BGE

# BUDAPESTI GAZDASÁGI EGYETEM

# PÉNZÜGYI ÉS SZÁMVITELI KAR

## SZAKDOLGOZAT

Sipos Gergely

Levelező

Gazdaságinformatikus

Üzleti adatelemző informatikus

BUDAPESTI GAZDASÁGI EGYETEM

PÉNZÜGYI ÉS SZÁMVITELI KAR

# KÉPFELISMERÉSI TECHNIKÁK ÖSSZEHASONLÍTÁSA MÉLY TANULÁSI ÉS A HAGYOMÁNYOS MÓDSZEREK KÖZÖTT

Belső konzulens: Dr. Kovács Endre

Külső konzulens: Kovács Tibor

Sipos Gergely

Levelező

Gazdaságinformatikus

Üzleti adatelemző informatikus

2023

NYILATKOZAT

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

A szakdolgozatban felhasznált adatokat a szerzői jogvédelem figyelembevételével alkalmaztam.

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

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

Budapest, 2023 év 12 hónap 11 nap

.....

hallgató aláírása

# Tartalomjegyzék

# 1. Bevezető

## 1.1 Ismertetés és témaválasztás

Az utóbbi időben nagy reflektorfényt kapott a mesterséges intelligencia és a mély tanulási megoldások, viszont ez a fejlődés nem most kezdődött. A gépi tanuló és AI modellek, algoritmus megoldások több évtizedre tekintenek vissza. Az első neurális hálózatot 1951-ben hozták létre és az 1960-as évek végén már több kutatás is folyt olyan rendszerek megalkotására, amelynél a cél az emberi leutánzása. (Minsky & Papert, 2017)

A motivációm, ami miatt a képfeldolgozás témáját választottam abból fakad, hogy a gyakorlati felhasználása és technológiai háttere lenyűgöző. A vállalatok egyre inkább törekednek a digitális átállásra, ami magával hozza az intelligens képfelismerő rendszerek implementálását. A szakmai múltamra tekintve többségében „retail", azaz kiskereskedelmi megoldásokkal dolgoztam, innen tudnék több lehetőséget említeni, hogy miként lehetne a képfeldolgozási megoldásokat hasznosítani, alkalmazni. (Grewal et al., 2017)

- Készlet menedzsment és „Visual merchandising": A kamerák segítségével a készletszint követése, a vásárlók interakciója a boltban lévő kihelyezésekkel. (Grewal et al., 2017)
- Planogram ellenőrzés: A program könnyedén tudja biztosítani az üzletben készült képek alapján, hogy betartják-e a megegyezett kihelyezési tervet, azaz megfelelően vannak-e kihelyezve, elrendezve a termékek a boltok polcain. (Grewal et al., 2017)

Mivel nagyon sokféle problémára keresünk megoldást, így szükség van a rendelkezésre álló módszerek megkülönböztetésére. A mélytanulási megoldások nagyon jó eredményeket mutatnak, viszont előfordulhatnak olyan esetek amikor a hagyományos gépi algoritmusok rugalmasabban implementálhatók, például valamely speciális eset miatt.

A szakdolgozatom célja, hogy egy átfogó összehasonlító elemzést végezzek a konvolúciós neurális hálózatok és a hagyományos gépi tanuló algoritmusók között a képosztályozás kontextusában. A hagyományos algoritmusokból a k-legközelebbi szomszéd (kNN) és a szupport vektor gép (SVM) alkalmazhatóságát és eredményeit fogom bemutatni. A vizsgálat célja, hogy rávilágítson a különböző megoldások relatív erősségeire és korlátaira, valamint az implementálhatóságára. Kísérleti adathalmazként a CIFAR-10 és CIFAR-100 (Krizhevsky, 2009) adatbázist alkalmazom, mivel ezek remekül előkészített és tiszta adatállományok,

amelyek jó példák a kis és nagy adathalmazokra. A szakdolgozatomban az alábbi fő kérdéskörökre fogom keresni a választ. (I. Goodfellow et al., 2016)

- Az adatállomány karakterisztikáinak hatása: Hogyan befolyásolja a mérete, összetétele és más egyedi jellemzői a teljesítményt és az általánosítási képességét? Kísérletezéssel is célom feltárni a jellemzők és a modellek közötti kapcsolatot a különböző teljesítmény metrikákon keresztül és ezáltal egy általános útmutatást nyújtani a modellválasztáshoz.
- A képkategóriák közötti hatékonyság különbség: A két megközelítés között melyik mutat jobb eredményeket a különböző képkatógriákon. Van-e egyáltalán hatása a kategóriák változásának? Ha van hatása indokolt lehet-e egy hibrid megoldás implementálása.
- Általános hatékonysági mutatók elemzése: Hogyan különböznek a mélytanulási és a hagyományos gépi tanuló modellek értékelési metrikái. A munkám során elemezni fogok több mérési mutatót, mint például „F1-score", „precision", „recall". Valamint megvizsgálom, hogyan változnak az eredményeink különböző metrikákat figyelembe véve.

# 2. Elméleti Háttér

## 2.1 Digitális képek

A digitális képek matematikailag is leírhatók. A fekete-fehér / szürkeárnyalatos kép leírható egy valós értékű függvényként. Képzeljünk el egy teljes képet egy lapos síknak, amelyet matematikailag $\Omega \subseteq R^2$ -ként ábrázolunk. Itt $\Omega$ a kép terét jelenti, $R^2$ pedig ezt a lapos síkot jelöli. Ennek a síknak bármely adott pontjára meghatározható annak fényereje vagy szürke árnyalata. Ezt az összefüggést a következőképpen tudjuk leírni: $f : \Omega \to R_0^+$. Tulajdonképpen a kép bármely tetszőleges pontjára, amelyet x és y koordináták határoznak meg, az $f\ (x, y)$ megadja, hogy mennyire világos vagy sötét a képpont. A színes képek egy kicsit összetettebbek, a vörös, a zöld és a kék különböző árnyalatait keverik ki, hogy abból különböző színeket hozzanak létre. A fekete-fehér képekhez képest, ahol csupán egy fényerőség értékünk volt, most három különböző értéket kapunk a vörös, kék és zöld intenzitáshoz. Ezt a következőképpen tudjuk ábrázolni: $f : \Omega \to (R_0^+)^3$. Ez azt jelenti, hogy bármely $f\ (x, y)$ -el ábrázolt ponthoz három fényerő-komponens tartozik, a $f\ \mathrm{R}(\mathrm{x}, \mathrm{y})$ a piroshoz, $f\ \mathrm{G}(\mathrm{x}, \mathrm{y})$ a zöldhöz, $f\ \mathrm{B}(\mathrm{x}, \mathrm{y})$ a kékhez. (Ballard et al., 1982; Louridas, 2020)

Ezt követően a matematika ábrázolás után áttérek a RGB színmodell gyakorlatibb alkalmazására és ábrázolására. A színes digitális képeket legtöbbször az RGB (red mint piros, green – zöld és blue – kék) modellel jellemezzük. Ahogy már a matematikai leírásban említettem, a választott pixel három intenzitásértéke kerül továbbításra. Ezek az értékek minden egyes pixel koordinátához (x, y) hármasával jellennek meg. Igy a M x N dimenziójú kép az alábbiként írható le. (Szeliski, 2022)

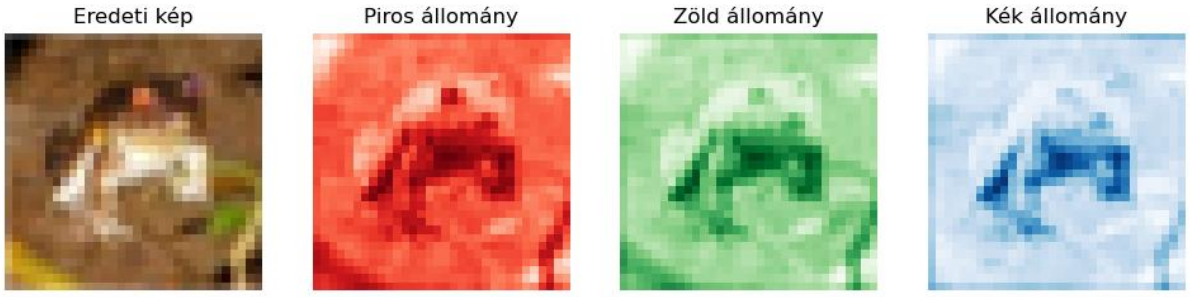
$$\mathrm{K}(\mathrm{x},\mathrm{y}) = \big(\mathrm{R}(\mathrm{x},\mathrm{y}), \mathrm{G}(\mathrm{x},\mathrm{y}), \mathrm{B}(\mathrm{x},\mathrm{y})\big)$$

$$0 \leq \mathrm{x} < \mathrm{M}, 0 \leq \mathrm{y} < \mathrm{N}$$

$$0 \leq R \leq 255, 0 \leq G \leq 255\ , 0 \leq B \leq 255$$

Az R, G és B a színcsatornák intenzitás mátrixát jelöli. Az intenzitásokat álltalában 8 bites értékként kvantálják, emiatt az értékek 0 és 255 között változhatnak. 0 érték azt jelenti, hogy nincs hozzáadott értéke, míg a 255-ös érték az úgynevezett teljes telítettséget jelöli. (Ballard et al., 1982)

Alább látható a CIFAR-10 (Krizhevsky, 2009) adatkészlet első képe, az eredeti majd ezt követően a különböző színcsatornák képei.




*ábra 1 A színcsatornák szerinti bontás (forrás: eredeti kép CIFAR-10 dataset és python manimuláció)*

Fontos megemlíteni, hogy az RGB színtéren kívül vannak másfajták is. Mint például a HSV (Hue, Saturation, Value) vagy az YCbCr színcsatorna. (Szeliski, 2022)

---

## 2.2 A gépi tanulás bevezető

A gépi tanuló algoritmusok felügyelt és nem felügyelt tanulásra oszthatók. A felügyelt tanulásnál („supervised learning”) az algoritmus párosított bemeneteket és kimeneteket használ fel. Ezzel szemben a felügyelt tanulásra („unsupervised learning”) jellemző, hogy statisztikai

mintákákat adunk meg a modellnek, viszont megcímkézett kimenetek nélkül. A felügyelt tanulás a bemeneti párokat és a hozzájuk tartozó kimeneti cél értékeket használja fel. Ilyenkor a cél, hogy az algoritmus paramétereit úgy állítsuk be, hogy előrejelzései szorosan illeszkedjenek az adott célértékhez. Ezek a predikciók lehetnek diszkrét címkék, amelyek egy adott osztályból jönnek, vagy folytonos, vektor értékek. (I. Goodfellow et al., 2016)
A tanulási fázis közben az adathalmaz alapos elemzésére és legtöbbször több iteráción keresztül a modell finomítására van szükség. Ezután a modell készen áll, hogy előjelzéseket adjon új és még nem látott adatokkal kapcsolatban. Bár ezt a szakaszt tesztelési fázisnak nevezzük viszont fontos, hogy ne csak a teszt készlet eredményeit vegyük figyelembe, hanem egy olyan modellt tudjunk megépíteni, amely remekül általánosít és emiatt számtalan lehetséges bemenetre fel van készítve, hogy ne essünk a túl illesztés hibájába.
Az elsődleges cél a modellek tanítása folyamán, hogy a prediktált és a tényleges célérték között összhang legyen. Tekintettel az adatainkban rejlő bizonytalanságra, ami jelentheti az adatok zajosságát, hiányosságát vagy félreérthetőségét, a legfőbb feladat az algoritmus hatékonyságának növelése, vagyis a hibák csökkentése. Az L veszteség függvény („loss function”) képes számszerűsíteni a modell $f(x_i, w)$ előrejelzéséhez kapcsolódó „költséget” (cost) egy adott $x_i$ bemenet és $w$ modellparaméter esetén, ha a kívánt célérték $y_i$ (Szeliski, 2022). Az általános költségfüggvény a következő (Leeuwen, 1998).

$$L = Cost(y_i, f(x_i, w)).$$

Az osztályozási feladatokban a tipikus cél a téves osztályozás arányának csökkentése (Szeliski, 2022). Alapvetően az osztályozás ideális forgatókönyve, ha a hibákat egységesen kezeljük. Ez azt jelenti, hogy függetlenül attól, hogy egy adott adatpont melyik osztályhoz tartozik, az adott osztály előrejelzésének minden hibáját egyenlően kezeljük. Ez általában elérhető egy osztálysemleges delta függvényen keresztül, amely következetes büntetést alkalmaz minden kategóriában az osztályozás minden hibája esetén. Viszont az elméleti egyszerűség azonban nem mindig ideális, nem tükrözi a valós világ problémáit. Nem minden téves besorolásnak van egyforma súlya, különösen, ha nagy a tét, akkor számít a kontextus. Ebben az esetben a legjobb példa az orvostudomány, ezeknek a predikcióknak a helytelen eredményei életeket változtathatnak meg. A hamis negatív diagnózis, amikor egy káros állapotot nem észlelnek és a beteg további beavatkozás nélkül maradhat, vagy ennek éppen az ellentetje, amikor hamis pozitív eredményt észlelnek és ez gyakran még több stresszhez és vizsgálathoz vezethet. Ezek az esetek rámutatnak arra is, hogy a klasszifikációs eredmények minden aspektusát meg kell

figyelnünk. Gondolok ez alatt a valódi pozitív és negatív, valamint a hamis pozitív és hamis negatív értékekre.

## 2.3 A kép klasszifikáció alapjai

A képek klasszifikációja a számítógépes „látás” elsődleges és legalapvetőbb eleme, amely magába foglalja adott képek vagy képkockák besorolását több előre meghatározott osztály egyikékbe. Az algoritmusok vagy mélytanulási modellek felismerik a kép elsődleges tartalmát vagy tárgyakat, lehet ez egy macska, egy ember vagy esetleg egy gyümölcs. A képosztályozás a számítógépes látás egyik sarokköve, ez alapozza meg a későbbi sokkal összetettebb feladatokat, mint például az arcfelismerést vagy a képszegmentálást.
A klasszifikáció bonyolultságát a következő példán keresztül fogom szemléltetni. Tegyük fel, hogy a modell egy képet vesz bemenetként és megpróbálja egy címkéhez rendelni, a négy érték közül, ami legyen most kutya, autó, ember vagy asztal. A képen egy kutya látható, ami átméretezésre került, emiatt jelenleg 100 pixel széles és 100 pixel magas, valamint 3 színcsatornája van. (Szeliski, 2022)

*ábra 2 Klasszifikációs modell bemeneti képének kutya példánya (forrás: (*Dachshund Dog Stock Photo*, n.d.)*

```
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 253 253 254 255 255 255 254 252 154  40  39  21  15  14  18  24  22  12  12  14  19
 15  19  74  37  30  26  30  30  22  15  16  16   8  15  18  31  28  49  47 191 251 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255
```

*ábra 3 Kutyakép pixelértékeinek reprezentációja (saját forrás)*

Ezért a kép 100 x 100 x 3 pixelből, vagyis 30.000 értékből áll. A modell feladata, hogy ezt a 30.000 értéket egyetlen címkéve alakítsa, például kutya. Látható, hogyha innen közelítjük meg nem is olyan egyszerű, mint az emberi agy számára, viszont a számítógépes algoritmusnak

ezekkel is meg kell birkózniuk. A kihívás elsődleges oka a képek nyers feldolgozása és tárolása a fényerőértékek 3 dimenziós tömbjeként.

A következőkben bemutatom azokat a főbb kihívásokat a képfeldolgozás területén, amelyek a képek készítésének módjaiból adódnak.

Egyik főbb nehézség lehet a nézőpontok változatossága. Egy tárgynak a kamerához viszonyított pozíciója jelentősen megváltoztatja annak megjelenését a képen. Klasszikus példa erre a bögre, amely felülről nézve kör alakú, oldalról viszont henger formát ölt.

Továbbá a távolság is bonyolítja az algoritmusok feladatát. A körülöttünk lévő világban a tárgyak bármilyen méretűek lehetnek. Egy távoli repülő az égen és egy játékrepülő közelről fotózva nem ugyanazt a pixelteret foglalja el. Ez a példa azt is bizonyítja, hogy az algoritmusoknak jó általánosító képességgel kell rendelkeznie a képen megjelent mérettől függetlenül.

Emellett a valóságban a tárgyak nem mindig merevek, ami a deformáció kihívását jelenti. Gondoljunk csak egy egyszerű pólóra; a megjelenése nagymértékben változhat, ha laposan, összegyűrve vagy viselve van. Annak garantálása, hogy ezeket a különböző formákat ugyanannak a tárgykategóriának ismerjük fel, döntő fontosságúvá válik.

Az eltakarás egy következő akadályt jelent. Sok esetben az érdekes tárgyak részben más tágyak mögött rejtőzhetnek. Mondjuk, ha egy macska a függöny mögött ül, és csak a farka látszik ki, akkor a kihívást az jelenti, hogy felismerjük a tárgyat ebből a kevésbé látható részből is.

