehezen felismerhetővé teheti a tesztet, ez által befolyásolva a megfelelő használhatóságát.

Az egyik legbiztonságosabb védelmi mechanizmus a videó alapú CAPTCHA tesztek használata, hiszen ebben az esetben a felhasználóknak különféle kérdésekre kell válaszolniuk a bemutatott videó alapján. Viszont az ilyen tesztek hátránya, hogy a felhasználóknak általában több időt kell eltölteni a videó megértésével és a kérdések megválaszolásával, mint más CAPTCHA tesztek esetében.

A Google által fejlesztett No CAPTCHA vagy más néven reCAPTCHA rendszer esetében csak egy négyzetet kell kipipálni, azonban itt a háttérben a számítógépes egér mozgásának követése történik meg. Ezáltal a biztonsági rendszer el tudja dönteni, hogy a kurzor mozgatása mennyire hasonlít arra, ha azt valóban egy átlagos ember hajtaná végre. Abban az esetben, ha az eredmény nem teljesen egyértelmű, akkor a felhasználónak egy kép alapú tesztet is ki kell töltenie.

Az egyéb CAPTCHA változatok közé tartozik a kirakó alapú CAPTCHA, ahol például különböző matematikai kifejezéseket kell megoldania a felhasználónak a kognitív képességeiket felhasználva. Az ilyen tesztekben elsősorban a kirakó darabjain szereplő különböző mintákat kell beazonosítani, majd azokat a megfelelő helyre rakosgatni. Hátránya, hogy az ilyen logikai tesztek leggyakrabban szótár alapú támadások segítségével próbálják megfejtetni, ezáltal a különböző kérdésekre a válaszokat már viszonylag könnyen meg tudják adni.

Adathalmaz felépítése és elemzése

A felhasznált adathalmazban összesen 1040 darab png kiterjesztésű és 30 darab jpg kiterjesztésű CAPTCHA kódot tartalmazó kép található, amelyeken a szöveg 5 darab alfanumerikus karakterből áll. Ezek a szövegek összesen 36 féle különböző betűből és számból épülhetnek fel: abcdefghijklmnopqrstuvwxyz0123456789.

Azonban szöveges CAPTCHA tesztek generálásakor általában bevett szokás, hogy a **0**-át, az **1**-et, az **o** betűt, az **i** betűt, a **z** betűt és még néhány másik karaktert kihagyják a lehetséges karakterek közül, hiszen ezek a karakterek nem csak a gépek, de még az emberek számára is megtévesztőek tudnak lenni.



1. ábra: Véletlenszerűen kiválasztott CAPTCHA képek az adathalmazból

A képek mérete 200x50-es, melyek mindegyikén valamiféle zajt alkalmaztak, hogy nehezebb legyen a felismerésük, mint például az egymásba logó elmosódott karakterek vagy a szövegen keresztül húzott vonalak alkalmazása.

A tanító adathalmaz 856 képet tartalmaz, a validációs adathalmaz 107 képet, illetve a teszt adathalmaz is 107 képet tartalmaz.

	Adathalmaz mérete
Tanító adathalmaz	856
Validációs adathalmaz	107
Teszt adathalmaz	107

2. tábla: A különböző adathalmazok mérete

Gépi tanulási modell létrehozása

A modell felépítése

A Keras által létrehozott modell különböző rétegekből tevődik össze. Az első réteg egy bemeneti réteg (Input), amelynek az átadott paraméterében a hálózat mérete kerül megadásra.

```

image = keras.layers.Input((50, 200, 1))
conv1 = keras.layers.Conv2D(16, (3, 3), activation='relu',
padding='same')(image)
mp1 = keras.layers.MaxPooling2D((2, 2), padding='same')(conv1)
conv2 = keras.layers.Conv2D(32, (3, 3), activation='relu',
padding='same')(mp1)
mp2 = keras.layers.MaxPooling2D((2, 2), padding='same')(conv2)
conv3 = keras.layers.Conv2D(64, (3, 3), activation='relu',
padding='same')(mp2)
bn = keras.layers.BatchNormalization()(conv3)
flat = keras.layers.Flatten()(bn)
out = []

```

1. forrás: A modell felépítése

Ezután következik 3 darab konvolúciós réteg (*Conv2D*) létrehozása, amelyek között egy pooling réteg (*MaxPooling2D*) helyezkedik el. Ezen konvolúciós hálózatok segítségével kerül feldolgozásra a bemenetként kapott többdimenziós felépítésű adatok, ahol a neuronok az előző réteg egy kisebb tartományával vannak összekötve, így csökkentve a paraméterek számát.

A pooling rétegek segítségével (*MaxPooling2D*) a paraméterek és számítások mennyisége csökkenhető, mely segítségével a túltanulás mértéke is jobban szabályozható.

Majd ezeket követi egy *BatchNormalization* réteg, ahol minden egyes kimenethez egymástól függetlenül normalizálja a rétegekimeneteket, illetve egy *Flatten* réteg, aminek a feladata pedig, hogy mielőtt átadja a *Dense* rétegek számára az értékeket, azokat többdimenziósból egy dimenzióssá alakítsa át. Ez a réteg már nem tartalmaz tanítható paramétereket.

Végezetül a *Flatten* rétegből 5 féle elágazás jön létre. Minden elágazásban 2 *Dense* és 1 darab *Dropout* réteg helyezkedik el, és mindegyik elágazás a szöveges CAPTCHA tesztnek egy karakterét prediktálja meg. A *Dropout* rétegek használatával lehetőség van a hálózat optimalizálására, azonban a túl magas érték megadása a modell tanításában gondokat okozhat. Végül a hálózat kimenetét az utolsó réteg fogja adni.

```

for i in range(5):
    den1 = keras.layers.Dense(64, activation='relu')(flat)
    drop1 = keras.layers.Dropout(0.5)(den1)
    den2 = keras.layers.Dense(36, activation="sigmoid")(drop1)
    out.append(den2)

```

2. forrás: A különböző elágazások létrehozása

Adat augmentációs modell

```
image = keras.layers.Input((50, 200, 1))
zoom = keras.layers.experimental.preprocessing.RandomZoom(0.3)(image)
rotation = keras.layers.experimental.preprocessing.RandomRotation(0.1)(zoom)
```

A kétféle modell közül, amelyeket használtam, az egyiket adat augmentációs módszerekkel egészítettem ki, amelynek implementálását csak a fent látható három sorral egészítettem ki.

Az adat augmentáció egy olyan téma, amely az adatok manipulálását jelenti a tanuló algoritmusok teljesítményének javítása érdekében. Egy adat augmentációs technika lehet például egy adat képének forgatása, elforgatása, nyújtása, vagy átméretezése. A cél az, hogy az algoritmusnak több változatos adatra legyen lehetősége tanulni, és ezáltal jobban tudja generalizálni a bemenő adatokat.

A modell tanítása

A modell tanítását a **fit()** metódus meghívásával lehet elindítani. Első paraméterként megadásra kerül a tanító adathalmaz, majd a batch paraméter mérete, aztán a validációs adathalmaz kerül átadásra. Ezt követi a tanítási egységek (*epoch*) számának meghatározása, míg végül a verbose paraméter is kap egy opcionális értéket, attól függően, milyen részletességgel szeretnénk nyomon követni a tanítási folyamatokat.

```
hist = model.fit(train_X, [train_Y[0], train_Y[1],
                           train_Y[2], train_Y[3], train_Y[4]],
                 batch_size=32,
                 validation_data=(validation_X,
                                [validation_Y[0], validation_Y[1],
                                validation_Y[2], validation_Y[3],
                                validation_Y[4]]),
                 epochs=50, verbose=2)
```

3. forrás: A fit() metódus felparaméterezése

A 3. ábrán látható a neurális hálózat teljes modellje, a különböző rétegek sorrendje és egymáshoz való csatlakozásuk, illetve a kapott bemenetek és átadott kimenetek értékei.