A megvilágítási viszonyok is döntő szerepet játszanak a tárgy megjelenésének megváltoztatásában. Az árnyékok, a fény intenzitása és iránya megváltoztathatja a tárgy pixelértékekben való megjelenítését. Teszem azt, egy helység teljesen másképp nézhet ki délben, mint a szürkület homályos fényében, ami nyomatékosíja, hogy az osztályozó algoritmusoknak rugalmasnak kell lenniük az ilyen megvilágítási eltérésekkel szemben.

A különböző hátterek tovább bonyolíthatják a helyzetet. Gyakran előfordulhat, hogy a tárgyak nem különülnek el a környezetüktől. Klasszikus példa erre egy zebra, amely zavartalanul beleolvad a hosszú fűbe és az egymásba tűnő árnyékokba. Amikor egy tárgy pixelei összefonódnak a háttér pixeleivel, a tárgy határainak megkülönböztetése komoly feladattá válik (Szeliski, 2022).

Végezetül nem hagyható figyelmen kívül az osztályon belüli eltérés fogalma sem. Egyetlen osztályon belül a variabilitás tartománya óriási lehet. A "kutya" osztály kiváló példa erre. A chihuahuáktól egészen a dán dogokig terjedő spektrum jól mutatja az egyetlen osztályon belüli hatalmas különbségeket, mind a méret, mind a megjelenés tekintetében. A képosztályozó

algoritmusok számára elengedhetetlen az ilyen széles variációk figyelembevétele és a hatékony kategorizálás biztosítása egy címke alatt.

## 2.4 NN és KNN

A k-legközelebbi szomszéd („k-Nearest Neighbours” rövidítve kNN) egy egyszerű, de hatékony felügyelt gépi tanuló algoritmus, amelyet osztályozási és egyaránt regressziós feladatokban is használnak. A kNN példányalapú osztályozó algoritmus, viszont mielőtt bemutatnám a k-legközelebbi szomszédot helyezzük egy szélesebb osztályozási keretbe, amely a folyamatokat induktív és deduktív érvelési fázisokra oszhatunk. Az induktív szakasz elsődleges célja egy osztályozási modell létrehozása. A rendelkezésre álló adatok felhasználásával a változók közötti mintákat és kapcsolatokat feltárják, valamint előrejelzési modellé alakítják. A következő a deduktív szakasz, amely a modellt a tesztesetekkel szemben használja fel.

A különböző gépi tanuló algoritmusok eltérően illeszkednek ebbe a keretbe. A mohó tanulók („eager learners”) mint például a döntési fák és a szabály alapú osztályozók, amelyek kifejezetten olyan modellek készítésére lettek tervezve, amelyek azonnal az osztálycímkékre alakítják a bemeneti attribútumokat amikor a tanuláshoz szükséges adatok rendelkezésre állnak. Az ilyen modellek proaktívak, általánosításokat készítenek a lekérdezések fogadása előtt. Ezzel teljes ellentétben a lusta tanulók („lazy learners”), akik azért születtek, hogy a modellezési folyamatot elhalasszák amíg nem kell osztályozni a teszteseteket. A „rote” osztályozó ezt a kategóriát szemlélteti, minden betanítási példányt a memóriába helyez, és csak akkor osztályoz, ha pontos egyezést talál (*CS231n Convolutional Neural Networks for Visual Recognition*, n.d.-a).

Visszatérve a k-legközelebbi szomszédra, ez az algoritmus jól egyesíti az előbb bemutatott különböző gépi tanuló algoritmusok legjobb tulajdonságait. Ahelyett, hogy a pontos egyezésekre támaszkodna, szélesíti a „látókörét” és olyan tanuló példányokat keres, amelyek attribútumok tekintetében nagyon hasonlítanak a tesztpéldányra. Ezek a legközelebbi szomszédok lesznek a kulcs a tesztpéldány osztály címkéinek megtalálásához. A következő mondás tökéletesen szemlélteti az algorimus logikáját, miszerint: „Ha valami úgy totyog, mint egy kacsa, kacsaként hápog és kacsának néz ki, akkor valószínűleg kacsa” (Ayto & Crofton, 2010).

A legközelebbi szomszéd osztályozó minden egyes esetet egy adatpontként reprezentál egy d -dimenziós térben, ahol d az attribútumok száma. Egy adott esetben a tanító adatok összes többi

pontjához való közelsége különféle távolságmérők segítségével számíthatók ki. A távolság kiszámítására az Euklideszi, a Manhattan és a Minkowski-távolságmérők használhatók, utóbbi leginkább a folyamatos adatoknál használatos. A Hamming-távolság kifejezetten alkalmas kategorikus adatokhoz, kiszámítja, hogy két azonos hosszúságú karakterláncban a megfelelő szimbólumok hány helyen különböznek egymástól. A számítástechnikában széles körben használják hibadetektálásra vagy hibajavításra, amikor az adatokat számítógépes hálózatokon keresztül továbbítják. A Koszinusz-távolság a vektorizált adatokhoz használható fel amikor egy vektortérben két nem null vektor közötti szög koszinuszának mérését akarjuk végrehajtani. Fontos megemlíteni, a távolságmérő megválasztása jelentősen befolyásolhatja a kNN algoritmus teljesítményét. Ezen túlmenően, a távolságok kiszámítása előtt döntő fontosságú annak biztosítása, hogy a jellemzők normalizálódjanak, mivel figyelemmel kell lennünk a mutatók skálaérzékenységére.

A módszertan meghatározása után nem lehet eléggé hangsúlyozni a k hiperparaméter fontosságát, amely a figyelembe veendő legközelebbi szomszédok számát jelöli. A k kis értékei érzékenyek az adathalmazban lévő zajra, és túlillesztéshez vezethetnek, míg a nagyobb értékek más osztályokból származó pontokat is tartalmazhatnak, ami potenciálisan téves osztályozáshoz vezethet. A kNN algoritmusnak a folyamat során minden egyes tesztesettet végig kell ismételnie, és a kiválasztott metrika segítségével ki kell számítania a távolságot a teszteset és a képzési adatok minden egyes példánya között. E távolságok rendezése után kiválasztjuk a k legközelebbi példányt, és e szomszédok közül a tesztesethez a többségi osztályt rendeljük. Bár a kNN elve egyszerű, a számítási bonyolultsága jelentős lehet, különösen nagy adathalmazok esetén. Minden egyes tesztpéldányhoz távolságszámításra van szükség minden egyes képzési példánnyal. A különböző optimalizálási technikák, például indexelés és hatékony adatszerkezetek azonban felgyorsíthatják a legközelebbi szomszédok keresését.

## 2.5 SVM

A támasztó vektor gép más néven „support vector machine” jelentős szerepet tölt be a hagyományos gépi tanulási algoritmusok között, amikor a gépi látásról beszélünk, különösen akkor, ha kis vagy közepes adathalmazokkal dolgozunk. Az SVM-ek lineáris és nemlineáris osztályozásra való képessége sokoldalúvá teszi őket. Alapvetően a „support vector machine” algoritmus egy olyan gépi tanulási modell, amelynek feladata az, hogy megtalálja azt az optimális hipersíkot, amely maximalizálja a margót a különböző osztályok között az adathalmazban. Ezt nevezzük nagy margós osztályozásnak, amely főként akkor használható,

ha az adathalmaz lineárisan szeparálható. A nem tökéletesen lineárisan szeparálható adatkészletek kezelése és a kiugró értékek csökkentése érdekében a lágy margós osztályozás lehetővé teszi a margó bizonyos mértékű megsértését, amelyet az SVM-modellekben a C regularizációs hiperparaméter szabályoz. (Chollet, 2018)

Matematikailag az SVM úgy írható le, ha tegyük fel, van egy adatkészletünk $x_i$ jellemzőkkel és $y_i$ címkékkel, ahol $y_i$ értéke 1 vagy -1, mely azt jelzi, hogy $x_i$ melyik osztályba tartozik. Az SVM célja, hogy megtalálja azt a döntési függvényt, amely kielégíti:

$$y_i(w^T x_i + b) \geq 1 \; \forall i.$$

A W vektor és a b skalárok amelyek meghatározzák a hipersíkot, valamint a feltétel biztosítja, hogy minden adatpont a margón kívül legyen. A margó szélessége $\frac{2}{||w||}$ emiatt az optimalizációs probléma egy minimalizálássá egyszerűsödik, mivel egyenértékű a $\frac{1}{2}||w||^2$ minimalizálásával (a fenti egyenlőtlenség által adott korlátok mellett). A lágy margós osztályozásban $\xi_i$, úgynevezett laza változót vezetünk be, hogy egyes pontok a margón belül vagy akár a hipersík másik oldalán is lehessenek. A feltétel így módosul:

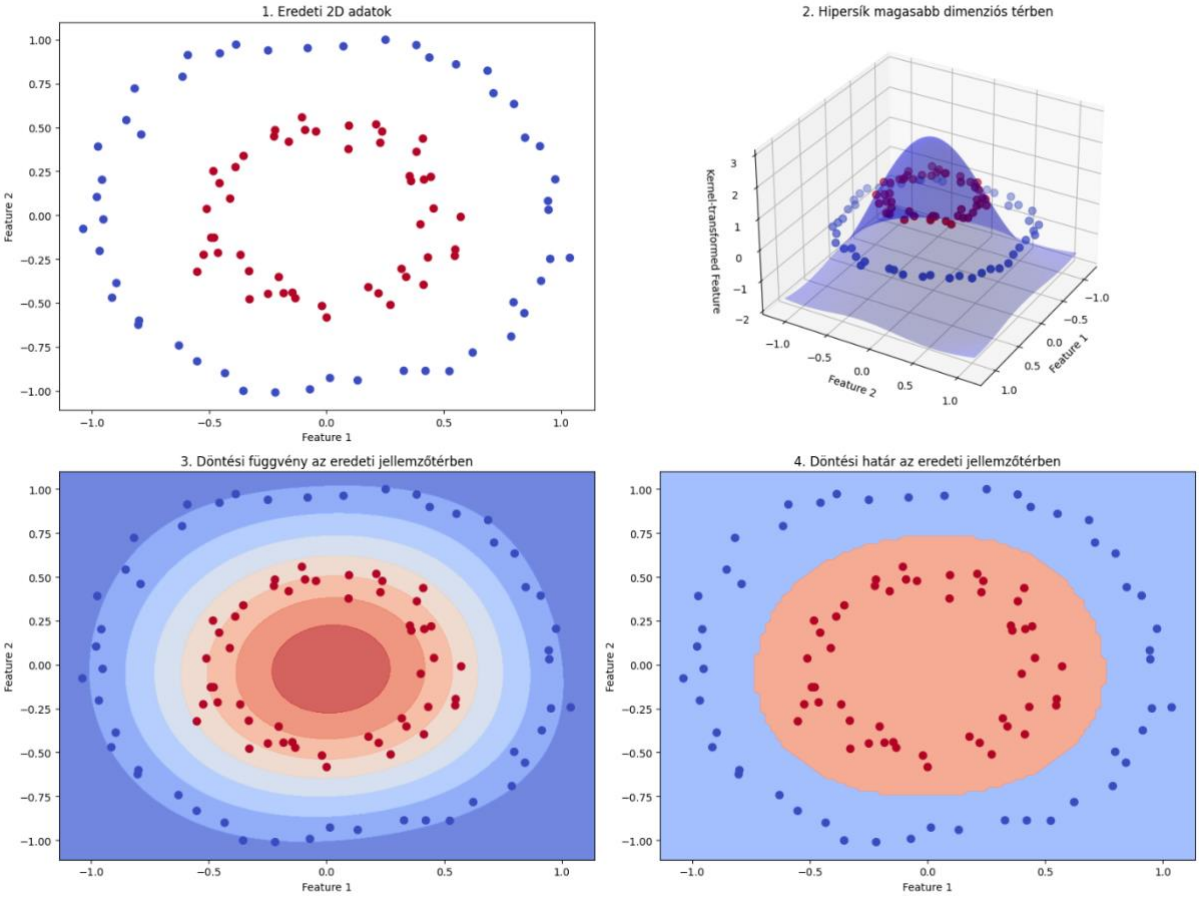
$$y_i(w^T x_i + b) \geq 1 - \xi_i \; és \; \xi_i \; \geq 0 \quad \forall i$$

A célfüggvényünk egy további tagot fog kapni, ami a $\frac{1}{2}||w||^2 + C\sum_{=i}^{m} \xi_i$. A C az úgynevezett regularizációs paraméter, amely a margó maximalizálása és az osztályozási hiba minimalizálása közötti kompromisszumot szabályozza. (Boyd & Vandenberghe, 2004)

Az SVM alkalmazása a képosztályozásban magában foglalja a jellemzők kinyerését, majd az SVM algoritmus használatát a képek osztályozására. A jellemzőkivonás tartalmazhat színhisztogramokat, textúramintázatot vagy élek felismerését, amelyek kulcsfontosságúak a képtartalom reprezentálásában. Számos adatkészlet nem lineárisan szeparálható, ennek megoldásaként az SVM-ek a kernel trükköt alkalmazzák a nagydimenziós jellemzőterek hatékony kezelésére.

Ez azt jelenti, hogy az eredeti adatokat egy magasabb dimenziós térbe vetíti, ahol az osztályozási probléma lineárisan szeparálhatóvá válhat. Ez a transzformáció nem kerül explicit módon kiszámításra, ehelyett egy kernel függvény kiszámítja a jellemzőtérben lévő összes adatpár képe közti skalár szorzatot. A rendelkezésre álló különböző kernelek közül a „Radial Basis Function" (RBF) kernelt széles körben használják, mivel sokoldalúan képes megragadni az összetett struktúrákat. Ez a folyamat a következő ábrán keresztül könnyen bemutatható. Először az eredeti 2D-s adathalmaz jelenik meg, megfigyelhető, hogy ez nem lineárisan szeparálható. Ezt követően az RBF kernelen keresztül történő transzformáció kerül

ábrázolásra, amely a hipersíkot egy magasabb dimenziós térben ábrázolja. Ezt követi a döntési függvény ábrázolása, amely megmutatja, hogy az SVM hogyan rendel értékeket az átalakított tér különböző tartományaihoz. Végül a döntési határt visszahelyezzük az eredeti jellemzőtérbe, szemléltetve az SVM által az osztályok közötti egyértelmű felosztást. A kernel-trükk eleganciája abban rejlik, hogy képes kezelni az adatok komplexitását, miközben megőrzi a számítási hatékonyságot, ami a mintafelismerés területén hatékony eszközzé teszi (Géron, 2019).




*ábra 4 SVM döntési határok és kernel trükk vizuális szemléltetése (saját forrás)*

## 3. Mélytanulás és neurális hálozatok

### 3.1 Deep learning és CNN

A mai modern technológiai korszakban, a mély tanulás („deep learning”) mindenütt felbukkan a mindennapjainkban, viszont ez azt is jelenti, hogy alapjaiban változtatja is meg azt. Rengeteg ágazatot alakít át gyökeresen. A mély tanulás lényegében a gépi tanulás egy részhalmaza, amely maga is a mesterséges intelligencia („AI”) egyik ága. A hagyományos algoritmusokkal ellentétben, amelyek kifejezetten programozott utasításokat követnek, a mélytanulási

algoritmusok az adatokból tanulnak. Komplex mintákat elemeznek és modelleznek több rétegű neurális hálózattok segítségével, innen ered a mélytanulás kifejezés. (I. Goodfellow et al., 2016; Louridas, 2020)

Míg a „deep learning" alapkoncepciója az 1940-es és 1950-es évekig nyúlik vissza, a neurális hálózattok megjelenésével, addig a népszerűsége csak az elmúlt évtizedekben nőtt meg (I. Goodfellow et al., 2016). Számos dolog segítette, hogy a mély tanulás a mai technológia élvonalába kerüljön.

Először is, a digitális korban rendelkezésre álló hatalmas mennyiségű adat biztosítja az alapvető nyersanyagot az összetett neurális hálózattok tanításához. A social média bejegyzésektől, a NASA nagyfelbontású infravörös „képeiig", az adatok mennyisége és sokfélesége exponenciálisan nő (Szeliski, 2022). Az algoritmusok ezt az adatbőséget kihasználva egyre gazdagabb reprezentációkat tanulnak meg, ahogy a képzési adatok mennyisége növekszik.

Másodszor, a számítási teljesítmény, különösen a GPU-k fejlődése kulcsszerepet kap a „deep learning" folyamatos fejlesztésében. A modellek betanítása hatalmas számítási erőforrást igényel, a GPU-k amik eredetileg a grafikai rendelerésre lettek kifejlesztve, kivételesen alkalmasok a mély tanulásban szükséges párhuzamos számításokhoz, ezáltal lecsökkentve a betanítási időt. (Géron, 2019)

És végezetül, a nyílt forráskodú kezdeményezések lehetővé tették a technológiához és tudáshoz való szabad hozzáférést. Az olyan platformok, mint a TensorFlow (Abadi et al., 2016) és PyTorch (Paszke et al., 2019), valamint az online tanfolyamok elterjedése lehetővé tette a rajongók és kutatók számára, hogy elmélyedjenek ezen a területen, valamint egy olyan közösséget hozott létre, amely gyorsan iterálta és javította a meglévő technikákat.

Több különbség is van a hagyományos képfeldolgozási és mélytanulási megoldások között. A képfeldolgozás lényege a kép manipulációja a kívánt eredmények elérése érdekében, mint például a jellemzők megerősítése vagy az értelmes információk kinyerése. A hagyományos képfeldolgozási technikák, mint például az élek felismerése, a szűrés és a hisztogram kiegyenlítés, manuálisan készített jellemzőkön alapulnak. (Szeliski, 2022) A mérnökök és szakértők aprólékos munkával tervezik és hangolják ezeket a jellemzőket, hogy azok az adott feladatokhoz megfelelőek legyenek. Ezek a módszerek azonban gyakran megbicsaklanak, amikor a valós világ képeinek változatosságával, például változó fényviszonyokkal, elfedésekkel vagy változatos háttérrel szembesülnek. A mélytanulás ahelyett, hogy a manuálisan tervezett jellemzőkkel dolgozna, automatikusan hierarchikus reprezentációkat tanulnak meg az adatokból. A képfeldolgozás szemszögéből megközelítve, ez azt jelenti, hogy

a konvolúciós hálozat kezdetben megtanul egyszerű mintákat, például éleket vagy textúrákat és fokozatosan felismeri az összetetteb struktúrákat, például alakzatokat vagy objektumokat. Az adatokból való adaptív tanulási képesége miatt a neurális hálózattok képesek felülmúlni a hagyományos technikákat, különösen olyan esetekben mint a képfelismerés, objektum felismerés és szegmentáció. (I. Goodfellow et al., 2016)

## 3.2 Mélytanulás és neurális hálózattok bemutatása:

A hálózatok alapvető szintjén a neuronok helyezkednek el. A biológiai neuronokhoz hasonlóan ezek a mesterséges változatok több bemenetet fogadnak, feldolgozzák azokat, és egy vagy több kimenetet állítanak elő. Minden neuron egy sor súlyt és egy torzítási tényezőt használ. A súlyok egy adott bemenet erősségének vagy ráhatásának mértékének tekinthetők, míg a torzító tényező („bias") lehetővé teszi a neuron kimenetének rugalmas beállítását. Matematikailag egy neuron működése a bemenetek súlyozott összegeként írható le, amely aztán egy aktiváló függvényen keresztülhalad. (Géron, 2019)

Formálisan, adott $x_1, x_2, \dots x_n$ bemenetek és $w_1, w_2, \dots w_n$súlyok esetén az $y$ kimenet az aktiválás előtt a következő:

$$y = \sum_{i=1}^{n} w_i x_i + b$$

Itt a b a torzítást (bias) jelöli, ami a neuron kimenetét a bemenetektől függetlenül eltolja. Az aktiválás utáni végső kimenet f(y), ahol f az aktiválási függvény. (Louridas, 2020)

### 3.2.1 Aktiválási függvények

*Aktiválási függvények alapjai*

Az aktiválási függvény megválasztása kulcsfontosságú a nemlinearitások kezelésében. Az aktiválási függvények nemlinearitást vezetnek be a hálózatba, lehetővé téve, hogy az adatokban lévő összetett összefüggéseket megtanulja. Ha nem használnánk ezeket a függvényeket akkor bármilyen mély is lenne a hálózatunk ugyanúgy viselkedne, mint egy egyrétegű lineáris modell. Fontos, hogy megfelelően válaszuk ki az aktiválási függvényeinket. Több problémát is fel kell sorolni, amikor a megfelelőt keressük. (Géron, 2019)

A neurális hálózatok hatékony eszközként jelennek meg különböző alkalmazásokban, beleértve a képosztályozást is. E hálózatok teljesítménye, különösen az olyan feladatokban,

mint a minták vagy jellemzők felismerése a képeken, a megfelelő képzésükön múlik. A mély neurális hálózatok képzése azonban nem mindig egyszerű.

**Aktivitási függvénynél felmerülő problémák:**

A gradiens alapú tanulási algoritmusokban a gradiensek játszanak szerepet a neurális hálózat súlyainak frissítésében. Ugyanakkor felmerül egy jelentős kihívás. Ahogy a gradiensek visszafelé terjednek a rétegeken át, értékeik vagy zsugorodhatnak, ami az "eltűnő gradiens problémához" vezet, vagy kontrollálatlanul nőnek, ami a "robbanó gradiens problémát" okozza. (I. Goodfellow et al., 2016)

- Eltűnő gradiens probléma: Bizonyos aktiválási függvények, mint például a szigmoid, használatakor a gradiensek túl kicsivé válhatnak. Ennek eredményeképpen a súlyfrissítések elhanyagolhatóvá válnak, ami a hálózatot gyakorlatilag taníthatatlanná teszi. Ez a probléma különösen hangsúlyos a „deep” (mély) hálózatoknál. Ha például egy több rétegből álló mély neurális hálózatot tekintünk, a gradiensek láncolódása miatt a gradiensek közel nulla értékre csökkenhetnek. Ez azt jelenti, hogy a hálózat súlyai szinte változatlanok maradnak, és a tanulás megáll. A probléma lényege az aktiválási függvény deriváltjában rejlik. A szigmoid aktiválás esetén a maximális derivált 0,25. A mély hálózatokban ez egy olyan gradienst eredményez, amely a mélységgel exponenciálisan csökken, és következetlenül kicsi lesz. (Géron, 2019)
- Robbanásszerű gradiens probléma: Másrészről, amennyiben a gradiens értékek túlzottan magasak, akkor hatásuk láncreakcióként jelentkezhet, ami kezelhetetlenül nagy súlyfrissítésekhez és instabil hálózathoz vezet. Figyelembe véve a jelenlegi előrehaladásunkat a témában, vizsgáljuk meg azt az esetet, ahol a hálózat súlyai - egyszerűsítés céljából - valamennyien nagyobbak mint 1. Ez a helyzet a kimeneti értékek exponenciális növekedéséhez vezethet, ami rendkívül magas értékeket eredményezhet. Amikor ez a folyamat visszafelé halad, az eredményül kapott nagy súlyfrissítések instabilitást okozhatnak a modellben, ami nehezíti annak megbízható tanítását és előrejelzési képességét. Ezeknek a problémáknak a felismerése éberséget igényel. A jelek közé tartozik a gyenge teljesítmény a képzési adatokon, az instabil tanulás a veszteség drasztikus ingadozásával, vagy akár a veszteség nulla értékűvé válása. (Géron, 2019)
- A következő probléma, ami felmerül az az adaptív tanulás. Egyes aktiválási függvények, mint például a Parametrikus ReLU (PReLU), betanítható paramétereket vezetnek be. A képosztályozásban ez azt jelenti, hogy a függvény a képek vagy

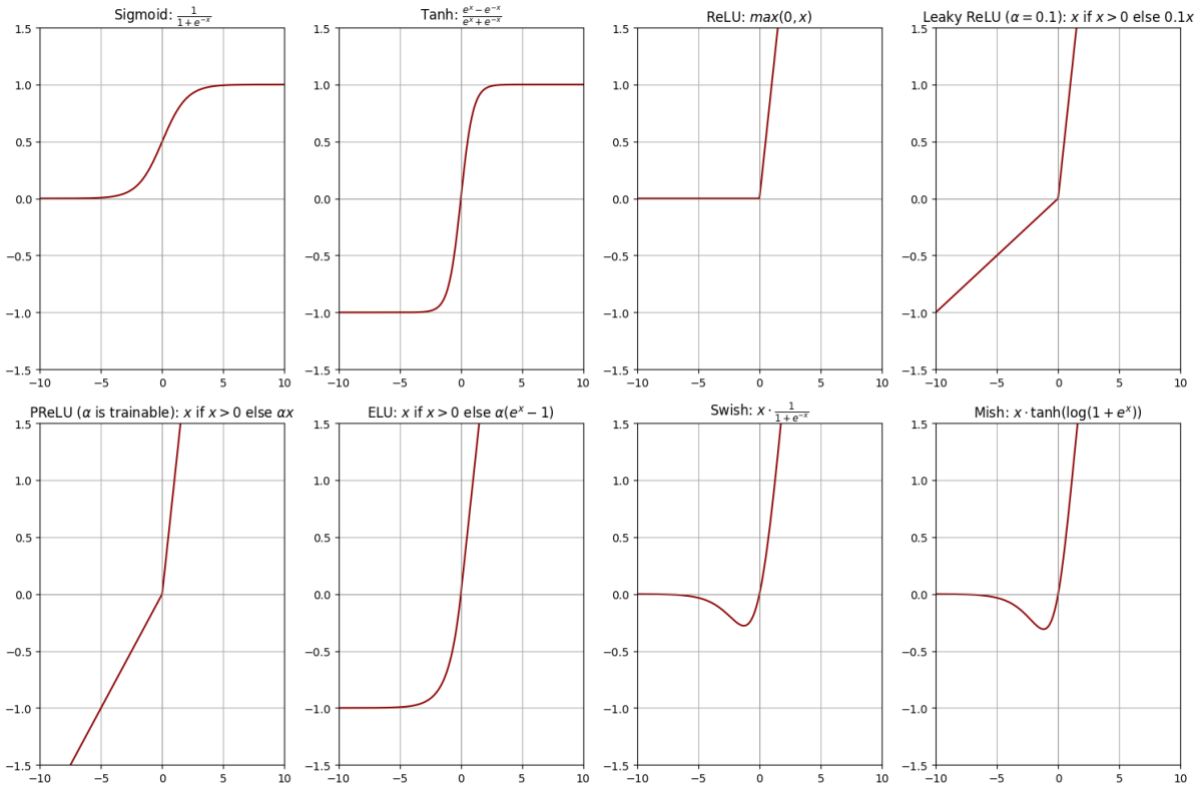
jellemzők típusa alapján alkalmazkodik, ami potenciálisan jobb jellemzőreprezentációt és jobb osztályozást eredményezhet. (He et al., 2015)

- A legutolsó felmerülő probléma az úgynevezett halott neuron esete. A hagyományos ReLU aktiválási függvény olyan neuronokhoz vezethet, amelyek soha nem aktiválódnak, lényegében haszontalanná válnak, ezt a jelenséget "haldokló ReLU"-nak nevezik. Ez problémás lehet a képosztályozásban, ahol minden neuron hozzájárulása létfontosságú lehet az egyes jellemzők felismerésében.

A korábban említett problémákra válaszként az alábbi megoldásokat alkalmazhatjuk:

- Aktiválási függvények megválasztása: A szigmoidról más aktiválási függvényekre, például „ReLU”-ra („Rectified Linear Units”) való váltás enyhítheti az eltűnő gradiens problémáját. A „ReLU” minden pozitív értékre lineáris, negatív értékekre pedig nulla, így kevésbé érzékeny a eltűnő gradiensre. Másfelől ez magával hozza sajátos kihívásokat is, mint amilyen a 'dying ReLU' jelenség, ami akkor fordul elő, amikor a ReLU aktivációs függvény kimenetei huzamosabb ideig nulla értéket vesznek fel, ezzel csökkentve a hálózat tanulási képességét. Ezt a problémát alternatív megközelítések, mint például a 'leaky ReLU' (szivárgó ReLU) alkalmazásával lehet mérsékelni, amely biztosít egy kis gradiens áteresztést a negatív tartományban is.
- Batch normalizálás: Az aktiválások normalizálása egy rétegen belül stabilizálhatja a tanulási folyamatot és enyhítheti a robbanó gradiens problémáját. Ezt a technikát még részletesen bemutatom a szakdolgozatomban. (Ioffe & Szegedy, 2015)
- A súlyok inicializációja: A súlyok megfelelő inicializálása, a megfelelő skála biztosítása megakadályozhatja, hogy a gradiensek a képzés kezdetekor túl kicsik vagy túl nagyok legyenek.
- Gradiens-korlátozás: Ez egy küszöbérték beállítását és a küszöbértéket meghaladó gradiensek skálázását jelenti. Ez egy közvetlen módszer annak biztosítására, hogy a gradiensek ne robbanjanak fel.

(He et al., 2015b)




*ábra 5 Neurális hálozati aktivációs függvények vizuális összehasonlítása (saját forrás)*

*RELU (Rectified Linear Unit)*

A korrigált lineáris egység (ReLU) a mélytanulási modellek egyik alapvető aktivációs függvényévé vált. Az $f(x) = max(0, x)$ egyenlet által definiált ReLU nemlinearitást vezet be a modellekbe anélkül, hogy számításigényes műveletekre támaszkodna. Az egyszerűségével tűnik ki a többi aktivációs függvény közül. A függvényben bármilyen negatív bemenet nullává alakul át, míg a pozitív értékek változatlanok maradnak. Ez a logika a függvény egyszerűsége ellenére figyelemre méltó képességeket kínál. Például a többrétegű hálózatokban a ReLU alkalmazásával a modell bonyolult hierarchikus mintákat tanulhat, ami olyan feladatoknál, mint a képfelismerés, előnyös tulajdonság. (Géron, 2019; I. Goodfellow et al., 2016)

Előnyök:

- A ReLU felgyorsítja a sztochasztikus gradiens süllyedés konvergenciáját az olyan hagyományos függvényekhez képest, mint a szigmoid vagy a tanh. Ez a gyorsulás a lineáris, nem szaturáló alkatának tulajdonítható. (Krizhevsky et al., 2017)
- A ReLU megvalósításának egyszerűsége, amely az aktiválások nullánál történő küszöbérték-aktiválását foglalja magában, számítási hatékonyságot kínál a

tanh/szigmoid függvényekkel szemben, amelyek számításigényesebb műveletekre támaszkodnak.

Hátrányok:

- A képzés során a ReLU-egységek „törékenyek” lehetnek és „meghalhatnak”. A neuronon átáramló nagy gradiensek úgy frissíthetik a súlyokat, hogy a neuron soha nem aktiválódnak egyetlen adatpontra sem, így a gradiens nulla lesz. Ez a „haldokló ReLU” néven ismert jelenség a hálózat jelentős részeinek inaktívvá válásához vezethet, különösen akkor, ha a tanulási sebesség nem optimálisan van beállítva.

*Szigmoid:*

Az $f(x) = \frac{1}{1+e^{-x}}$ által leírt szigmoid függvényre áttérve egy olyan függvénnyel találkozunk, amely elegánsan vált át 0-ról 1-re, így ideális a bináris osztályozási feladatokhoz. S-alakú görbéje vagy logisztikus görbéje egyedülálló képességet ad neki, hogy bármilyen bemenetet, legyen az bármilyen pozitív vagy negatív, képes 0 és 1 közötti értékre összenyomni. (Géron, 2019) Például egy neurális hálózatban, amelynek célja egy betegség diagnosztizálása a különböző tünetek alapján, akkor a Szigmoid függvény meg tudja becsülni a betegség jelenlétének valószínűségét.

*Hiberbolikus tangens:*

A Tanh-függvény vagy hiperbolikus érintő egy másik kulcsfontosságú aktiválási függvény, amelyet az $f(x) = \frac{2}{1+e^{-2x}} - 1$ függvény ír le. A Szigmoidtól eltérően a Tanh mínusz egy és egy közötti értékeket ad ki, nullás kimenetet kínálva. Ez azt jelenti, hogy a negatív bemenetek erősen negatív, a nulla bemenetek pedig nulla közelében lesznek a kimeneten, ami néha segíthet a hatékonyabb képzésben. Tegyük fel, hogy egy olyan modellt építünk, amely a szöveges bemenet alapján előrejelzi a pozitív vagy negatív hangulatot. A Tanh a tartományával a negatív érzelmeket jobban leképezheti, mint a Szigmoid függvény. (Géron, 2019)

*Leaky ReLU:*

A "dying ReLU" probléma megoldására bevezették a Leaky ReLU függvényt. Ezek kis pozitív meredekséggel rendelkeznek a negatív tartományban, jellemzően 0,01 körül. Matematikailag úgy ábrázoljuk, hogy $f(x) = 1(\alpha x < 0) + 1(x \geq 0)(x)$, ahol $\alpha$ egy kis konstans. Bár egyesek sikerrel jártak a Leaky ReLU-val, az eredmények feladatonként eltérőek. Egy adaptívabb

változat, a PReLU lehetővé teszi, hogy a negatív tartományban a meredekség tanulható paraméter legyen, ami nagyobb rugalmasságot kínál. (Géron, 2019)

*Maxout:*
A hagyományos aktiválási függvényeken túl, (I. J. Goodfellow et al., 2013) által bevezetett Maxout neuron egy általánosabb megközelítést kínál. A $\max(w_1^T x + b_1, w_2^T x + b_2)$, függvényt számítja ki, amely speciális esetként magában foglalja a ReLU-t és a Leaky ReLU-t is. A maxout neuronok egyesítik a ReLU előnyeit (lineáris működési rend, telítődés nélkül) azonban annak hátrányai nélkül. Viszont a függvényben növekszik a paraméterek száma, ami potenciálisan bonyolítja a modellt. (I. J. Goodfellow et al., 2013)

### 3.2.2 A neurális hálózatok architektúrája

A neurális hálózatok hierarchikus rétegrendet követnek, ami azt jelenti, hogy a hálózattokat az összekapcsolt neuronok aciklikus gráfjaként lehet modellezni. Ez az aciklikus struktúra biztosítja az információ szisztematikus áramlását, ahol bizonyos neuronok kimenetei bemenetként szolgálnak mások számára, kiküszöbölve a végtelen ciklus lehetőségét. (I. Goodfellow et al., 2016) Az egyértelmű és hatékony feldolgozási paradigma kialakítása érdekében ezeket a hálózattokat általában különálló rétegekbe szervezik. A hagyományos neurális hálózatokban található rétegek sokfélesége közül kiemelkedik a teljesen összekapcsolt réteg. Ebben a kialakításban a szomszédos rétegek teljes kapcsolatot mutatnak, biztosítva, hogy az egyik réteg minden neuronja kapcsolatban áll a következő réteg minden neuronjával. Az egyes rétegeken belül azonban a neuronok elszigeteltek és összeköttetés nélküliek maradnak. (Szeliski, 2022)
A neurális hálózati architektúrák tárgyalásakor a terminológiai egyértelműség kiemelkedően fontos. Például egy N-rétegű neurális hálózat nem tartalmazza a bemeneti réteget. Következésképpen az egyrétegű neurális hálózat olyan struktúrát jelent, amely nem tartalmaz rejtett rétegeket, ami a bemenetről a kimenetre történő közvetlen leképezést jelenti. Az ilyen konfigurációk néha az egyrétegű neurális hálózatok kezdetleges formáinak tekinthetők. Ezeket a struktúrákat "mesterséges neurális hálózatoknak" (ANN) vagy "többrétegű perceptronoknak" (MLP) is nevezhetjük. (Géron, 2019)
Azonban a neurális hálozati struktúrák és a biológiai neurális hálózatok közötti bonyolultságra és különbségekre tekintettel, a szakterületen sokan inkább az "egységek" („unit”) kifejezést használják a "neuronok" helyett.