3. ábra: A neurális hálózat modelljét tartalmazó rétegek

A különböző rétegek létrehozása után következik a tanítási folyamat konfigurálása, ahol a **compile()** metódusnak három paraméter kerül átadásra. Az első paraméterként a veszteségfüggvény (*loss*) lett megadva, majd az optimalizáló (*optimizer*) paraméter értéke, végül pedig a mérés módja (*metric*), vagyis, hogy a tanítás eredményessége mi alapján jelenjen meg.

```
model = keras.Model(image, out)
model.compile(loss='categorical_crossentropy',
              optimizer="adam",
              metrics=['accuracy'])
```

4. forrás: A modell konfigurációja

A veszteség mértéke a categorical crossentropy függvény segítségével került kiszámításra, hiszen jól illeszkedik a különböző osztályozási feladatokra. A függvény a címkék (label) és az előrejelzések (prediction) közötti veszteséget számítja ki az alábbi képlet segítségével:

$$loss = -\sum_{i=1}^n y_i \cdot \log \log \hat{y}_i,$$

ahol az n a skaláris értékek száma a modell kimenetében, a $\hat{y}_i$ az i -edik skalár érték a modell kimenetében, a y_i pedig a vele megegyező célérték.

A veszteség mértékének csökkentése érdekében a modellekben az adaptive momentum (adam) optimalizálót használtam, amelyben az RMSProp és a Momentum optimalizáló algoritmusok kerültek kombinálásra a következő lépésekben:

1. $v_k = \beta_1 \cdot v_{k-1} + (1 - \beta_1) \cdot J\theta_k$, ahol $\beta_1 \in [0, 1]$
2. $(\tau_k)_{(i)} = \beta_2 \cdot (\tau_{k-1})_{(i)} + (1 - \beta_2) \cdot J\theta_k$, ahol $\beta_2 \in [0, 1]$
3. $(\theta_k)_{(i)} = (\theta_{k-1})_{(i)} - \alpha \cdot (v_k)_{(i)} / (\sqrt{(\tau_k)_{(i)}} + \epsilon)$, ahol $\alpha \in \mathbb{R}_{++}$

Az adam optimalizáló ötvözi a két módszer előnyét, hiszen simítja a lépési irányt, mint a Momentum, illetve adaptálódik a hiba felülethez, mint az RMSprop.

Futtatási paraméterek

Threshold: A threshold képi transzformáció egy olyan módszer, amely a kép szürkéskáláját bináris képpé alakítja, azaz a kép két állapotot vehet fel: fekete és fehér. Ennek a transzformációnak a segítségével könnyen kiemelhetők a képen található kontrasztos területek, például a textúra vagy a szegmensek. A threshold képi transzformáció során a kép szürkeárnyalatait összehasonlítják egy küszöbértékkel, és azok a pixelek, amelyek szürkeárnyalata nagyobb, mint a küszöbérték, fehérek, míg azok, amelyek alacsonyabbak, feketék.

Erosion: Az erosion képi transzformáció egy olyan képfeldolgozási technika, amely a kép szegmentációját célozza. Egy bináris képen az erosion egy olyan művelet, amely csökkenti a kép kontrasztját, és kicsinyíti a kép kontúrjait.

Ennek eléréséhez, a kép minden pixeljét a körülötte lévő pixel-struktúrával (kernel) összehasonlítjuk, ha a kernel valamelyik pixel-je fekete akkor azt az adott pixel-t feketévé teszi, ezzel a kép kontúrjait eltávolítva.

Ezt az operációt általában többször megteszük, így nagyon kicsi objektumok is eltűnnek a képen.

Ezt az eljárást alapvetően szegmentációs feladatokban használják, mint például az objektumok körvonalainak eltávolítása, vagy a felesleges részletek eltávolítása a képről

Dilation: A dilation képi transzformáció egy olyan képfeldolgozási technika, amely a kép kontúrjait és a képen lévő részleteket növeli. A dilation egy olyan művelet, amely növeli a kép kontrasztját és növeli a kép kontúrjait.

Ez a transzformáció egy bináris képen a kép minden pixeljét a körülötte lévő pixel-struktúrával (kernel) összehasonlítjuk. Ha a kernel valamelyik pixel-je fehér, akkor az adott pixel-t fehérre teszi. Ennek az eredménye, hogy a kép kontúrjai vastagabbak lesznek.

Ezt az eljárást alapvetően a szegmentációs feladatokban használják, például a kép objektumainak körvonalainak megerősítéséhez vagy a kép kontrasztjának növeléséhez. A dilation gyakran használják az erosion után, hogy visszaállítsa a kép kontúrjait.

Szegmentáció: A kép szegmentáció egy olyan képfeldolgozási technika, amely a kép megosztására szolgál különböző régiókra. A szegmentáció célja a kép részleteinek és jellemzőinek meghatározása, amelyek fontosak a kép feldolgozása és értelmezése szempontjából.

A kép szegmentáció számos módszert alkalmazhat, például a szürkekép-küszöbölés, az alak és mozgás alapú, vagy a színes kép-klaszterezés, amelyek mindegyike a kép részleteinek meghatározására irányul.

Az első futtatás során az összes kombinációját létrehoztam a normál, adat augmentációs, threshold, erosion, dilation, szegmentáció paramétereknek, majd ezekkel a paraméterekkel az adathalmazon végrehajtottam a képi transzformációkat. Először az összes kapott transzformált kép halmazra lefuttattam a neurális hálón egyszer. Ami 191 futást jelentett. Az adat augmentációs transzformációk eredményeit egy külön fájlba gyűjtöttem, és ezután azokat külön kezeltem. A kapott eredményeket sorba rendeztem a tanulási találat alapján és kiválasztottam az első 20 legjobb eredményt, hogy ezután csak a legjobb eredményekkel foglalkozzak tovább. Az adat augmentációs modell hosszabb futási ideje miatt azokból csak a legjobb 10 eredményt választottam ki. Ezután a kiválasztott legjobb eredményekhez tartozó transzformált kép halmazokat lefuttattam úgy, hogy a 20.-at 3-szor 19.-et 4-szer és így tovább az elsőt 23-szor.

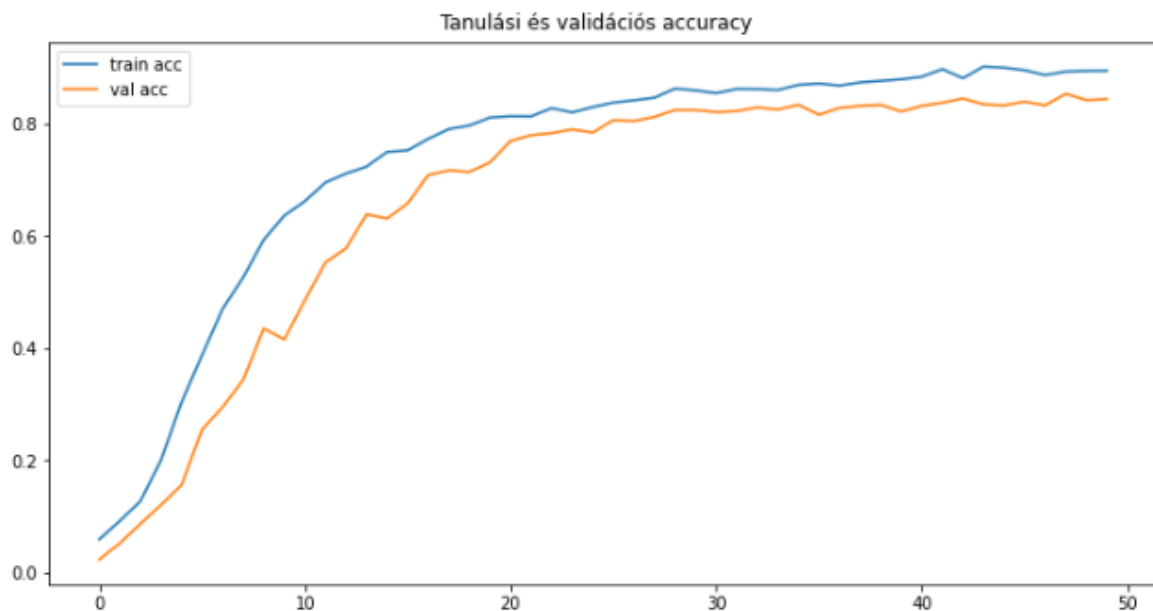
0.6303333461284637	image_data_generator_otsu_threshold_segmentation
0.5719999969005585	image_data_generator_otsu_gaussian_threshold_dilation_segmentation
0.5656666696071625	image_data_generator_otsu_gaussian_threshold_erosion_segmentation
0.55866666882038116	image_data_generator_adaptive_threshold_erosion_segmentation
0.5543333172798157	image_data_generator_dilation_dilation2_segmentation
0.5510000228881836	image_data_generator_adaptive_threshold_erosion
0.5503333389759064	image_data_generator_otsu_gaussian_threshold
0.5480000138282776	image_data_generator_adaptive_threshold_dilation_erosion
0.5420000076293945	image_data_generator_otsu_threshold_dilation
0.5230000257492066	image_data_generator_otsu_gaussian_threshold_dilation2
0.5223333418369294	image_data_generator_otsu_threshold_dilation_erosion_segmentation
0.507000008225441	image_data_generator_adaptive_threshold_dilation_dilation2_segmentation
0.5060000061988831	image_data_generator_adaptive_threshold_dilation_erosion_segmentation
0.49400001764297485	image_data_generator_otsu_threshold_erosion
0.4749999910593033	image_data_generator_otsu_threshold_dilation_dilation2_segmentation
0.4699999988079071	image_data_generator_otsu_gaussian_threshold_dilation_erosion
0.45666668713092806	image_data_generator
0.4559999942779541	image_data_generator_erosion_dilation2_segmentation
0.45233333110809326	image_data_generator_otsu_gaussian_threshold_dilation_dilation2
0.3930000066757202	image_data_generator_otsu_gaussian_threshold_erosion_dilation2

4. ábra: Első 20 legjobb adat augmentációs eredmény

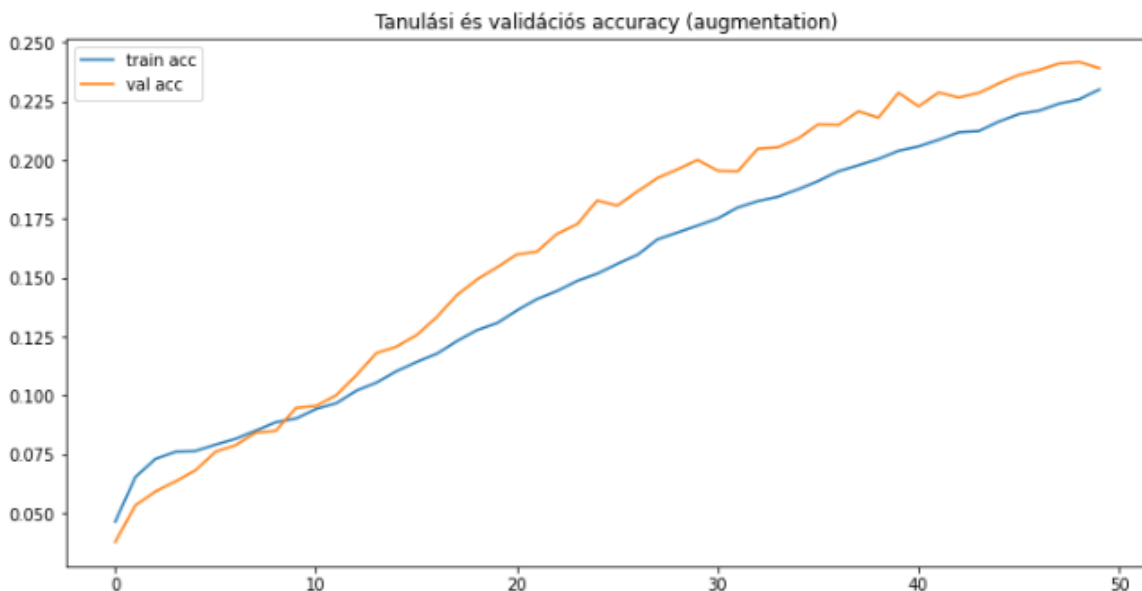
0.9806666851043702	otsu_gaussian_threshold
0.9773333430290222	normal_otsu_gaussian_threshold_dilation
0.9710000038146973	adaptive_threshold_dilation2
0.9706666588783264	normal_otsu_threshold_dilation2
0.9686666607856751	normal_otsu_gaussian_threshold_dilation_erosion_dilation2
0.9683333277702332	otsu_threshold_erosion
0.9643333196640015	normal_otsu_gaussian_threshold_dilation_dilation2
0.96266667842865	normal_otsu_threshold_erosion
0.9613333225250245	normal_otsu_threshold
0.9546666741371155	otsu_threshold
0.9543333292007447	normal_adaptive_threshold_dilation
0.9523333191871644	otsu_threshold_dilation
0.9486666798591614	erosion_dilation2
0.9396666646003723	normal_adaptive_threshold_erosion_dilation2
0.9386666536331176	adaptive_threshold_erosion
0.9383333444595336	normal_adaptive_threshold_dilation_erosion
0.9376666665077209	adaptive_threshold_dilation_dilation2
0.9329999923706055	normal_adaptive_threshold_erosion
0.9319999933242797	normal_otsu_threshold_erosion_dilation2
0.9306666731834412	otsu_gaussian_threshold_dilation_dilation2

5. ábra: Első 20 legjobb normál paraméteres eredmény

Első futási eredmények összehasonlítása



6. ábra: Tanulási és validációs találati átlag normál paraméterek esetében

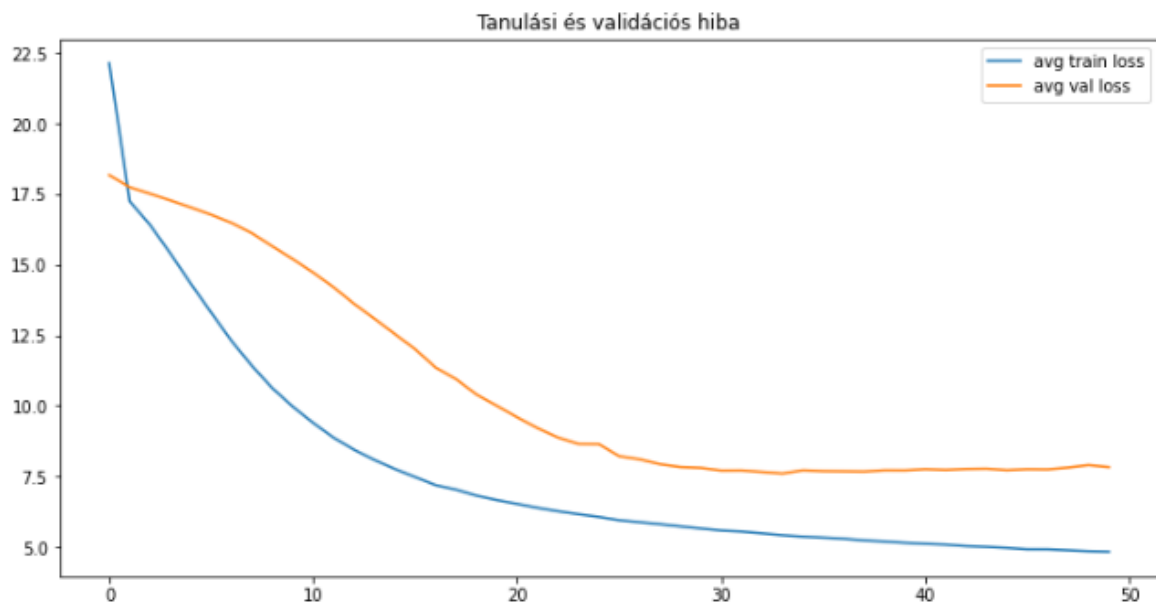


7. ábra: Tanulási és validációs találati átlag adat augmentációs paraméterek esetében

A fenti ábrákon a tanulási és validációs eredmények átlagai vannak összesítve, ami a 6. ábrán 122 futás eredménye, mert a 191 futásból külön kezeltük az adat augmentációs paraméterekkel rendelkező 69 futást, ami a 7. ábrán található. A 6. ábrán azt láthatjuk, hogy átlagosan jó tanulási és validációs eredmények születtek. A tanulási találat végig megelőzi a validációs találatokat. A 7. ábrán azt láthatjuk, hogy átlagosan véve nagyon rossz eredményeket értek el az adat augmentációs kép halmazok. Viszont ami érdekesség, hogy itt a validációs találat megelőzi a tanulási találatok átlagát. Ez azért lehetséges, mert az adat augmentáció használatával növeljük