A neurális hálózatok kimeneti rétegében lévő neuronok gyakran nem rendelkeznek specifikus aktiválási függvénnyel, vagy úgy fogalmazhatók meg, hogy egyenesen lineáris identitású aktiválási függvénnyel rendelkeznek. Ez a tervezési módszer tükrözi a kimeneti réteg szerepét, amely jellemzően osztályozási feladatokban osztálypontszámokat, vagy regressziós esetekben valós értékű célértékeket képvisel.

A neurális hálózatok méretének és komplexitásának értékelése a hálózattervezés és -elemzés egyik központi szempontja. A leggyakrabban alkalmazott mérőszámok közé tartozik a neuronok teljes száma, vagy döntőtöbbségben a paraméterek teljes száma. (Szeliski, 2022) A jelenlegi alkalmazásokra, különösen a konvolúciós hálózatokra jellemző, hogy nem ritka a több millió paramétert tartalmazó, több rétegre kiterjedő hálózatokkal való találkozás, amelyek ezáltal a mélytanulás mélységét mutatják be. (Szeliski, 2022)

## 3.3 CNN

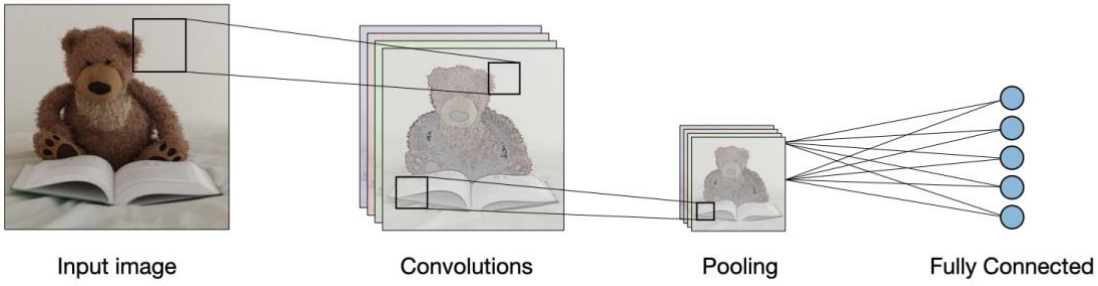
A konvolúciós neurális hálózattok (Convolutional Neural Networks, CNN, ConvNets) a neurális hálózattok egy speciális fajtája, amelyek rácsszerű topológiával rendelkeznek, mint például a képek. Ezek a hálózattok a konvolució matematikai műveletet, a lineáris műveletek egy speciális fajtáját használják, így rendkívül hatékonyak a képek pixeladatainak feldolgozásában és a jellemzők hierarchiájának megtanulásában. A CNN-ek alapvetően megegyeznek a hagyományos neurális hálózattok legalapvetőbb elemeivel, beleértve a tanulható súlyokkal és torzítással rendelkező neurális hálózattokat. A legtöbb esetben a neuronok úgy számítják ki a kimeneti értéket, hogy a súlyaik és a bemenet között skaláris szorzatot alkalmazzák, amelyet egy aktiválási függvény követ. (*CS231n Convolutional Neural Networks for Visual Recognition*, n.d.-b; Géron, 2019)

A CNN-ek azonban olyan speciális megoldásokat vezetnek be, mint a lokális receptív mezők, a megoszló súlyok és a térbeli hierarchiák, amelyek kihasználják a képekben jelenlévő térbeli-lokális korrelációt. Ezek az újítások lehetővé teszik a CNN-ek számára, hogy hatékonyan megragadják a képek strukturális és kompozíciós mintáit. A hagyományos neurális hálózatokkal, azaz a teljesen összekapcsolt hálózatokkal ellentétben a CNN-ek a bemenetként képeket feltételeznek. Ez a feltételezés lehetővé teszi, hogy hatékonyabban lehessen tervezni őket, és a hálózati architektúrába kódolni a sajátos tulajdonságokat. A hagyományos neurális hálózatok nem skálázódnak jól a képeken, mivel felépítésük miatt minden neuront összekötnek a következő rétegben lévő összes többi neuronnal. Ez hatalmas számú paraméterhez vezet, ami a hálózatot számításigényessé és túlillesztésre hajlamossá teszi. A CNN-ek a térbeli hierarchiák

és megosztott súlyok kódolásával jelentősen csökkentik a paraméterek számát, csökkentve a számítási igényeket és a túlillesztés kockázatát. (Chollet, 2018)

A CNN-ek létfontosságú részét képezik a mélytanulás egyre bővülő univerzumának, különösen a képfelismerés területén és számítógépes látás. Míg a hagyományos neurális hálózatok a bemeneti adatokat egyszerű vektorként kezelik, a CNN-ek megőrzik az adatok térbeli struktúráját, beleértve a képek szélességét, magasságát és mélységét (színcsatornáit). Ez a térbeli hierarchia megőrzése, amely döntő fontosságú a képekkel kapcsolatos feladatokban elért sikereikben. A „feedforward„ neurális hálózatoktól (többrétegű perceptronoktól) eltérően, amelyek a mélytanulás alapját képezik, viszont ezek nem rendelkeznek térbeli érzékenységgel, a CNN-ek eredendően rendelkeznek ezzel a tulajdonsággal, ami alkalmassá teszi őket a képi adatok kezelésére. A rekurrens neurális hálózatok (RNN), amelyek a szekvenciális adatfeldolgozásban, például idősor-előrejelzésben vagy természetes nyelvi feldolgozásban való jártasságukról ismertek, nem használják ki a képek térbeli függőségeit olyan hatékonyan, mint a CNN-ek. (Géron, 2019)

A CNN-ek egyik sarokköve a hierarhikus tanulás képessége. A képek területén ez azt jelenti, hogy kezdetben alapvető mintákat, például éleket vagy színeket ismernek fel, majd ezekre az alapokra építve összetettebb struktúrákat, például alakokat vagy egész tárgyakat képesek felismerni. A hálózat korábbi rétegei egyszerű jellemzők által aktivált szűrőket használnak, viszont az egyre mélyebb rétegekben már a textúrák és minták aktiválják és végül az utolsó rétegekben már lehetséges, hogy összetett jelenetek vagy tárgyak aktiválják. A CNN-ek ugyanazt a szűrőt vagy kernelt alkalmazzák a kép különböző részein. Ez a súlymegosztás azt jelenti, hogy ha egy CNN egyszer megtanul egy jellemzőt (például egy orr görbületét vagy egy ajtó szélét), akkor azt a kép bármely pontján képes felismerni. Ez a transzlációs invariancia néven ismert tulajdonság felbecsülhetetlen értékű a képelemzésben, mivel a tárgyak számtalan módon jelenhetnek meg a fényképeken. A CNN-ek emellett olyan rétegek segítségével csökkentik a képadatok dimenzióját, mint a "pooling" és a "subsampling", és a kép különböző részeinek megkülönböztetését segítő alapvető jellemzőkre összpontosítanak. Ez a dimenziócsökkentési folyamat kritikus fontosságú a hatékony képelemzéshez.

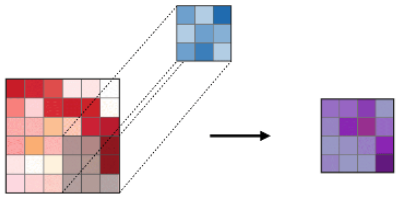



*ábra 6 Konvolúciós neurális hálózat folyamatának szemléltetése (forrás: (Amidi, n.d.) )*

### 3.3.1 Konvolúciós réteg:

A konvolúciós rétegek a CNN-ek alapvető építőkövei, az első réteg a hálózatban mindig egy konvolúciós réteg. Ez a réteg egy konvolúciós műveletet alkalmaz a bemenetre majd ezt továbbítja a következő rétegnek. A konvolúció a receptív mezőben lévő összes képpontot egyetlen értékké alakítja át. Ha egy képre használjuk a műveletet akkor egyrészt csökken a kép mérete, másrészt a mezőben lévő összes információt egyetlen pixelbe foglalja össze. Leggyakrabban használt konvolúció típus a 2D konvolúciós réteg, amelyet általában conv2D rövidítéssel szoktunk használni. A conv2D rétegben egy szűrő („filter") vagy kernel csúszik a 2D-s bemeneti adatokon. (Géron, 2019; I. Goodfellow et al., 2016)

A jellemződetektor egy kétdimenziós (2D) súlyok mátrixa, amely a kép egy részét reprezentálja. Bár ezek mérete változhat, a szűrő mérete jellemzően egy 3x3-as mátrix. Ez határozza meg a receptív mező méretét is. A szűrőt ezután a kép egy területére alkalmazzuk, és a bemeneti pixelek és a szűrő között skaláris szorzatot számolunk. Ez az eredmény ezután egy kimeneti mátrixba (*What Are Convolutional Neural Networks?*, n.d.) majd a szűrő egy lépéssel eltolódik, és a folyamat addig ismétlődik, amíg a kernel át nem fut a teljes képen (*What Are Convolutional Neural Networks?*, n.d.). A bemenet és a szűrő skaláris szorzatának eredményeinek sorozatából származó végső kimenetet „feature map"-nek, aktivációs térképnek hívjuk. (*What Are Convolutional Neural Networks?*, n.d.)



*ábra 7 Konvolúciós művelet szemléltetése (forrás: (Amidi, n.d.) )*

Fontos, hogy a jellemződetektor súlyai a képen való mozgás során rögzítettek maradnak, amit paramétermegosztásnak is nevezünk. Néhány paraméter, például a súlyértékek, a képzés során, a „backpropagation” és a gradiens ereszkedés folyamatán keresztül módosulnak. Van azonban három olyan hiperparaméter, amelyek a kimenet méretét befolyásolják. (*What Are Convolutional Neural Networks?*, n.d.)

Az első befolyásoló tényező a filterek száma, amely meghatározza a kimenet mélységét. A második a lépésszám, amely a pixelek számát jelenti, amennyivel a kernel a bemeneti mátrixon áthalad. A nagyobb lépésszám kisebb kimenetet eredményez. Az utolsó tényező, a nulla kitöltés, más néven „zero-padding”, akkor hasznos, ha a szűrök nem illeszkednek a bemeneti képhez. A nulla kitöltés a bemeneti mátrixon kívüli elemeket nullára állítja, emiatt ez a módszer befolyásolja a kimeneti méretet.

A konvolúciós hálózattok egyik legfontosabb jellemzője, hogy taníthatóak több-rétegű konvolución keresztül. Ez a (Lecun et al., 1998a)) és több más szakértő által népszerűsített koncepció forradalmasította a képi feldolgozásának módszerét.

Egy konvolúciós rétegben a művelet nem csupán egy szűrő egyszerű alkalmazása a képen. Több bemeneti és kimeneti csatorna közötti összetett kölcsönhatást foglal magában. A konvolúciós neurális hálózatokban (CNN) minden konvolúciós kernel úgy működik, hogy bemenetként veszi az előző réteg összes csatornáját, amelyeket egy kis területre, azaz egy ablakra szűkít. Ezután a kernel értékeket állít elő a következő réteg egyik kimeneti csatornájához. Ez az eljárás megismétlődik az összes kimeneti csatornán, és minden csatornának saját kernel súlykészlete lesz. Ez a több csatornás megközelítés lehetővé teszi a hálózat számára, hogy helyi jellemzőket építsen fel, és ezeket kombinálva diszkriminatívabb és szemantikailag tartalmasabb tulajdonságokat állítson elő.

Annak érdekében, hogy részletesebben megérthessük a neurális hálózatokban a konvolúciós réteg viselkedését meghatározó paramétereket, térjünk ki az egyes elemekre:

*Kitöltés („padding”):*

A rétegben végrehajtott kitöltési eljárások kétféle formát ölthetnek. Az elsőként említendő a zéró kitöltés, amely során, mint azt korábban már jeleztem, nullák hozzáadásával bővítjük ki a bemeneti adatok peremterületeit Ez a kimeneti jellemzőtérkép méretét a bemeneti mérethez közel tartja, megakadályozva a konvolúció utáni csökkentést. A másik a pixel replikáció, ilyenkor a nullák helyett a szélső pixelek replikálódnak a kitöltésben.

Ha van egy 5x5-ös bemeneti mezőnk és egy 3x3-as szűrőnk, a kitöltés nélkül a kimeneti jellemzőtérkép 3x3-as lenne. De nulla kitöltéssel (1 réteg nullával a bemenet körül) a kimenet

5x5 marad (*CS231n Convolutional Neural Networks for Visual Recognition*, n.d.-b). Ezt matematikailag a következő képpen tudnánk leírni:
Kitöltés nélkül, ahol a kimenet mérte O és a bemeneti mértet N, valamint a filter mérete F:

$$O = N - F + 1$$

P számú kitöltéssel:

$$O = N - F \ + 2P + 1$$

*Lépésszélesség („stride”):*
A lépésszám a bemeneti mátrixon történő pixeleltolódások számát jelöli. A nagyobb lépésszélesség (pl.:3) azt jelenti, hogy a szűrő egyszerre 3 pixelt mozgat. Ez csökkenti a kimeneti jellemzőtérkép méretét, és kihagyhat apróbb részleteket. Ha a lépésszélesség egy akkor egyszerre egy képpontot mozgat a szűrő, ami részletes jellemzőtérképet eredményez.
Matematikailag leírva, ha S a lépésszélesség:

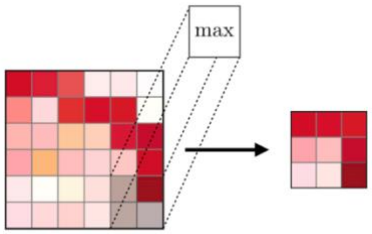
$$O = \frac{N - F + 2P}{S} + 1$$

*Dilatációs konvolúció:*
A (Yu & Koltun, 2016) által kifejlesztett újítás egy további híperparaméter bevezetését foglalja magába a konvolúciós (CONV) rétegek területén, melyet dilatációnak titulálnak. Ennek lényege, hogy bizonyos esetekben a szűrőelemek közötti távolság növekedhet, ami térközök kialakulását eredményezi a cellák között; ezt a jelenséget dilatációnak nevezik.

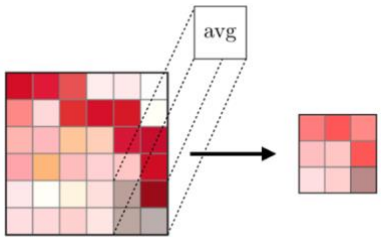
### 3.3.2 Pooling réteg

Az úgynevezett „pooling” rétegek a konvolúciós neurális hálózattok architektúrájának fontos elemei. Elsődleges szerepük a bemenetek térbeli dimenziójának csökkentése, ami csökkenti a paraméterek és számítások számát a hálózatban. Ez a redukció döntő fontosságú a túlillesztés mérséklése és a számítási hatékonyság javítása szempontjából. A „pooling” rétegek a bemenet minden egyes mélységi szeletén egymástól függetlenül működnek és azt térbeli értelemben átméretezik. (Chollet, 2018; Géron, 2019; I. Goodfellow et al., 2016)
Több fajta pooling művelet létezik, az első és legtöbbet használt az a „max pooling”. A max pooling a legnagyobb elem kiválasztását jelenti a jellemzőtérkép azon régiójából, amit a szűrő lefed. Például egy 2x2-es szűrőben a négy érték közül a legmagasabbat választjuk ki. Ez a megközelítés különösen hatékony a legmarkánsabb jellemzők megragadásában és a bemeneti

képen a jellemzők pozíciójának kisseb eltéréseivel szembeni robusztusság megteremtésében. A következő az átlagos összevonás („average pooling”), ez a módszer a szűrő területén lévő összes elem átlagát számítja ki. Bár régebben elterjedt volt, a legtöbb mai CNN architektúrában nagyrészt felváltotta a „max pooling”, mivel ez utóbbi a gyakorlatban jobb teljesítményt nyújt (Boureau et al., 2010). A harmadik legelterjedtebb és az utolsó a most tárgyaltak között, az a „L2-Norm Pooling”, ez a gyakorlatban kevésbé elterjedt. Ez a módszer a szűrő területén lévő elemek négyzetösszegének négyzetgyökét számítja ki. Ez számításigényesebb az előző két esetben tárgyalt módszernél.