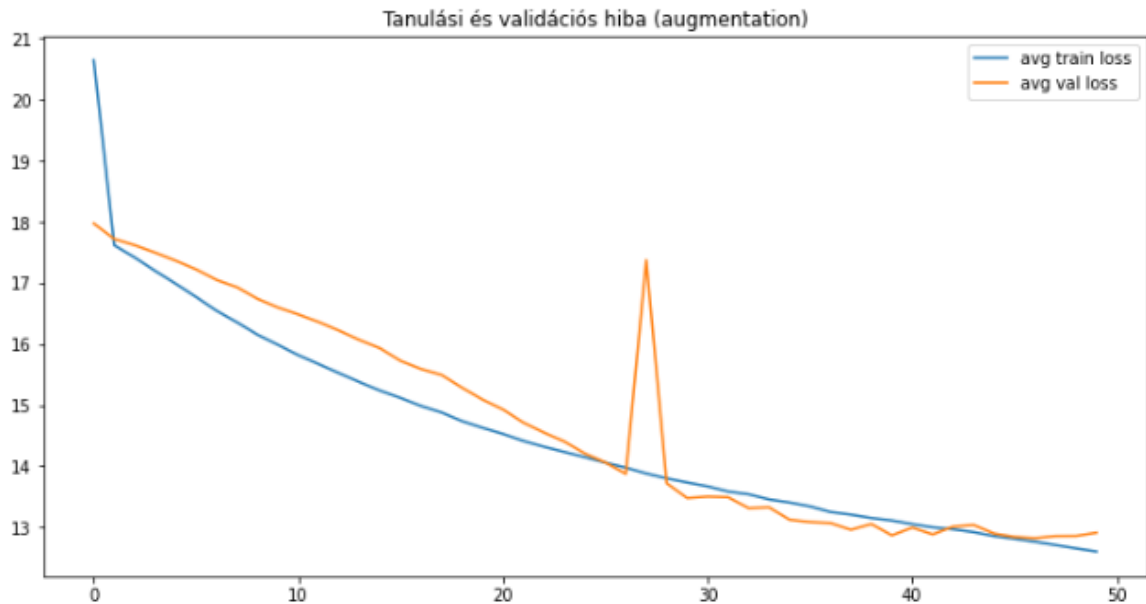
az adatok mennyiségét, ezáltal a modellnek több információ áll rendelkezésére a tanulás során. Az adat augmentáció során az eredeti adatokat módosítjuk oly módon, hogy azokat újabb és változatosabb adatokként használhassuk a tanuláshoz. Ennek eredményeként a modellnek jobban általánosítani kell, mert a tanulási adatokban szereplő változatosság növekszik.

Egy másik ok is lehet, amiért az adat augmentációs modell esetén a validációs találati eredmények megelőzhetik a tanulási találati eredményeket az, hogy az overfitting elkerülésére végett egyes konfigurációkban az adataugmentációt kizárólag a validációs adatoknál alkalmazzák, így ez szintén hozzájárulhat a jobb validációs eredmények eléréséhez.



8. ábra: Tanulási és validációs hiba átlagok normál paraméterek esetén

A 8. ábrán látható a tanulási és validációs hiba arányának alakulása az 50 epoch során az adat augmentáció nélküli kép halmazok futtatásakor.

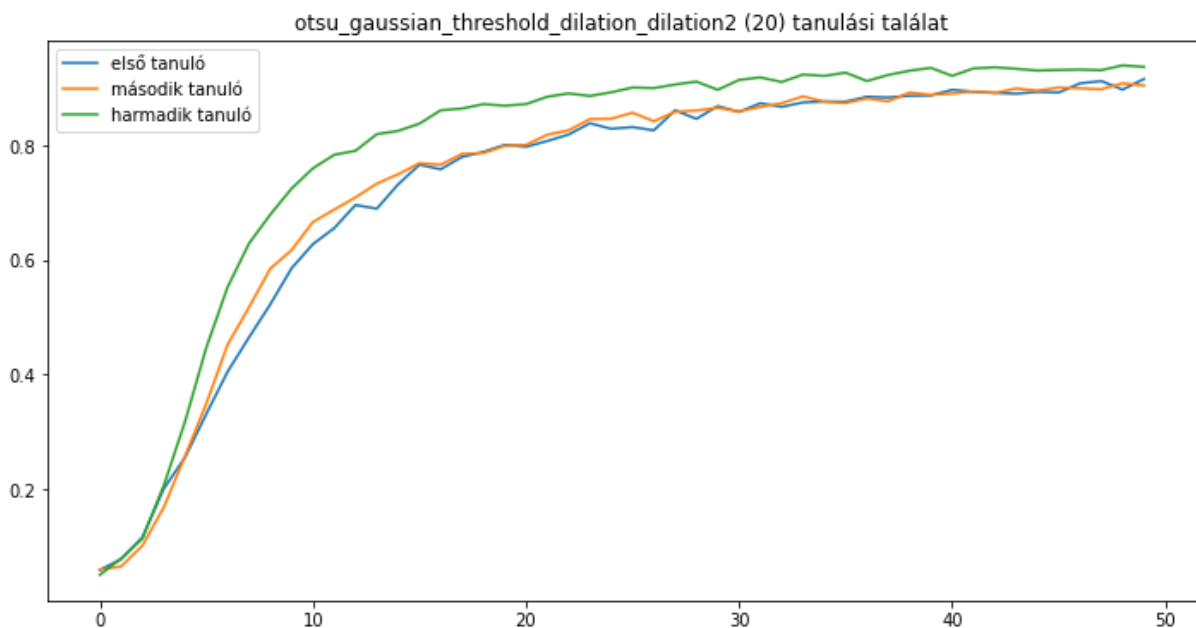


9. ábra: Tanulási és validációs hiba átlagok adat augmentációs paraméterek esetén

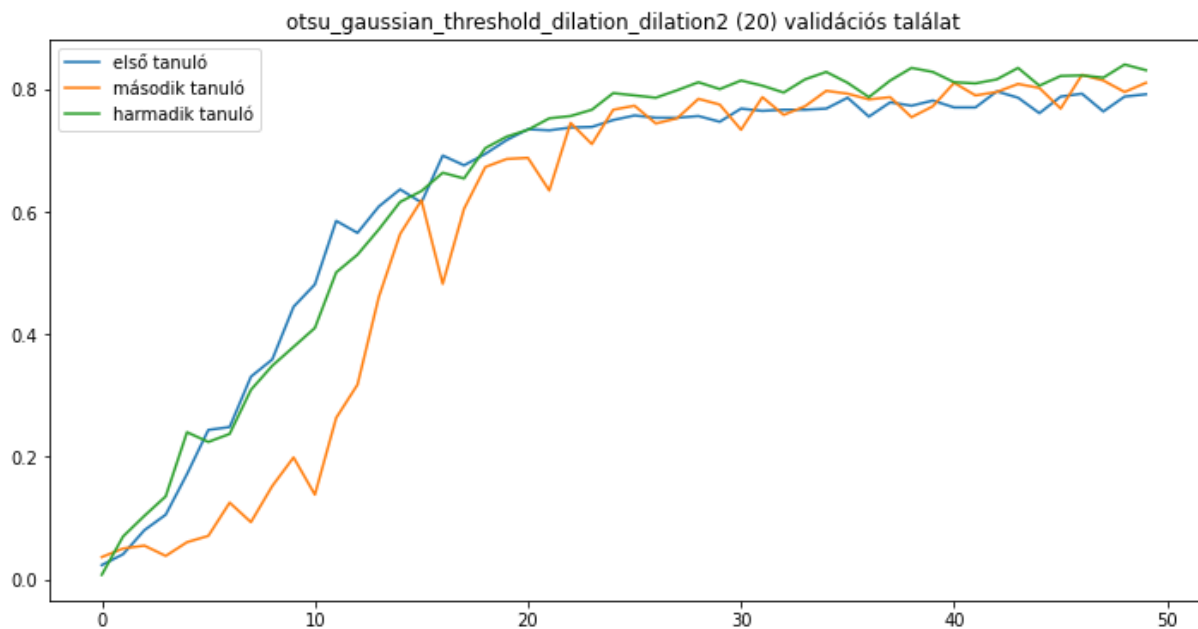
A 9. ábrán látható a tanulási és validációs hiba arányának alakulása az 50 epoch során az adat augmentációs kép halmazok futtatásakor. Ami érdekesség lehet az a kiugró megnövekedett validációs hiba arány a 20. és a 30 epoch között.

Újrafuttatás eredmények összehasonlítása

A következő ábrákon kiválasztottam a 20. 10. és 1. helyezett tanulási és validációs találati eredményeit és azokat hasonlítom össze.

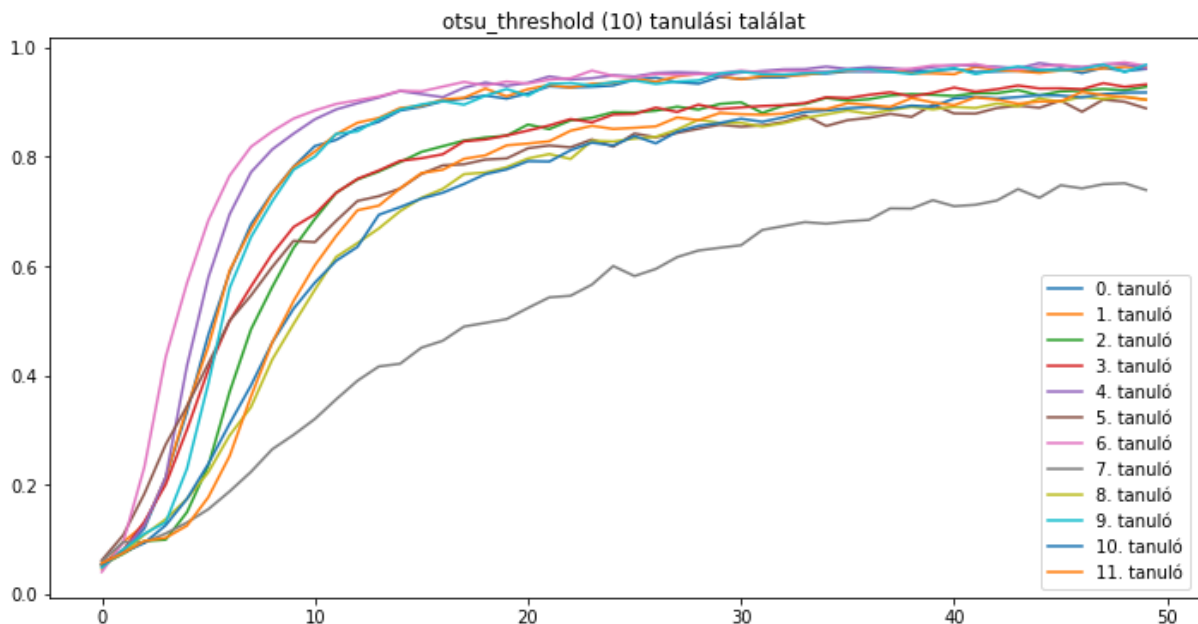


10. ábra: 20. helyezett tanulási találati arányok 3 tanulóval

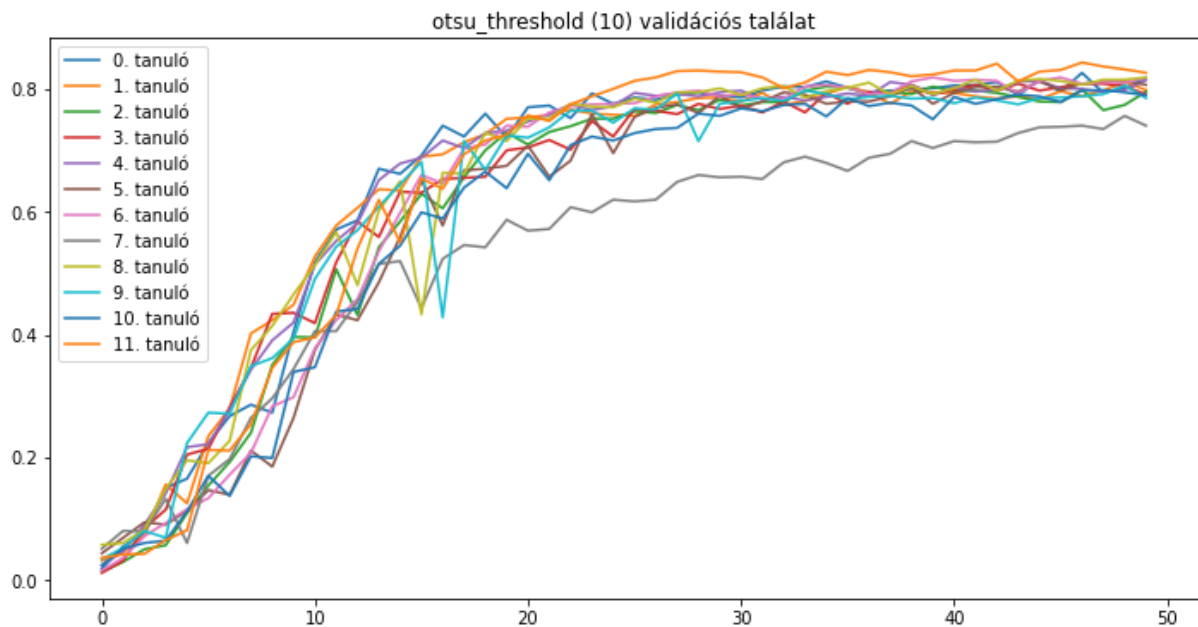


11. ábra: 20. helyezett validációs találati arányok 3 tanulóval

Az otsu gaussian threshold, dilation és dilation2 paraméterrel rendelkező képhalmazt 3 tanulóval futtattam újra. Az látható a 10. és 11. ábrákon, hogy közel hasonló eredményeket értek el a tanulók. A validációs találati eredmények sem sokkal vannak lemaradva a tanulási eredményektől.