*ábra 8 Max pooling művelet (forrás: (Amidi, n.d.))*




*ábra 9 Átlagos összevonás művelet (forrás: (Amidi, n.d.))*

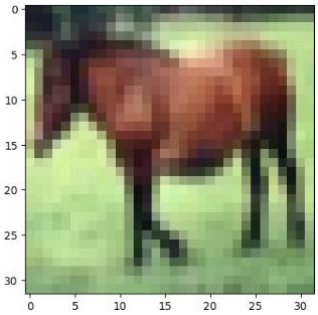
Ezek a műveletek nem csak a dimenzió és partaméter csökkentésben játszanak szerepet, hanem a transzformációkkal szembeni rugalmasságát is növeli a modellnek. Az összevonás a bemeneti adatok kis mértékű transzlációjával, elforgatásával és skálázásával szemben egy bizonyos szintű invariabilitást eredményez. Különösen a „max pooling” biztosítja a robusztusságát a modelnek a kisebb változásokkal és torzulásokkal szemben, ami általánosítóbbá teszi a hálózatot.

### 3.3.3 Teljesen összekapcsolt réteg

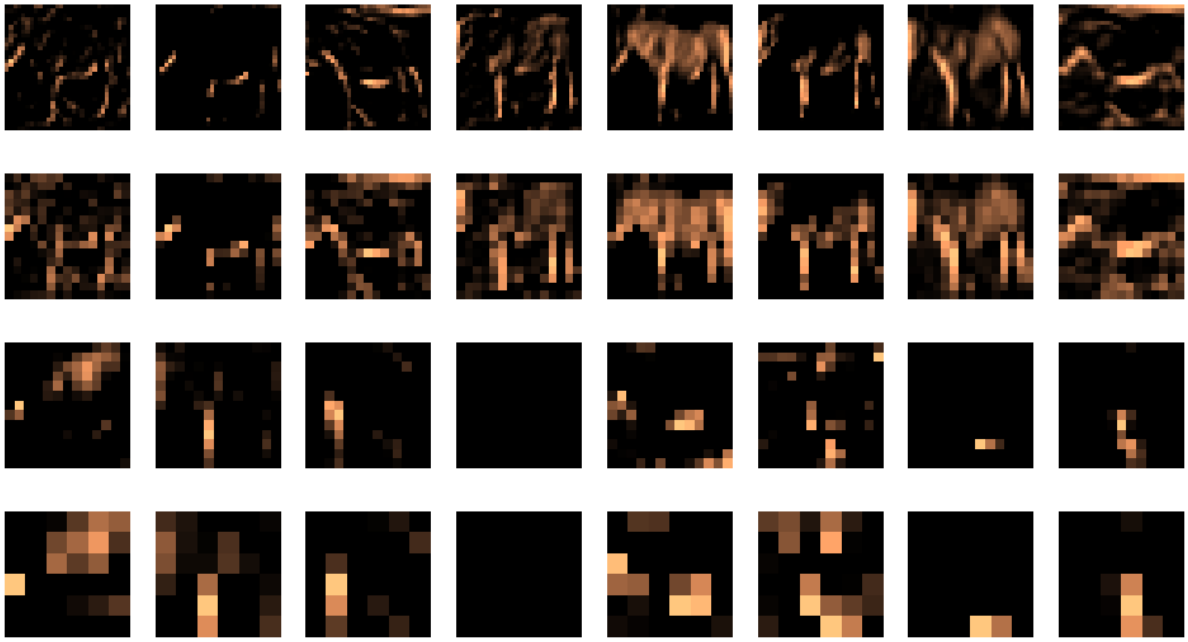
Az utolsó legfontosabb összetevője a CNN-eknek azok a teljesen összekapcsolt („fully connected”) rétegek. Ezek a rétegek gyakran a CNN architektúrák végén találhatók, döntő szerepet játszanak a döntéshozatali folyamatban (I. Goodfellow et al., 2016). A konvolúciós

és „pooling” rétegeken keresztül történő jellemzőkivonás után a teljesen összekapcsolt rétegek felelősek a kivont jellemzők végső kimenetekké való átalakításáért, mint például az olyan osztályozási feladatokban, mint amit én mutatok be a szakdolgozatomban, ahol a osztálypontszámokat ad a végső kimenet (I. Goodfellow et al., 2016). Lényegében a teljesen összekapcsolt rétegben lévő neuronok az előző réteg összes aktivációjával kapcsolatban állnak, ahogyan azt a hagyományos neurális hálózatokban látjuk. Ez az átfogó kapcsolódás biztosítja, hogy az teljesen összekapcsolt réteg globálisan megérti az előző rétegek által kinyert adatokat. Az teljesen összekapcsolt rétegben végzett műveletek úgy írhatók le, mint a bemeneti jellemzők mátrixszorzása, amelyet egy bias eltolás követ, ami a kimeneti értékek létrehozásához vezet. A teljesen összekapcsolt rétegek elsődleges feladata a konvolúciós rétegek által kinyert jellemzők értelmezése és az ezek alapján történő döntéshozatal. Például egy arcfelismerési feladatban, míg a konvolúciós rétegek olyan jellemzőket azonosítanak, mint az élek és a textúrák, a teljesen összekapcsolt rétegek meghatározhatják, hogy ezek a jellemzők egy adott személynek felelnek-e meg.

A teljesen összekapcsolt rétegek azonban nagyszámú paraméterrel rendelkeznek, ami túlillesztéshez vezethet, különösen olyan esetekben, amikor a képzési adatok korlátozottak. Ennek enyhítésére gyakran alkalmaznak olyan technikákat, mint a „dropout”, ahol a véletlenszerűen kiválasztott neuronokat figyelmen kívül hagyják a képzés során, ami segít a modell általánosító képességében. A CNN-ek szerves részét képezik a teljesen összekapcsolt rétegek, amelyek a konvolúciós rétegek által kinyert térbeli jellemzőket előrejelzésekké vagy osztályozásokká alakítják. Tervezésük és megvalósításuk kulcsfontosságú a CNN-modellek általános teljesítménye és hatékonysága szempontjából.




*ábra 10 Bemeneti kép a neurális hálózatba (forrás: CIFAR-10 )*



*ábra 11 Konvolúciós jellemzőkinyerés egy ló képén keresztül (forrás: eredeti kép CIFAR-10)*

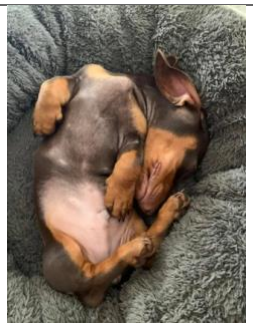
# 4. Adatelőkészítési lépések és egyéb technikák a modellezésben

## 4.1 Adatbővítés

Az adatbővítés létfontosságú lépés a képfeldolgozás területén, emiatt minden kép klasszifikációs feladat elején fontos ezzel is foglalkoznunk. A valós életben tele vagyunk képi elemekkel és ezek majdnem minden vizuális esetben nagyon komplexek. Itt arra kell gondolnunk, hogy gyakorlatilag lehetetlen egy kép minden lehetséges variációját rögzíteni. Az adatbővítés („data augmentation”) ezt a feladatot hívatott végrehajtani az adathalmaz mesterséges bővítésével. A folyamat a képek méreteinek változtatásával és azok sokféleségének növelésével javítja a modell általánosítási képességét. (Géron, 2019)

Miért fontos a képosztályozás szempontjából?
A való világ még egy egyedi tárgynak is rengeteg változatát kínálja. Itt gondolhatunk például a tacskókra, ami változhat a szín, a szőrzet, a pozíció, a fényviszonyok és a háttér tekintetében bárhogy. A bővítés ezt próbálja elősegíteni, hogy a modellünk minél több ilyen variációnak legyen kitéve.

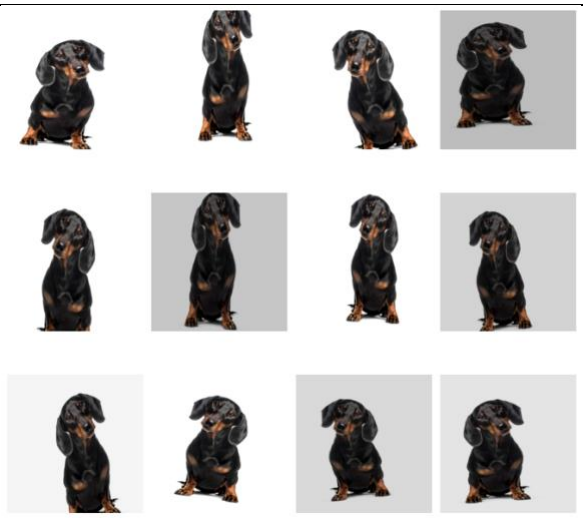
*ábra 12 Tacskó fajták szemléltetése (forrás: (*Wire Haired Dachshunds*, 2021))*



*ábra 13 Az alakok és formák különbözőségének szemléltetése (forrás: (*Sleeping Dachshund*, n.d.) )*

*ábra 14 Az alakok és formák különbözőségének szemléltetése (forrás: (*Dogphotographer.co.uk Kurt Pas*, 2023) )*

Másik oka még ennek a megoldásnak, hogy a kisebb adathalmazoknál fennáll a kockázata a túlillesztésnek, mivel ilyenkor a modellek a gyakorló halmaz mintáit memorizálják, az általánosítás helyett. Másrészről, a bővítés növeli az adathalmaz effektív méretét. A harmadik ok, a kibővített adatokon tanult modellek ellenálnak a különböző valós torzulásoknak, például a megvilágítás változásainak vagy a különböző kameraszögeknek.



*ábra 15 Adatnövelési technikák szemléltetése (forrás: eredeti kép (*Dachshund Dog Stock Photo*, n.d.))*

*ábra 16 Adatnövelési technikák szemléltetése véletlen kivágással (forrás: eredeti kép (*Dachshund Dog Stock Photo*, n.d.))*

Gyakran használt adatbővítési technikák:

- Megfordítás („flipping”): A tükrözés során a képet vízszintesen vagy függőlegesen megfordítjuk. Ez a transzformáció egyszerűen növeli az adathalmaz sokféleségét és segít abban az esetben, ha olyan adatkészlettel dolgozunk, amelynek eredendően irányfüggő torzításai vannak.
- Forgatás („rotation”): Ezzel a technikával a képeket egy meghatározott intervallumon belül elforgatjuk egy adott szöggel.
- Eltolás („translation”): A kép vízszintes, függőleges vagy minkettő irányba történő eltolását jelenti. Ez a módszer nagyon hasznos, mivel a legtöbb objektum a képen szinte bárhol elhelyezkedhet.
- Nyírás („shearing”): Ahogyan az előbbiekkel így a nyírással is találkozhattunk a mátrix algebrában. Ezzel a módszerrel a képet egy adott tengely mentén torzítjuk, vízszintesen, akár függőlegesen.
- Fényerő és kontraszt változtatás: Ezek az átalakítások a képek fényerősségét és a kontrasztarányt állítják át és a fényviszony változását szimulálják.
- Véletlenszerű kivágás („random cutout”): Ezzel a megoldással egy téglalap véletlenszerű kiválasztását és majd nullákkal vagy véletlen értékekkel töltjük fel. Ez arra kényszeríti a modellt, hogy hiányos adatokból tanuljon. A való életben is gyakran előfordul, hogy az adott tárgyat valami eltakarja.
- Gauss zaj („Gaussian noise”): A Gauss-zaj a statisztikai zaj egy olyan típusa, amely normális eloszlást követ. Képeknél alkalmazva több célt is szolgál. A zaj hozzáadása a tanulás során növelheti a modell robusztusságát, mivel biztosítja, hogy a modell nem alkalmazkodik túlságosan a pixelpontos részletekhez. Továbbá, bizonyos területeken, például az orvosi képalkotásban, a Gauss-zaj felhasználható a valós mérési zaj szimulálására. Mátrixalgebrai kifejezésekkel élve, a Gauss-zaj bevezetése a képmátrixhoz egy "zajmátrix" (Gauss-eloszlásból vett értékekkel) elemenkénti hozzáadását jelenti. Ez a folyamat megváltoztatja a kép pixelértékeit anélkül, hogy a kép teljes mérete vagy szerkezete megváltozna.

(*Building Powerful Image Classification Models Using Very Little Data*, n.d.; Géron, 2019)

Összefoglalva, az adatbővítés kulcsfontosságú eszközzé vált a képosztályozásban, hogy javítsa a modell azon képességét, hogy a különböző valós helyzetekben általánosítani tudjon. Azáltal, hogy variációkat vezetünk be az adatkészletünkbe, szimuláljuk a számtalan kihívást, amellyel egy algoritmus találkozhat. A bővítési technikák hatékonysága azonban, a megfontolt alkalmazásukon múlik. Feltétlenül meg kell különböztetni, hogy mely átalakítások relevánsak

az adott feladat szempontjából. Bár számos orientáció generálása lehetséges, nem mindegyik releváns minden alkalmazáshoz. Egy fejjel lefelé fordított autó lehet, hogy releváns egy biztosítási kárigény osztályozó számára, de irreleváns a szokásos közúti helyszínek klasszifikálása szempontjából. Ezért a megfelelő egyensúly megtalálása kulcsfontosságú. Ahelyett, hogy válogatás nélkül alkalmaznánk minden technikát, figyelembe kell vennünk a feladat kontextusát és a modell lehetséges valós forgatókönyveit. Lényegi megfontolások szerint az adatbővítés, bár jelentős segítséget nyújthat a modell megbízhatóságának fokozásában, igazi potenciálja az átgondolt, valamint a kontextus figyelembevételével történő alkalmazásában mutatkozik meg. Ez biztosítja, hogy adataink diverzifikálása során az adatok valóságos szituációkhoz való igazodását is szem előtt tartsuk.

## 4.2 Normalizálás

Az előfeldolgozás lépései között, kritikus szerepe van a normalizálásnak és gyakran meghatározza a későbbi modellezési próbálkozások sikerét. Ez a folyamat azt jelenti, hogy a képpontok intenzitásértékeinek tartományát úgy állítjuk be, hogy az egy adott skálába illeszkedjen. A képek ilyen szabványosítása biztosítja, hogy a CNN-modellbe táplált bemeneti adatok konzisztensek legyenek, ezáltal segítve a tanulási folyamat optimalizálását. (Krizhevsky et al., 2017)

A CNN-ek mélytanulási jellegükből adódóan érzékenyek a bemeneti értékek tartományára és eloszlására. A következetlen vagy nem normalizált bemeneti adatok nem megfelelő tanuláshoz vezethetnek, ahol a modell inkább a bemeneti tartományok eltéréseinek megtanulására összpontosíthat, ahelyett, hogy érdemi jellemzőket nyerne ki a képekből. A normalizálási folyamat ezt a problémát enyhíti azzal, hogy garantálja minden bemeneti kép azonos skálán való reprezentációját, ami által a tanulási folyamat hatékonysága és stabilitása jelentősen növekszik (Simonyan & Zisserman, 2015).

A normalizálásnak több módját is használják a képi feldolgozás és konvolúciós hálózzatok esetén. Az első módszer, a normalizálás legegyszerűbb módja, a pixel újra skálázás („pixel rescaling”). Ebben az esetben a pixelértékeket a maximális pixelértékkel (általában 255) való osztással 0 és 1 közötti tartományba skálázzuk. Ezt a módszert széles körben használják az egyszerűsége és a magas hatékonysága miatt.

A következő módszer az átlag kivonás („mean subtraction”) és kontraszt normalizálás. Ezek során minden egyes pixelből kivonjuk az egyes képpontok átlagos értékét, és opcionálisan úgy méretezzük az értékeket, hogy a szórás egy legyen. Az ilyen normalizáció segít az adatok

optimalizálásában és a megvilágítási különbségek hatásának csökkentésében. (Krizhevsky et al., 2017)

A harmadik és egyben most bemutatott módszerek között az utolsó az a ZCA fehérítés ("ZCA whitening"). Ez az eljárás egy lépéssel továbbmegy a képadatok lineáris transzformálásával olyan módon, hogy a kovariancia mátrixa az azonossági mátrix legyen. Ez csökkenti a pixelképek redundanciáját a mátrixban, kiemelve a képeken belüli struktúrát és jellemzőket. (Coates et al., 2011)

A CNN modellekre nagy hatása van normalizálásnak. A normalizált képek gyorsabb konvergenciához vezetnek a képzés során, mivel a gradiens süllyedés szerinti optimalizációs eljárás hatékonyabban tud dolgozni a standardizált adatokon. Segít továbbá csökkenteni a modell érzékenységét a különböző képek közötti fény- és színeloszlásbeli eltérésekre (Ioffe & Szegedy, 2015).