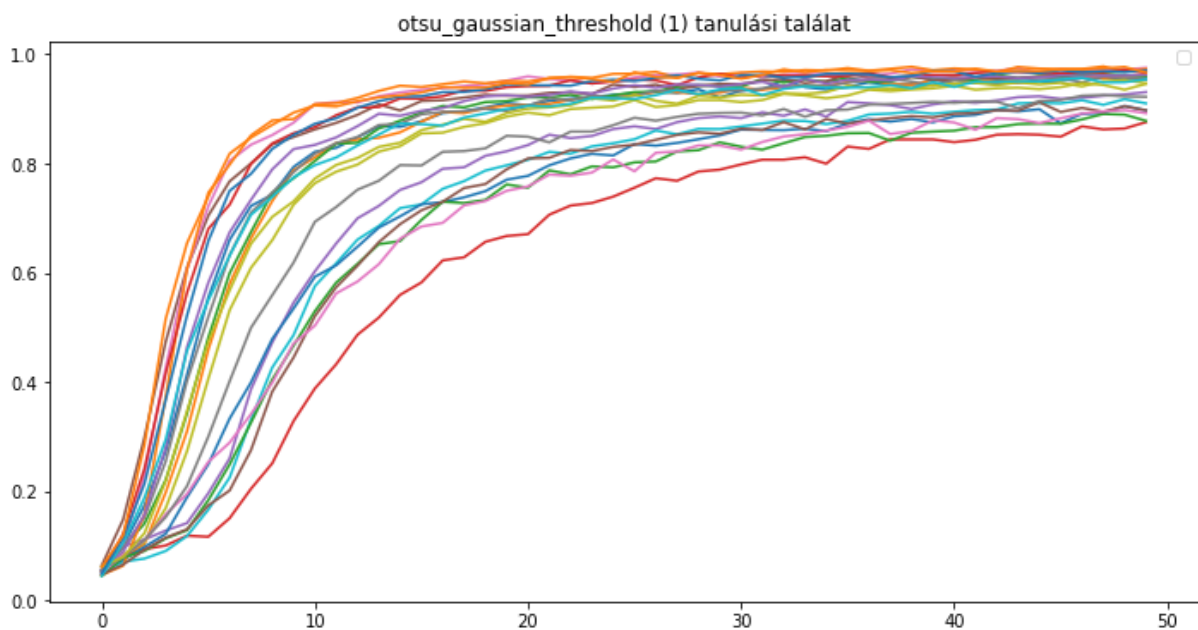


12. ábra: 10. helyezett tanulási találati arányok 12 tanulóval

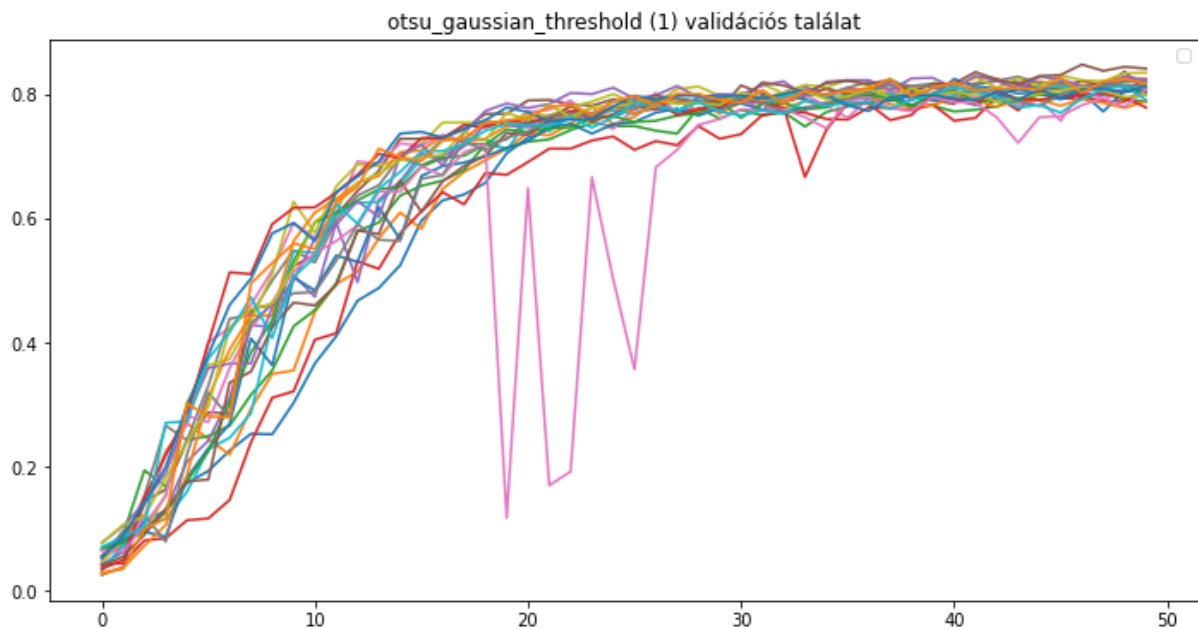


13. ábra: 10. helyezett validációs találati arányok 12 tanulóval

Az otsu threshold paraméterrel rendelkező képhalmazt 12 tanulóval futtattam újra. A 12. ábrán azt láthatjuk, hogy itt már valamivel nagyobb a szórása a tanulók eredményeinek és a 7. tanuló feltűnően rosszabb eredményt ért el a többitől. A 13. ábrán is azt láthatjuk, hogy a 7. tanuló ott is rosszabb eredményt ért el a többi tanulótól, igaz ott nem olyan nagy mértékkel tér el, mint a tanulási találat esetén.



14. ábra: 1. helyezett tanulási találati arányok 22 tanulóval

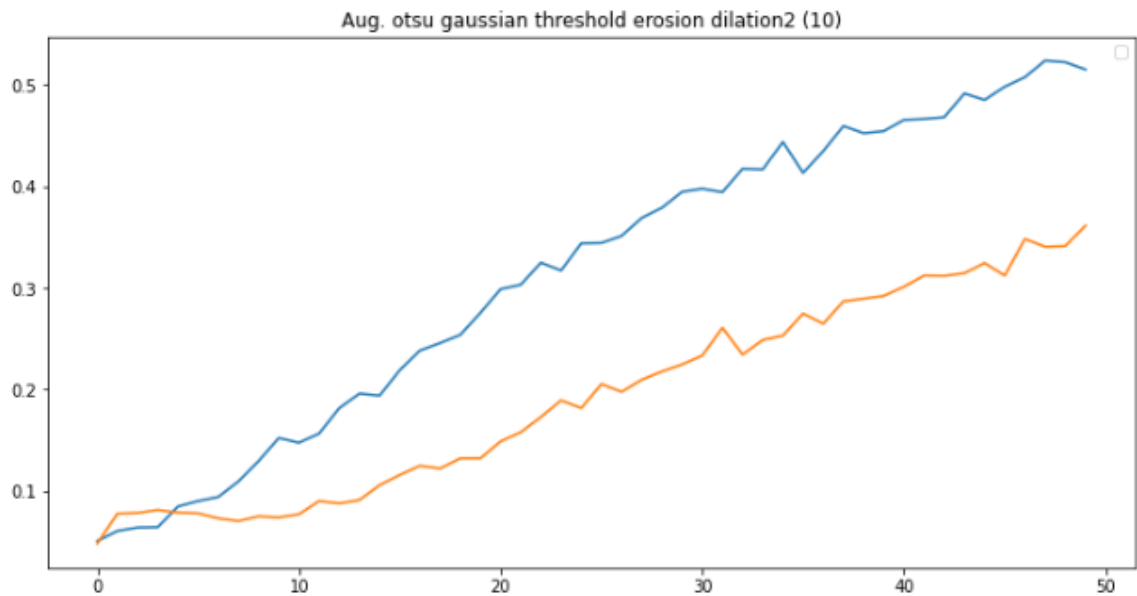


15. ábra: 1. helyezett validációs találati arányok 22 tanulóval

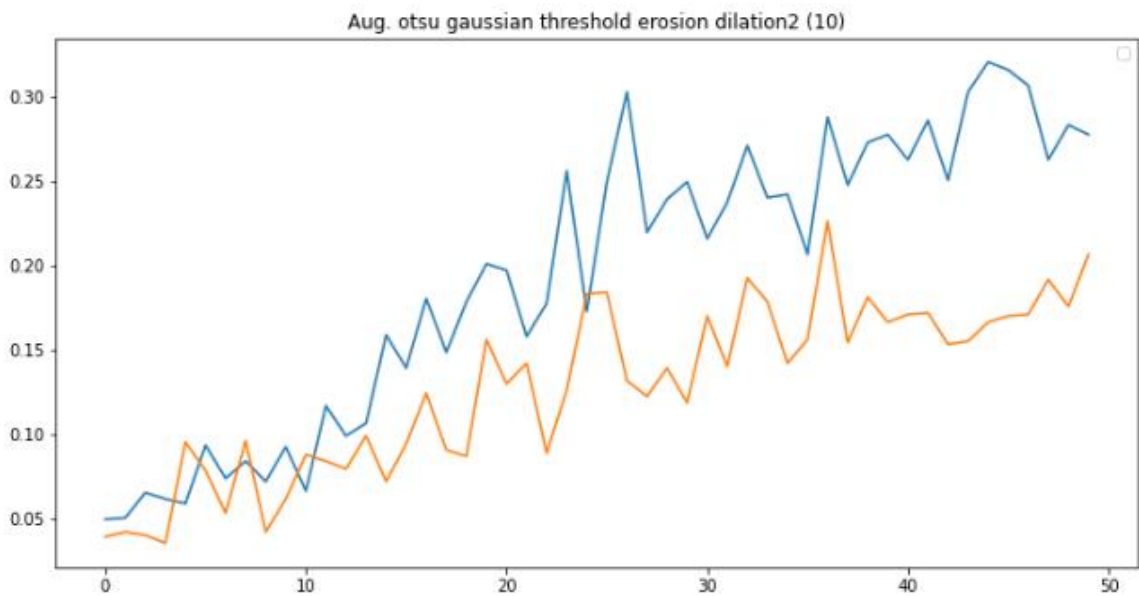
Az otsu gaussian threshold paraméterrel rendelkező képhalmazt 23 tanulóval futtattam újra. A 14. ábrán azt láthatjuk, hogy valóban úgy tűnik, hogy ez a legjobb paraméter, hiszen a sok futás ellenére is ennek a kép halmaznak a legkisebb a szórása a tanulási találatait nézve. A 15. ábrán azt láthatjuk, hogy itt még jobban közelítenek az eredmények egymáshoz, viszont a rózsaszínű vonallal jelzett tanulónál a 20. epoch körül óriási hiba keletkezett, hiszen 0,2 alá csökkent a pillanatnyi találati arány. De a 30. epoch-ra már újra meg tudta közelíteni a több tanuló eredményét.