## 4.3 Batch normalizáció

Az (Ioffe & Szegedy, 2015) által 2015-ben evezetett „batch" normalizáció (BN) a mély neurális hálózatok optimalizálásának egyik legfontosabb elemévé vált, különösen a konvolúciós neurális hálózatokkal (CNN) történő képosztályozással összefüggésben. Elsődleges funkciója nem önmagában optimalizációs algoritmusként, hanem az adaptív újra paraméterezés módszereként működik. Ez kulcsfontosságú a nagyon mély modellek tanításával kapcsolatos kihívások kezelésében (I. Goodfellow et al., 2016).

A mély neurális hálózatok tanítása több függvény vagy réteg felépítését foglalja magában. A komplexitás akkor jelentkezik, amikor az egyik réteg paramétereinek frissítései véletlenül befolyásolják a következő rétegek bemeneteinek eloszlását, ez a jelenség belső kovariancia-eltolódás néven ismert (Ioffe & Szegedy, 2015). Ez a probléma megnehezíti a képzési folyamatot, megnehezítve a megfelelő tanulási sebesség kiválasztását, és gyakran kisebb tanulási sebességek alkalmazását teszi szükségessé a modell konvergenciájának biztosítása érdekében. A „batch" normalizáció (BN) ezeket a kihívásokat az egyes rétegek bemeneteinek normalizálásával oldja meg. Ez magában foglalja az egyes rétegek aktivációinak beállítását és skálázását, hogy azok átlaga nulla és varianciája egy legyen. Pontosabban, egy adott réteg esetében a BN kiszámítja az aktiválások átlagát és szórását az aktuális mini-batch alatt, és ezeket a statisztikákat használja a kimenetek normalizálásához. Ez a normalizálási folyamat döntő fontosságú a tanulási folyamat stabilizálásában, és lehetővé teszi a nagyobb tanulási

sebességek használatát, ami jelentősen felgyorsíthatja a mély hálózatok képzését. (Ioffe & Szegedy, 2015)
A batch normalizáció bármely bemeneti vagy rejtett rétegre alkalmazható és a következő lépéseket foglalja magában (Ioffe & Szegedy, 2015):

A BN technika első lépése, hogy minden egyes mini-batchre (B) kiszámítja az aktivációk átlagát ($\mu_B$) és szórását ($\sigma_B^2$) ezután normalizálja azokat.

Adott egy B mini-batch m mintával $\left(x^{(1)}, x^{(2)}, \dots, x^{(m)}\right)$, ahol minden $x^{(i)}$ egy vektor, amely az $i$-edik minta egy rétegének aktivációját ábrázolja.

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m} (x^{(i)} - \mu_B)^2$$

Itt $\mu_B$ és $\sigma_B^2$olyan vektorok, amelyek minden eleme egy jellemzőnek („feature") felel meg.

A következő lépés az aktivációk normalizálása. Ez a lépés biztosítja, hogy az aktivációk átlaga nulla, szórása pedig egy legyen. A normalizált értékeket $\hat{x}^{(i)}$ minden egyes mintára a következőképpen számoljuk ki.

$$\hat{x}^{(i)} = \frac{x^{(i)} - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}$$

Ebben az egyenletben $\varepsilon$ egy kis konstans, amelyet a numerikus stabilitás érdekében adunk hozzá, a nullával való osztás elkerülése érdekében.

A normalizálás után a batch normalizáció aktiválásonként két új paramétert vezet be - egyet a méretezéshez ($\gamma$), egyet pedig az eltoláshoz ($\beta$). Ezeket a paramétereket a képzés során tanulják meg, és lehetővé teszik a hálózat számára, hogy visszavonja a normalizálást, ha az a tanulás szempontjából előnyös.

$$y^{(i)} = \gamma\hat{x}^{(i)} + \beta$$

Itt $y^{(i)}$ a batch normalizációs réteg kimenete.
A következő lépés a back-propagation kiigazítása. A batch normalizáció módosítja a back-propagation algoritmust is, biztosítva, hogy a gradiensek megfelelően skálázódjanak, és hogy a normalizálási lépések ne akadályozzák a hálózat más rétegeinek tanulását.

Mély neurális hálózatokra alkalmazva, különösen olyan feladatokban, mint a képosztályozás, a batch normalizáció bizonyítottan jelentősen javítja a teljesítményt. Lehetővé teszi szaturáló aktiválási függvények (szigmoid, tanh) és nagyobb tanulási sebességek használatát, ami gyorsabb konvergenciához és jobb általános pontossághoz vezet. (Ioffe & Szegedy, 2015)
Bár a BN számos előnnyel jár, ugyanakkor további komplexitást is bevezet a modellbe. Mind a képzés, mind a következtetés során növeli a számítási terhelést, amit más technikákkal ugyan kitudunk küszöbölni viszont ezt is be kell építenünk a modellbe.
Összefoglalva, a batch normalizálás a mély neurális hálózatok hatékonyabb és eredményesebb képzésének meghatározó technikájává vált. A belső kovariancia-eltolódás problémájának kezelésével gyorsabb tanulási időt, magasabb tanulási rátát tesz lehetővé, és javíthatja a modell teljesítményét. Hatását jól mutatja, hogy széles körben elterjedt a modern mélytanulási architektúrákban (Ioffe & Szegedy, 2015).

## 4.4 Előre képzett hálózatok („Pretrained networks”)

Az előképzett hálózatok a modern mélytanulás egyik kulcsfontosságú eleme, különösen az olyan feladatok esetében, mint a képosztályozás. Ezek olyan neurális hálózatok, amelyeket korábban nagy és széleskörű adathalmazokon, például az ImageNet-en (Deng et al., 2009) képeztek ki, amely több millió címkézett képet tartalmaz több ezer kategóriában (Krizhevsky et al., 2017). Ezek a hálózatok, miután ezekből a kiterjedt adathalmazokból megtanulták a jellemzők széles skáláját, jó kiindulópontként szolgálnak az új képosztályozási feladatokhoz.

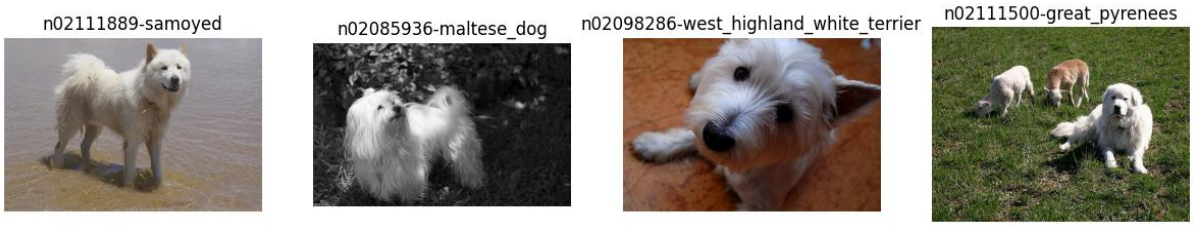
Az előképzett hálózattoknak több előnye is van. Az első, amit megemlíthetünk, hogy a modelleknek tanítása csökken, valamint számítási hatékonysága nő. Egy modell nulláról történő kiképzése jelentős számítási erőforrásokat és időt igényel (Géron, 2019)l, különösen a nagy hálózatok esetében. Az előképzett hálózatok, amelyek már megtanultak egy általános reprezentációt, jelentősen csökkentik mind a számítási terhet, mind a képzési időt (Razavian et al., 2014).
A következő, hogy a mélytanulás egyik legnagyobb kihívása a nagy méretű címkével ellátott adathalmazok igénybevétele. Az előképzett hálózatok azonban viszonylag kisebb adatkészletekkel is finomhangolhatók, miközben még mindig nagy teljesítményt érnek el. Ez különösen előnyös olyan területeken, ahol az adatgyűjtés és a címkézés drága vagy nem lehetséges (Razavian et al., 2014).

# 5. Az adathalmaz és modellek összehasonlítása

## 5.1 A felhasznált adathalmaz – „Stanford Dogs”

A különböző modellek és technikák bemutatására én a Stanford dogs adathalmazt választottam (Khosla et al., n.d.). Ez az adathalmaz a számítógépes látás területén egy kiváló gyűjtemény, 20 580 képet tartalmaz 120 kutyafajtáról. Az ImageNet-ből származó adatkészlet a finomabb képosztályozásra van szabva, és kihívás elé állítja a gépi tanuló és mesterséges intelligencia modelleket.

Az adathalmaz elsődleges jellemzője a kutyafajták széles skálája. Minden egyes fajta egyedi tulajdonságokkal rendelkezik, ugyanakkor hasonlóságokat is mutat másokkal. Ez a sokféleség kulcsfontosságú olyan algoritmusok kifejlesztésénél és alkalmazásánál, ahol szükség van a legapróbb részletek és különbségek megkülönböztetésénél, ami valójában a valós alkalmazást tükrözi.



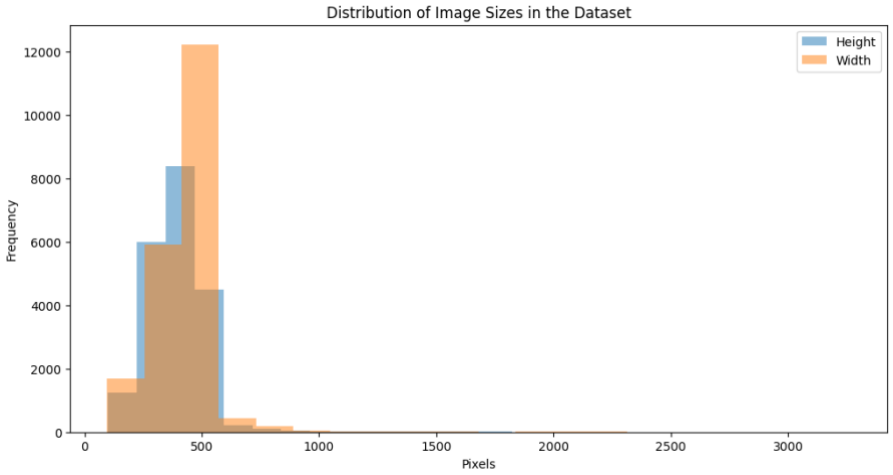

*ábra 17 Különböző fajtájú kutyák képei a Stanford Dogs adathalmazból (forrás: (Khosla et al., n.d.))*

Az osztályozási feladatot tovább nehezíti az adathalmaz részletes jellege. A fajták megkülönböztetése megköveteli, hogy az algoritmusok felismerjék és megtanulják a komplex jellemzőket, amelyek gyakran nem különülnek el egymástól. Az adathalmazban szereplő képeket a pózok, a környezet, a fényviszonyok és a minőség vonatkozásában jelentős eltérések jellemzik. Ez a változékonyság nagyban tükrözi a valós élethelyzeteket, ahol a kontrollált képfelvétel nem mindig lehetséges, így ideális terep a robusztus modellek gyakorlásához. Ezeknek a modelleknek nemcsak pontosnak, hanem elég sokoldalúnak is kell lenniük ahhoz, hogy a képi feltételek széles skáláját kezelni tudják.

A képek jellemzőinek vizsgálata két nagyon fontos aspektusát tárta fel az adathalmaznak. Az első a képméretek változatosságát, valamint a képek eloszlását címkék, osztályok között. Ezek a tényezők alapvető fontosságúak az adathalmaz összetételének megértésének, mivel ezután a gépi modellek részére előfeldolgozást kell végrehajtanunk.

Az adatkészlet széles mérettartományban tartalmaz képeket. Ez a változatosság azért jelentős, mivel a gépi modellek, különösen a konvolúciós neurális hálózattokon alapuló modellek a legtöbbször egységes méretű bemeneti képeket igényelnek.

Az adatkészlet különböző méretű képeket tartalmaz, mind magasságban, mind szélességben. Ez a változékonyság az adathalmaz nem egységes jellegét tükrözi, ahol a képek különböző beállításokból és körülményekből származnak. Bár a pontos átlagos méret az elemzett adathalmaz adott részhalmazától függően változhat, általában a képméretek széles spektruma létezik. Az egész adathalmazt vizsgálva az átlagos képmértet 385 x 442 pixel szélességű és magasságú.




*ábra 18 Képek méretének megoszlása a Stanford Dogs adathalmazban (saját forrás)*

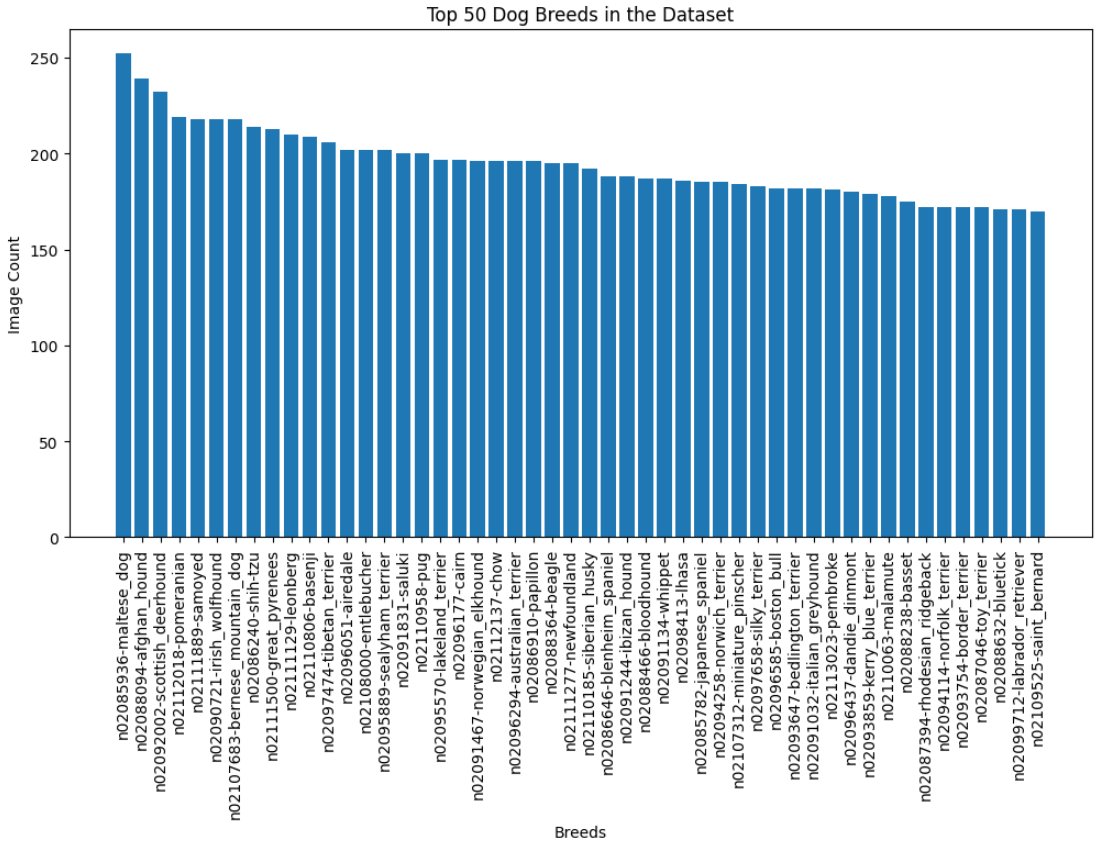
Tekintettel erre a méretbeli változatosságra, az olyan előfeldolgozási lépések, mint a méretezés, a képkivágás vagy a kitöltés elengedhetetlenek a képek szabványosításához, mielőtt betápláljuk őket a neurális hálózatba.

A következő pont, amit feltártátam az képek eloszlása az osztályok között. Az adathalmazra jellemző az osztályok kiegyensúlyozatlansága, egyes fajtákat több kép képvisel, mint másokat. Ez a kiegyensúlyozatlanság kihívásokhoz vezethet a modellek képzése során, mivel azok a gyakrabban képviselt fajták irányába torzíthatnak.

A kevesebb képpel rendelkező fajták esetében előfordulhat, hogy a modell nem tanul meg elegendő jellemzőt a pontos osztályozásukhoz. Ezzel szemben a több képpel rendelkező fajták túlreprezentáltak lehetnek a modell előrejelzéseiben.

Olyan technikák, mint az adatbővítés, az osztályok súlyozása alkalmazhatók az egyensúlyhiány hatásainak enyhítésére. Annak biztosítására, hogy a modell a rendelkezésre álló képek számától

függetlenül minden fajta esetében egyformán jól teljesítsen, alapvető fontosságú egy korrekt és hatékony osztályozási rendszer kifejlesztése.




*ábra 19 Képek számának megoszlása a címkék között a Stanford dogs adathalmazban (saját forrás)*

Fontos szem elött tartani, hogy a képméretek változatosságát kezelő előfeldolgozási lépések és az osztályok kiegyensúlyozatlansága elleni technikák elengedhetetlenek ahhoz, hogy az adathalmazban és a modelljeinkben rejlő lehetőségeket teljes mértékben kiaknázzuk.

## 5.2 LeNet-5

A Yann LeCun által kidolgozott és 1998-ban bemutatott LeNet arhitektúra a konvolúciós neurális hálózatok területén meghatározó fejlesztést jelentett (Lecun et al., 1998a). A LeNet architektúra jelentős elmozdulást mutatott a hagyományos neurális hálózatokhoz képest, mivel egy kifejezetten képfelismerési feladatokra szabott struktúrát vezetett be. A LeNet-5, a leghíresebb változata, lényegében a CNN-ek alapvető elveit testesíti meg, amelyek a mai napig relevánsak a modern architektúrában.

A LeNet-5 két fő részből áll, egy konvolúciós blokkból és egy teljesen összekapcsolt blokkból. A hálózat 5 réteggel rendelkezik, tanulható paraméterekkel. Három konvolúciós réteggel alkotja és ezekkel kombinálva több átlagos összevonás („average pooling”) tartozik. A konvolúciós és „pooling” rétegek után, két teljesen összekapcsolt rétegünk van, végül egy

Softmax osztályozó, amely a hálózat legvégén a képeket a megfelelő osztályba sorolja. Történelmileg a LeNet-et szürkeárnyalatos képekre alkalmazták, de alapelvei a modern RGB-képekhez is adaptálhatók.

LeNet-5 minden egyes konvolúciós rétege tanulható szűrők (kernelek) együttesével van elláttatva, amelyek térben (magasságban és szélességben) kicsik, és a bemeneti tartomány teljes mélységére kiterjednek. Ahogy ezek a szűrők végig csúsznak a bemeneti képen, olyan jellemzőtérképeket hoznak létre, amelyek a nyers pixeladatokból magasabb szintű jellemzőket absztrahálnak. Az első réteg jellemzően 6, míg a második 16 kimeneti csatornával rendelkezik. A konvolúciós rétegek szigmoid aktiválási függvényt használnak, ami a ReLU (Rectified Linear Unit) aktivitási függvény elterjedése előtti korszakot tükrözi. (Lecun et al., 1998b)

Minden egyes konvolúciós művelet után egy „average pooling” ("subsampling") réteg következik. Ez a réteg csökkenti a bemeneti térfogat térbeli dimenzióit (magasságát és szélességét) a következő konvolúciós réteg számára, hatékonyan tömörítve az információt és csökkentve a következő rétegek számítási terhelését.

A konvolúciós blokk után a hálózat egy sor teljesen összekapcsolt rétegre tér át. Ez a sűrű („dense”) blokk ellaposítja a konvolúciós rétegek kimenetét, és három teljesen összefüggő rétegen keresztül továbbítja azt. Az első két „dense” réteg 120, illetve 84 csomóponttal rendelkezik, és ugyanazt a szigmoid aktiválási függvényt használja.

A LeNet-5 utolsó rétege egy softmax kimeneti réteg, amelyet osztályozási célokra használnak. Ez egy valószínűségi eloszlást ad ki a célosztályokra, ahol minden egyes csomópont egy adott osztálynak felel meg.

A LeNet-5 számos jellemzője innovatív volt a bevezetésekor. A teljesen összekapcsolt hálózatokkal ellentétben a konvolúciós réteg minden egyes neuronja csak a bemeneti térfogat egy kis régiójához kapcsolódik. Ez a helyi kapcsolódási mintázat lehetővé teszi a hálózat számára, hogy a korai rétegekben alacsony szintű jellemzőkre összpontosítson, és ezeket a jellemzőket a mélyebb rétegekben magasabb szintű reprezentációkba állítsa össze.

A második innovatív megoldás közös megosztott súlyok és a térbeli invariancia. A megosztott súlyok használata a konvolúciós rétegekben nem csak a modell memóriaigényét csökkenti, hanem lehetővé teszi a modell számára a térbeli invariáns jellemzők tanulását is, ami azt jelenti, hogy a hálózat képes felismerni a jellemzőket, függetlenül azok térbeli elhelyezkedésétől a bemeneti képen.

Az eredeti LeNet-5 modell fútatása során több kulcsfontosságú felismerés rajzolódik ki a modell viselkedésével kapcsolatban. A modell 15 epochon keresztül futott, mind a tanulási mind a validálási veszteség fokozatos csökkenést mutatott. Annak ellenére, hogy LeNet

úttörőnek számított a CNN-ek területén, jelentős korlátokat mutat egy ennyire összetett képosztályozási feladatban. A modell alacsony pontossággal, 0,6% és magas „loss” értékkel jellemezhető teljesítménye azt jelzi, hogy a 120 kutyafajta megkülönböztetése nehézségekbe ütközik. Ebből több tényezőt emelnék ki, ami segít megérteni a teljesítményt.

Az első a képek összetettsége, a LeNet-5 architektúrát eredetileg egyszerűbb, alacsony felbontású képekre tervezték és használták abban az időben. Szürke árnyalatos, 32 x 32 pixeles számjegyeket ismertek fel vele. Ez az adathalmaz nagy felbontású és részletesebb képeket tartalmaz, valószínűleg meghaladják a modell jellemző kivonási képességét, még annak ellenére is, hogy nagyobb 64 x 64 pixeles képekké alakítottam őket. Érdemes megjegyezni hogy az átlagos képméret 385 x 442 pixel ami ekkora csökkentés mellet jelentős adatveszteség.
A modellt több változatban kisebb módosításokkal is fútattam. Az első változatban nem változtattam a rétegeken, aktivációs függvényen egyéb paramétereken. Az adatbővítés bevezetésének célja az volt, hogy a modell általánosító képeségét a tanuló adatok változatosabbá tételével javítsa. A teljesítménymutatók azonban nem mutattak jelentős javulást az eredeti LeNet-5 modellhez képest amikor csak az eredeti nem bővített adatokon futott. Ezek az eredmények arra utalnak, hogy bár az adatbővítés általánosságban előnyös, viszont a hatékonyságának határt szabnak a modell alapjául szolgáló arhicteturális elemek. A teljesítménymutatók gyene eredményei azt mutatják nem volt képes a modell leküzdeni az összetett képi adathalmaz jelentette kihívásokat.
A következő módosított modell, tartalmazott egy további sűrű („dense”) réteget és egy „dropout” réteget. A célom az volt, hogy a modell tanulási teljesítményét és a túlillesztés csökkentését elérjem. A teljesítmény mutatók azonban nagyrészt változatlanok maradtak. A pontosság érdemben nem változott, viszont a veszteség függvény értékei arra utalnak, hogy jobban sikerült a modellnek elkerülnie a túlillesztést. Az elmondható, hogy a modell összetettségének növelése nem mindig vezet a drasztikusan jobb teljesítményhez, az alap architektúra itt is meghatározó.
Összevetve a modelleket elmondható, minden modellváltozat esetében a tartósan alacsony pontosság és a magas veszteség egy közös pontra utal. A Stanford Dogs adatkészlet a nagy felbontású képeivel és részleteivel, valamint számos osztályával olyan osztályozási kihívást jelent, ami a LeNet-5 alapmodell alkalmazási körén túlmutat, függetlenül az elvégzett módosításoktól.

### LeNet-5 Relu aktivációs függvénnyel

A klasszikus LeNet-5 modellben más aktivációs függvényt választottam, a ReLU aktivációs függvény a modern architektúrákban alkalmazzák a jobb eredményei miatt. Az adaptálása jelentős javulást mutatott az eredeti szigmoid alapú modellhez képest. A ReLU aktivállassal a modell körülbelül 27,8%-os tanulási pontosságot és 4,4%-os validálási pontosságot ért el, ami drasztikus emelkedés a tanulási folyamat során. A képzési és validálási eredmények közötti eltérés azonban a túlillesztés irányába mutat, mivel a modell szorosan a képzési adatokhoz igazodik.

A túlillesztés kiküszöbölésére itt is tesztelésre került az adatok kibővítése. Bár ez a megoldás csekély mértékben, de javította a validációs pontosságot és nagyban csökkentette a tanuló adatkészleten is a túlillesztést, ami azt jelenti, hogy a megoldás jó irányba mozdította el a modell teljesítményét. Viszont a fontos megjegyezni, a validációs csúcsérték hasonlóan 4,6% körül mozgott.

A tanulási ráta csökkentése és egy „dense” valamint egy „dropout” réteg hozzáadása a modellhez, lassabb tanulási sebességet viszont stabilabb eredményt tudott felmutatni, valamint a modell reziliensebb lett a túlillesztésre. Az eredmények nem mutattak nagyobb eltérést a tesztelési pontosságban.

A képméretet az előfeldolgozás során 64 x 64 pixelre csökkentetem az előző modellek tekintetében azonban, ha a LeNet5 architektúrával és a képbővítéssel együtt 224 x 224 pixeles bemenettel együtt vizsgáljuk a teljesítményt akkor itt is előrehaladást érünk el a pontosság és a veszteség metrikákat megnézve. A pontosság a validációs halmazon 6% körül mozog, a veszteség érték 4,58. A pontosság érték itt mutatja a legjobb értékeket az összes eddigi modellt figyelembe véve.

Összefoglalva, minden modellváltozatban megfigyelhető a tanuló adatokon mért pontosság fokozatos javulása, viszont ezek a javulások azonban korlátozottak. A legnagyobb változást a ReLU aktiváció tudta kiváltani, ha a klasszikus LeNet-5 modelleket is figyelembe vesszük. A modellekről elmondható, hogy tanulnak a képzési adatokból viszont ezeket nehezen tudják az új adatokra általánosítani.

## 5.3 AlexNet

Az AlexNet jelentős mérföldkő a képosztályozás területén, különösen a mély, konvolúciós neurális hálózattok használata tekintetében. A (Krizhevsky et al., 2017)) által kifejlesztett AlexNet architektúra és annak képzési módszerei új szabványokat állítottak fel az képfelismerési módszerek között. Ahogy előzőleg láttuk a LeNet modell volt az úttörő a CNN

modellek között viszont az AlexNet volt az, amely teljes mértékben bebizonyította a mély neurális hálózattok erejét a nagyméretű képosztályozásban.

Az AlexNet nyolc tanítható paraméterekkel rendelkező rétegből áll, amelyek közül öt konvolúciós réteg, három pedig teljesen összekapcsolt réteg.

Az első réteg a 224×224×3 bemeneti képet 96 darab 11×11×3 méretű, 4 pixeles lépésközzel rendelkező kernellel szűri. A nagy receptív mező és a lépésköz azért van kialakítva, hogy több információt nyerjen a nagy felbontású bemeneti képekből.

Az első réteg kimenete egy max-pooling rétegen halad át, majd a második konvolúciós rétegbe kerül. Ez a réteg 256 darab 5×5×48 méretű kernelt használ, amely az első réteghez képest sűrűbb és részletesebb jellemző-kivonatolást biztosít.

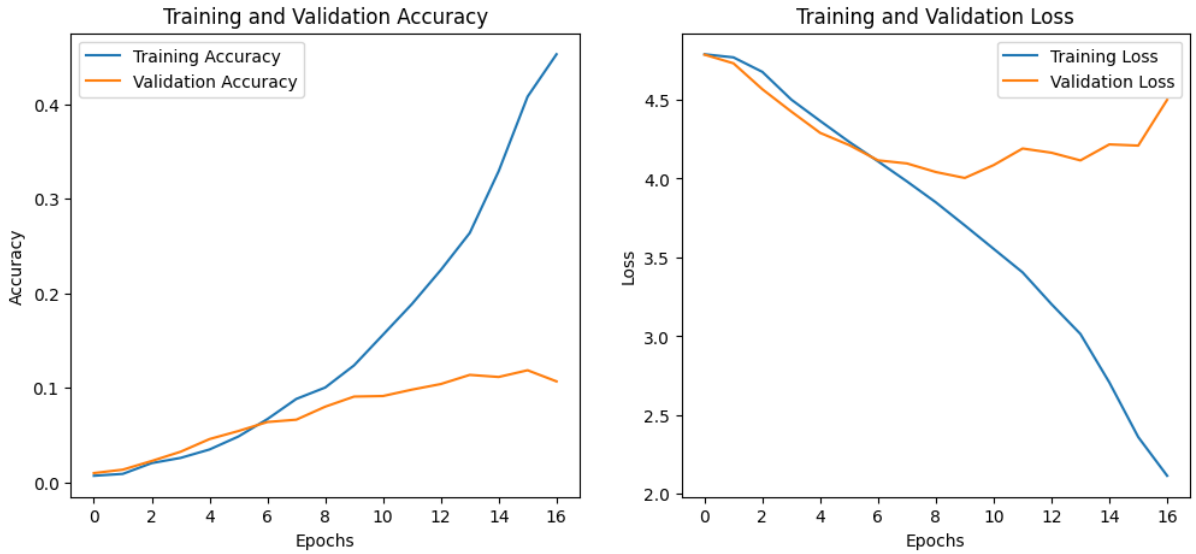
A harmadiktól az ötödik konvolúciós rétegek köztes pooling vagy normalizáló rétegek nélkül vannak összekapcsolva, ami összetettebb és mélyebb jellemzőhierarchiát tesz lehetővé. A harmadik réteg 384 darab 3×3×256 méretű kernellel, a negyedik réteg 384 darab 3×3×192 méretű kernellel, az ötödik réteg pedig 256 darab 3×3×192 méretű kernellel rendelkezik.

A konvolúciós rétegek után az AlexNet három teljesen összekapcsolt réteggel rendelkezik, amelyek egyenként 4096 neuronnal rendelkeznek. Az utolsó réteg egy 1000-utas softmax függvényhez van csatlakoztatva, amely 1000 osztályra vonatkozó valószínűségi eloszlást ad ki. (Krizhevsky, 2009)

Az AlexNet volt az egyik első olyan modell, amely az aktiválási függvényhez ReLU (Rectified Linear Units) funkciót használt, ami a hagyományos szigmoid vagy tanh függvényekhez képest gyorsabb képzést segített. Valamint az AlexNet az első és a második konvolúciós réteg között használ egy úgynevezett „local response” normalizációt, ez a technika a biólogia neuronok álltal ikhletett folyamat. A modell a neuronok kimenetei között közötti „verseny” elősegítésével fokozza a modell képességét a változatos és bonyolultabb jellemzők megtanulásásra. A legutolsó jelentős változás a „pooling” technikát érinti, a hagyományos, nem egymást fedő összevonási módszerekkel ellentétben az AlexNet átfedő összevonást alkalmaz. Ez a megközelítés egy 3×3 pixeles pooling ablakot foglal magában 2 pixeles lépésközzel, ami lehetővé teszi a pooling területek fedését. Az átfedéses összevonás bizonyítottan csökkenti a modell hibaarányát, ami robusztusabbá teszi a modellt az osztályozási feladatokban.

Az eredeti AlexNet modellt eredetileg adatbővítés nélkül terveztek és mutattak be, én viszont az előző tesztelési folyamatot alkalmaztam, amit már a LeNet-5 modellnél is bemutatta. A modell architektúráját az eredeti AlexNet struktúrájával összhangban tartottam, ide sorolva a réteg sorrendet és mennyiségét, valamint a normalizációs lépéseket. Az alap modell tanuló

folyamata 17 epoch alatt következetesen javuló pontosságot mutatott, a tanuló halmazon 45,3%-os pontosságot, a validációs halmazon viszont csak 10,6% pontosságot ért el. Ezek az eredmények alátámasztják a modell képességét, hogy képes hatékonyan tanulni a képzési adatokból, viszont a két halmaz közötti eltérés alapján a modell túlillesztést mutat. Ha a vesztesé görbét is megnézzük, ugyan ezt a következtetést tudjuk levonni.