Adat augmentációs eredmények

A következő diagramokban kiválasztottuk a 10. 5. és 1. helyezett tanulási és validációs találati eredményeit és azokat hasonlítjuk össze.

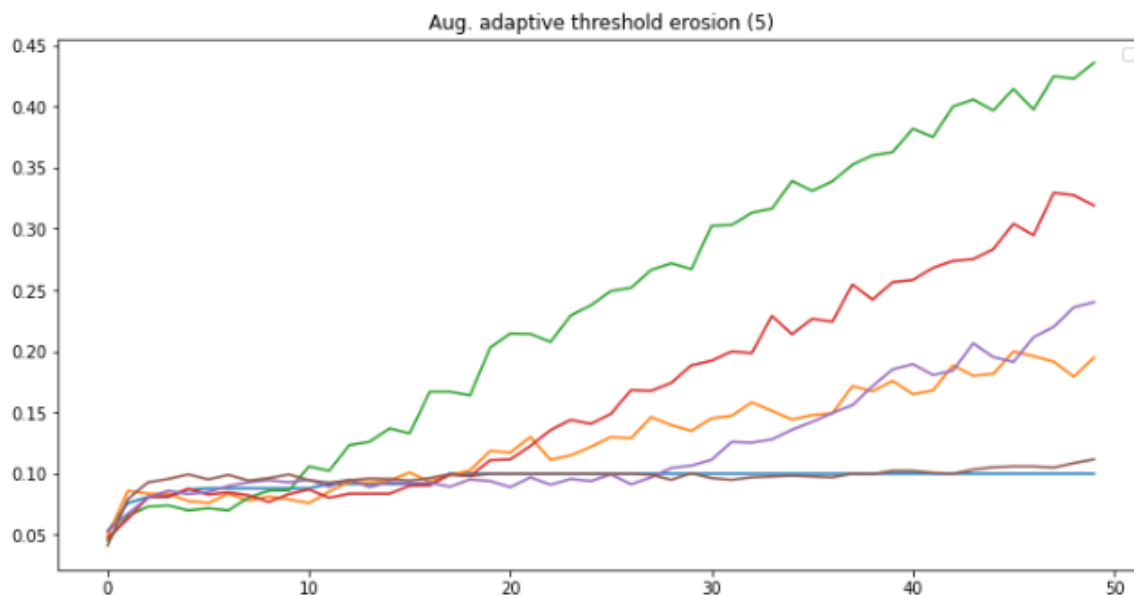


16. ábra: 10. helyezett tanulási találati arányok 2 tanulóval

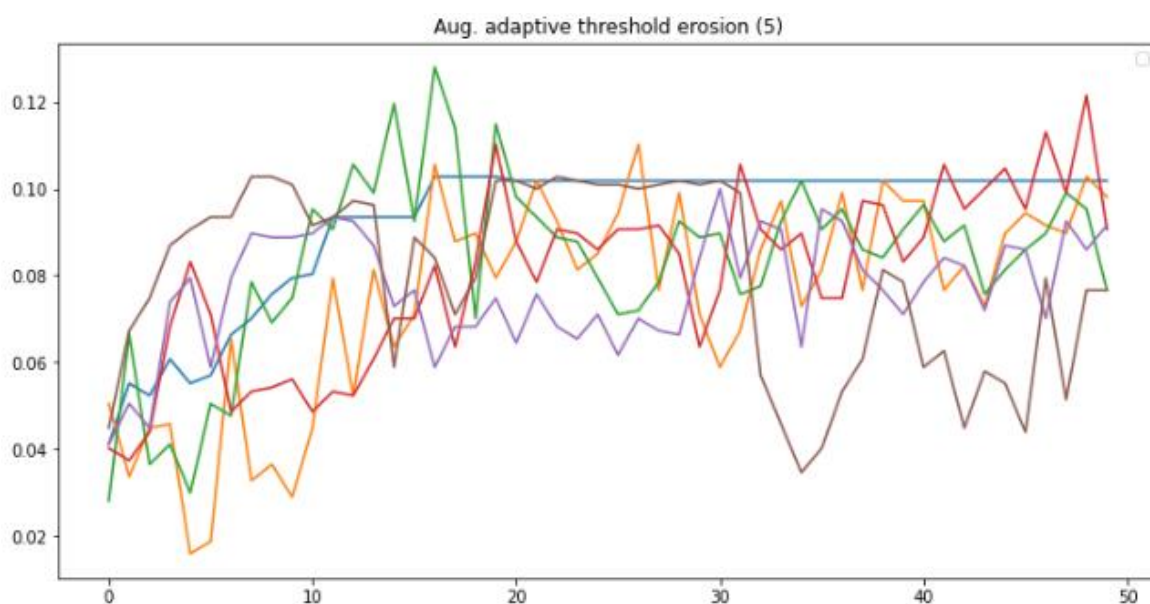


17. ábra: 10. helyezett validációs találati arányok 2 tanulóval

Az image data generator, otsu gaussian threshold, erosion, dilation paraméterrel rendelkező képhalmazt 2 tanulóval futtattam újra. A 16. ábrán az látható, hogy nem jó eredményeket értek el a tanulók, de megközelítik az első futás során elért 52 százalék körüli eredményt. A 17. ábrán az látható hogy a futás során eléggé változóak az eredmények és a végén sem értek el jó eredményt.

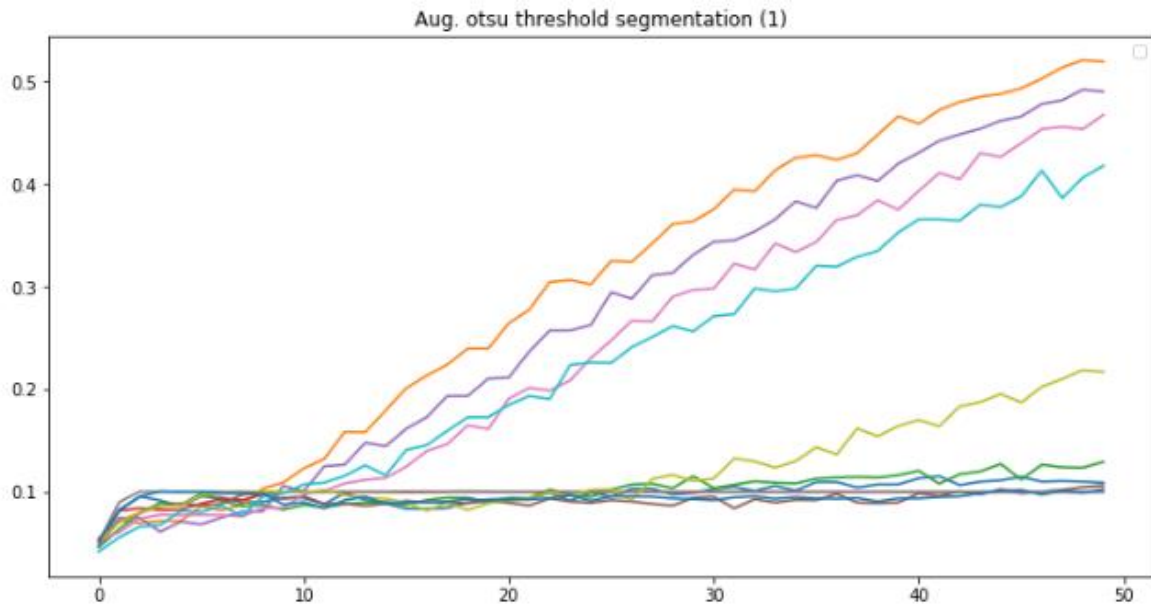


18. ábra: 5. helyezett tanulási találati arányok 7 tanulóval

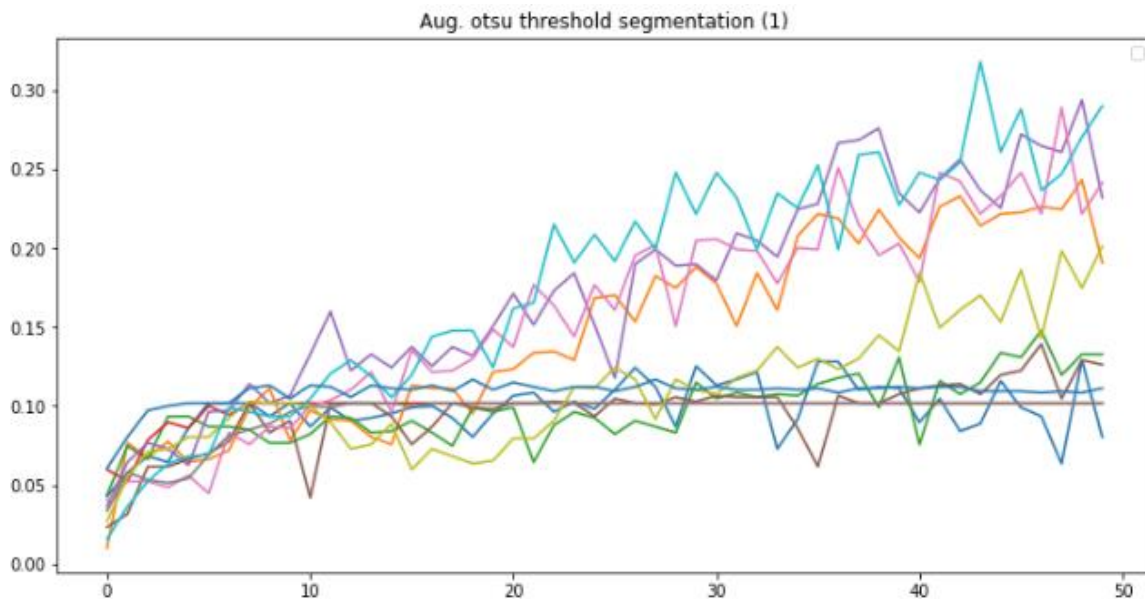


19. ábra: 5. helyezett validációs találati arányok 7 tanulóval

Az image data generator, adaptive threshold, erosion paraméterekkel rendelkező képhalmazt 7 tanulóval futtattam újra. A 18. ábrán az látható, hogy itt el sem érte a legjobb tanuló sem az előző ábrán látható, eredetileg rosszabb helyezést elérő tanulók eredményeit. Ez azt jelentheti, hogy a kiválasztás előtti első futáskor jó eredményt tudott elérni, de ezeknek a paramétereknek a használata nem kifizetődő több futtatás esetén. Az figyelhető még meg az ábrán, hogy nagyon változó eredményeket láthatunk, egy tanulónál a 10 százalékos találati eredményt sem éri el. Ez az eltérés az adat augmentáció által generált véletlenszerű kép transzformációkból adóthat. A 19.ábráról is hasonló mondható el.



20. ábra: 1. helyezett tanulási találati arányok 12 tanulóval



21. ábra: 1. helyezett validációs találati arányok 12 tanulóval

Az image data generator, otsu threshold segmentation paraméterekkel rendelkező képhalmazt 12 tanulóval futtattam újra. A 20. ábrán azt láthatjuk, hogy habár a három kiválasztott futási eredmények alapján itt született a legjobb eredmény, mégis itt is láthatunk sokszor arra példát, hogy 10 % fölé nem tud emelkedni a találatok száma. A 21. ábrán is azt láthatjuk, hogy a legjobb validációs találati eredmények itt születtek, de itt is megfigyelhető, hogy nem tud valamelyik tanuló 10 százalék fölé jutni.

Összefoglalás

Dolgozatomban egy szöveges, illetve szintelen CAPTCHA tesztek tartalmazó adathalmazon dolgoztam. Ahol a különböző karakterek azonosításához különféle paraméterekkel ellátott konvolúciós neurális hálózatot használtam fel, amelyet utána összesen öt darab teljesen összekötött réteget (*Fully Connected*) és kiejtéses réteget (*Dropout*) tartalmazó elágazásra bontottam szét. Az eredmények alapján elmondható, hogy a hálózatban használt többféle paraméter kombinációk alapján a modellek létrehozásának és tanulásának ideje, csak az adat augmentációs paraméternél nőtt meg, majdnem a háromszorosára. A tanulási és validációs eredmények is rosszabbak lettek ezzel a paraméterrel, így azt mondhatjuk, hogy 50 epoch esetén nem éri meg ennek a paraméternek a használata. Olyan adat halmazoknál éri meg az adat augmentációt alkalmazni, ahol kevesebb kép áll rendelkezésünkre a tanulásra, mert ezekkel a kép forgatás, nagyítási transzformációkkal újabb tanulási képekre tehetünk szert.

További kutatási célok

Az egyik jövőbeli tervem az adat augmentációs kép halmazokat lefuttatni nagyobb epoch számmal. Annak érdekében, hogy megnézzem mily módon alakulnak a tanulási és validációs eredmények, hiszen ahogy láthattuk az X. diagramon, a validációs találati arányok megelőzik a tanulási találati arányokat. További tervem egy saját kezűleg létrehozott szegmentáció használata, az OpenCV helyett. Ezután az így kapott eredmények összehasonlítása. Ezenkívül érdemesnek gondolom még a képi alapú CAPTCHA tesztekkel való foglalkozást is, hiszen napjainkban ezek nyújtják a legnagyobb biztonságot a weboldalakat támadó botokkal szemben. Szakmailag kihívó feladatnak tartom az ilyen jellegű adathalmazok kapott eredményeken való magas pontossági érték elérését.

Végezetül szeretném összehasonlítani az emberi és gépi tanulás pontosságát a képosztályozás mérésében, ahol az emberi és gépi teljesítő képességeket vetném össze egyenlő feltételek mellett. Az ilyen fajta vizsgálatokban az elsődleges szempont nem csak a találati arányok mérése lenne, hanem kísérletet lehetne tenni arra is, hogy megállapítsuk, hogy például melyik karaktert téveszti el legtöbbször a számítógép, illetve az ember.

Záró gondolatok

Ezúton szeretnék köszönetet mondani témavezetőimnek, Dr. Tajti Tibornak és Kovács Ádámnak, akik szakértelmünkkel, hasznos tanácsaikkal, a konzultációk során tett észrevételeikkel támogatták dolgozatom létrejöttét.

Irodalomjegyzék

1. BME-MIT. Neurális hálók tanítása során alkalmazott optimalizáció - Hadházi Dániel.
http://home.mit.bme.hu/~hadhazi/Oktatas/NN/Opt_exp.pdf [Letöltve: 2022. október]
2. Chen, J., & Luo, X., & Guo, Y., & Zhang, Y., & Gong, D. (2017). A Survey on Breaking Technique of Text-Based CAPTCHA. Security and Communication Networks.
3. Chollet, F. (2017). *Deep learning with python*. Manning Publications.
4. Deosatwar, S., & Deshmukh, S., & Deshmukh, V., & Sarda, R., & Kulkarni, L. (2020). An Overview of Various Types of CAPTCHA.
5. Kaggle. Captcha Recognition - Dhruv A. M.
<https://www.kaggle.com/code/dhruvaditya/captcha-recognition> [Letöltve: 2022. október]
6. Mahato, S., Saxena, V.P., & Bhavsar, D. (2015). A Survey of Captcha based Web and Application Security Methods and Techniques.
7. OneZero. Why CAPTCHA Pictures Are So Unbearably Depressing - Clive T.
<https://onezero.medium.com/why-captcha-pictures-are-so-unbearably-depressing-20679b8cf84a> [Letöltve: 2022. október]
8. Wilhelmy, R., & Rosas, H. (2013). captcha dataset