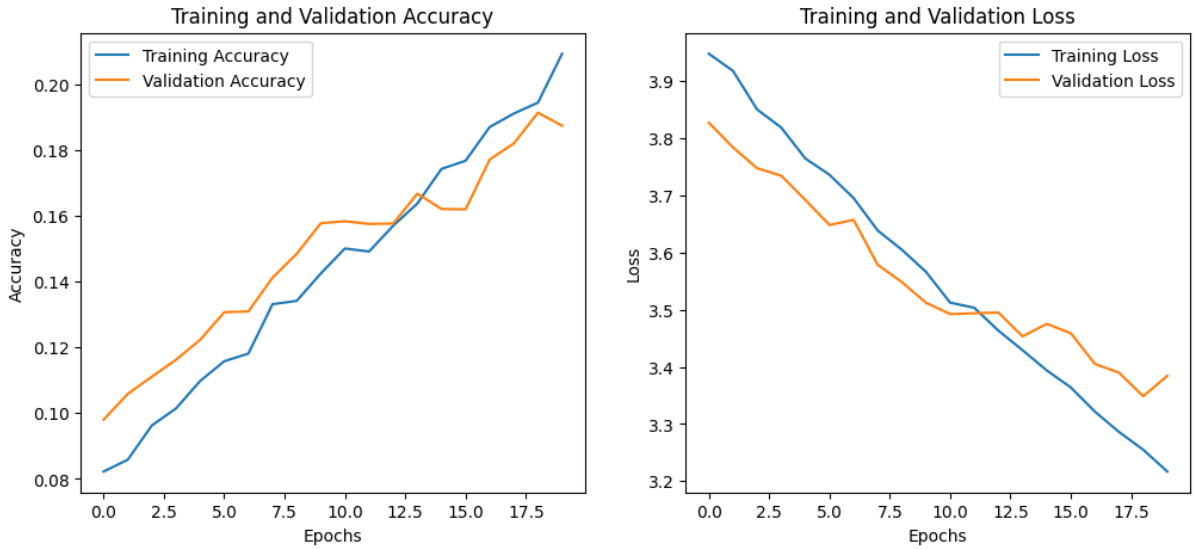



*ábra 20 A modell pontosságának és veszteségének alakulása a tanulási epochok során (saját forrás)*

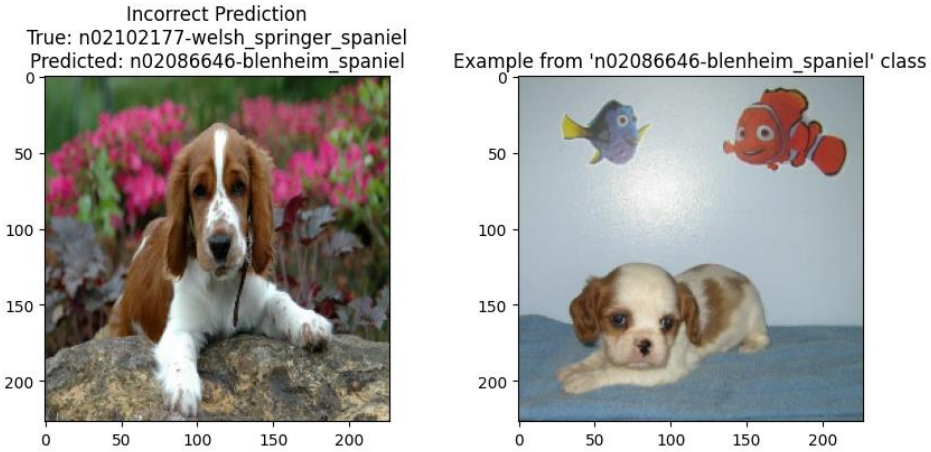
A második modell verzióban már adatbővítési technikát is alkalmaztam, beleértve a véletlenszerű vertikális tükrözést, forgatást és nagyítást 0,2 értékek mellet. A kiterjesztett modell lassabb de konzisztensebb javulást mutatott a pontosságban, ami jellemző a kiterjesztett adatokkal történő képzés során a képzési minták megnövekedett összetettsége és változatossága miatt. Ugyanezen 17 epocha végére a modell 17,8% körüli képzési pontosságot és 16,6%-os validálási pontosságot ért el. Figyelemre méltó, hogy a képzési és validálási pontosság közötti kisebb különbségek a kiterjesztett modellben a jobb általánosításra utalnak, annak ellenére, hogy az eredeti modellhez képest összességében alacsonyabb a pontosság. A két veszteség függvény a huszadik epoch körül kezd teljesen elválni egymástól ebben az esetben úgy tűnik, a paraméterek változtatására szorul, ha további eredményeket akarunk elérni a modellben.

Fontos kiemelni, hogy az eredeti és a módosított AlexNet modellek összehasonlító kiemeli az adatbővítés hatását a mélytanulási modellekre a képosztályozási feladatokban. Arra utal, hogy az adatbővítés javíthatja a modell robusztusságát és általánosítását. A LeNet-5 modelljeimnél említettem, hogy az architektúra meghatározó és nem feltétlenül segít, ha a paraméter módosítások vagy adatbővítést hajtunk végre. Ezeknél a modelleknél, jól láthatóan segít az

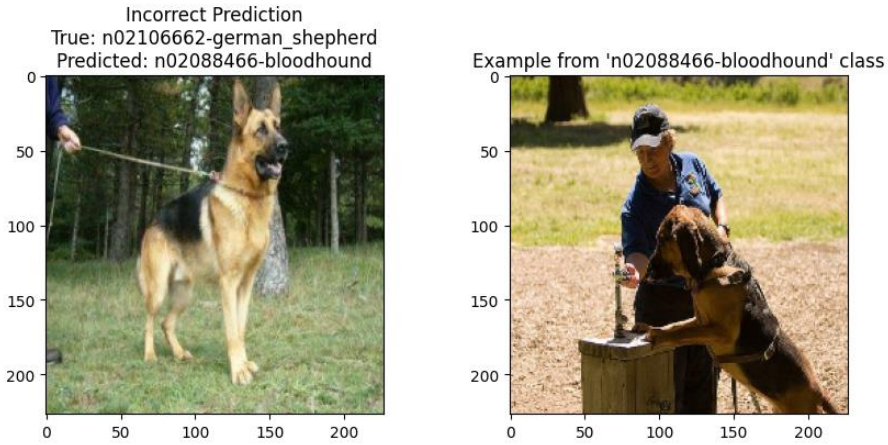
adatbővítés, a tanulási ráta módosítása a modellben, mivel a modellünk alapjaiban képes általánosítani és tanulni, egy szóval az alapképeségei megvannak és a módosításokkal csak segítünk a modellnek a jobb teljesítmény elérésében.




*ábra 21 A modell pontosságának és veszteségének alakulása a tanulási epochok során (saját forrás)*




*ábra 22 Helytelen klasszifikáció példája és a célosztály egy mintaképe (saját forrás)*




*ábra 23 Helytelen klasszifikáció példája és a célosztály egy mintaképe (saját forrás)*

## 5.4 VGGNet

Az oxfordi Visual Graphics Group kutatói által kifejlesztett VGGNet (Simonyan & Zisserman, 2015) (innen a VGG elnevezés) fontos állomás a képfelismerési feladatokhoz használt konvolúciós neurális hálózatok (CNN-ek) fejlődésében. Ez a Simonyan és Zisserman által 2015-ben bemutatott hálózat egyszerűségéről és mélységéről híres, és nagyban befolyásolta a vizuális adatok feldolgozására szolgáló mély CNN-ek tervezését.

A VGGNet architektúráját az egységesség és a mélység jellemzi. Kizárólag 3x3 egymásra épített konvolúciós rétegeket használ, amelyek mélysége elődeihez képest növekedett. Ez a kis kernelméret (3x3) hatékonyan rögzíti a képekben lévő térbeli hierarchiát a receptív mező fokozatos növelésével. (Géron, 2019) A második fő jellemző a mélysége a modellnek, a VGGNet változatai például a VGG16 és a VGG19 amelyek a rétegek számát jelölik. Ez a mélység lehetővé teszi a vizuális adatok hierarchikusabb reprezentációját, ami elengedhetetlen a komplex képfelismerési feladatokban. (I. Goodfellow et al., 2016)

A modell több ponton is a CNN architektúrákban elterjedt megoldásokat használja. Először is több konvolúciós rétegek követően a modell max poolingot alkalmaz a trébeli dimenziók csökkentésére. (Szeliski, 2022)) Valamint a hálózat végén a VGGNet három teljesen összekapcsolt réteget alkalmaz és ez is tükrözi a hagyományos CNN tervezés eredetét. (I. Goodfellow et al., 2016)

A VGGNet 3x3 szűrő mérete precedenst teremtett a neurális hálózattok tervezésében, ezzel előnyben részesítve az egyszerűséget és mélységet az architektúrák komplexitásával szemben. A VGGNet és ehhez hasonló mély hálózattok könnyebben ragadnak meg bonyolult struktúrákat az adatokból. Ez azt jelenti, hogy hatékonyan tanulja meg a modell a jellemzők széles skáláját, az egyszerűtől az összetettig. (I. Goodfellow et al., 2016)

A VGGNet architektúrája kiváló jelöltnek bizonyult a transzfer tanulás (“tranfer learning”) szempontjából. (Géron, 2019) Az a képessége, hogy jól általánosítható különböző feladatokra, számos alkalmazásban előnyös választássá teszi a jellemzőkinyeréshez.
A VGGNet kortársaihoz, például az AlexNet-hez képest jelentősen mélyebb, ami hozzájárul a jobb teljesítményéhez a jellemzőtanulásban. A későbbi architektúrákhoz, például a ResNethez képest azonban a VGGNet kevésbé hatékony a számítási erőforrások tekintetében, ahogyan arra (I. Goodfellow et al., 2016) rámutatott. A kihagyott kapcsolatok bevezetése a ResNet-ben (He et al., 2015a) megoldja a VGGNet-ben látott hiányosságokat, mint például a nagyon mély hálózatokban jelentkező eltűnő gradiens probléma.
A sikerei ellenére viszont a VGGNet modellnek több kihívással is szembe kell néznie. Az első a számítási intenzitás. A modell mélysége és uniformitása jelentős számítási erőforrásokat igényel, ami kevésbé teszi megvalósíthatóvá a valós idejű alkalmazások számára. A második legfontosabb kihívás, hogy a modell hajlamos túlillesztésre a kisebb adathalmazok esetén, emiatt regularizálási és adatnövelési technikákat is használnunk kell ezek elkerülésére.
Összeségében fontos megjegyezni, hogy bár az újabb architektúrák hatékonyságban és teljesítményben talán felülmúlják a VGGNet-et, a VGGNet szerepe a mély CNN-ek megértésében és a képfelismerésben való alkalmazásukban betöltött szerepében továbbra is tagadhatatlan. Ahogy a mélytanulás tovább fejlődik, a VGGNet alapelvei kétségtelenül továbbra is meghatározzák a terület jövőbeli innovációit és a hálózat építés megoldásait.

A VGG modellnél jelentős teljesítménybeli eltérést tapasztaltam. Kettő, egy nem előre tanított és egy az ImageNet alapján előre tanított verziót teszteltem. A „pretrained” verzió az adatbővítéssel jelentős javulást mutatott az eddigi modellekhez képest mind top-1, mind top-5 pontosságban és nem utolsó sorban a validációs veszteség is jelentősen csökkent. Ez arra utal, hogy az adatbővítés kombinálva a transzfer tanulással („transfer learning”) nagymértékben javítja a modellek teljesítményét. A transzfer tanulás hatékonysága igazoltnak tűnik annak a fényében, hogy az eddigi összes tesztelt modell, amit nulláról tanítottunk alulmúlta ezt. VGGNet és az AlexNet összehasonlítása során azt látjuk, hogy az AlexNet modellek jelentősen profitálnak az adatbővítésből, mind a top-1, mind a top-5 pontosságban javulást, valamint alacsonyabb validációs veszteséget mutatva az adatbővítés nélküli AlexNet alapmodellhez képest. Azonban még a legjobban teljesítő AlexNet modell sem éri el a legjobb VGGNet modell pontossági arányát, ami azt jelzi, hogy a VGGNet további mélysége és összetettsége jobb lehetőségeket biztosít a jellemzők komplex képekből való tanulására.

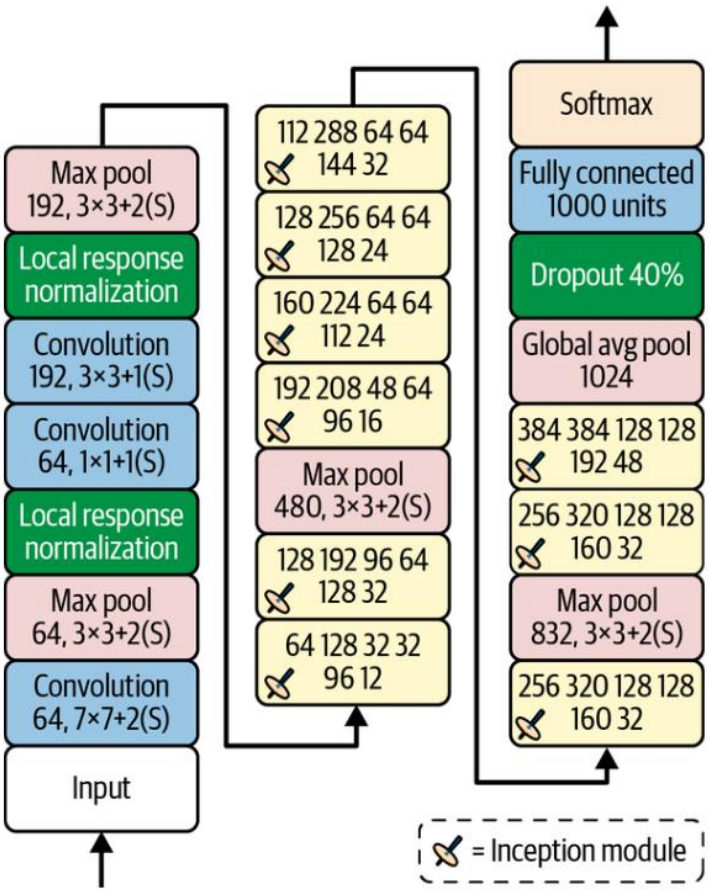
Összefoglalva, a VGGNet modellek, különösen az előzetesen betanítottak, a LeNet-5 és az AlexNet modellekhez képest kiváló teljesítményt nyújtanak a képosztályozási feladatokban. A transzfer tanulás és az adatok bővítése rendkívül hatékony módszernek tűnik a modell pontosságának és általánosító képeségének javítására. Az eddigi tesztjeim rávilágítanak a CNN-architektúrák folyamatos fejlődésére a kép felismerés és osztályozás területén.

## 5.5 GoogLeNet

A GoogLeNet, a deep learning és a számítógépes látás területén korszakalkotó architektúra. A modellt Christian Szegedy és társai fejlesztették ki, akik a Google Researchnél dolgoztak (Szegedy et al., 2014). Ez az úttörő architektúra jelentős lépés volt ezen a területen, megnyerte a 2014-es ImageNet (ILSVRC) versenyt és 7% alatti hibaarányt ért el. A GoogLeNet egyik, de a legfontosabb megkülönböztető jellemzője a kezdeti úgynevezett „inception” modul. Ez a modul egy párhuzamos rétegekből álló minihálózatot jelent, amely lehetővé teszi a hatékony paraméterhasználatot. A modell annak ellenére, hogy sokkal mélyebb, valójában tízszer kevesebb paraméterrel rendelkezik, mint elődje az AlexNet, amelyet már bemutattam. A paraméterek számát tekintve a 60 millió áll szemben a körülbelül 6 millióval. (Géron, 2019)
Az inception modulok különböző kernelméretű (1x1,3x3 és 5x5) konvolúciós rétegekből és egy max pooling rétegből állnak, ami lehetővé teszi a különböző méretű térbeli hierarchiák felismerését.
Az 1x1-es konvolluciós rétegek az inception modulon belül döntő szerepet játszanak. Ezek az 1x1 mivoltukból adódóan nem térbeli mintázattokat rögzítenek, hanem mélységi dimenziókra, csatornákon átívelő mintázatokra összpontosítanak. Azáltal, hogy kevesebb jellemzőtérképet állítanak elő, mint a bemenetük, szűk keresztmetszetű rétegként működnek, csökkentve a dimenzionalitást és a számítási költségeket, miközben felgyorsítják a képzést és fokozzák az általánosítást.
A GoogLeNet architektúrája különösen mély és összetett. Az első réteg egy nagy, 7x7-es kernelt használ, amely a kép magasságát és szélességét 4-gyel csökkenti, miközben az információ nagy részét megőrzi. Ezt követően egy LRN („local response normalization”) réteg biztosítja a különféle jellemzők megtanulását. Ezután két további konvolúciós réteg következik, ahol az első szűk keresztmetszetű rétegként működik. Az újabb LRN réteg után egy max pooling réteg tovább csökkenti a képméreteket. A modell magja egy kilenc egymásra épülő inception modul, amelyet a dimenziócsökkentést szolgáló max pooling rétegek váltanak fel. A hálózat ezután egy globális average pooling réteget alkalmaz, amely az egyes

jellemzőtérképek átlagát adja ki, elvetve a térbeli információkat, de megtartva az osztályozási feladatokhoz szükséges lényeges jellemzőket. Az utolsó rétegek között található egy dropout a regularizációhoz, egy teljesen összekapcsolt réteg 1000 egységgel (az adatkészlet 1000 osztályához igazítva), és egy softmax aktivációs függvény az osztályok valószínűségének megbecsüléséhez. A GoogLeNet eredetileg két segédosztályozót tartalmazott, amelyek a harmadik és a hatodik inception modulhoz kapcsolódtak. Ezek a segédosztályozók, amelyek az average pooling, a konvolúciós és teljesen összekapcsolt rétegekből álltak, az eltűnő gradiensek probléma mérséklését és a hálózat regularizálását célozták. Később azonban hatásuk csekélynek bizonyult emiatt eltávolították a modellből.

A GoogLeNet sikerét követően a Google kutatói több változatot javasoltak, köztük az Inception-v3 és az Inception-v4 változatot. Ezek a változatok némileg eltérő inception modulokat tartalmaztak, és még jobb teljesítményt értek el, bizonyítva az architektúra alkalmazkodóképességét és skálázhatóságát.




*ábra 24 GoogLeNet hálózat architektúrájának sémája (forrás: (Géron, 2019))*

Ahogyan az előző modelleknél, úgy a GoogLeNet-nél is több modellváltozatot teszteltem. A többi modellel összehasonlítva azt mutatja, hogy kiváló egyensúlyt teremt a komplexitás és hatékonyság között. Kevesebb paraméterrel, mint az AlexNet, a GoogLeNet még mindig versenyképes teljesítményt nyújt, jelentős top-1 és top-5 pontosságot ért el adatbővítés nélkül.

Ha megnézzük a két GoogLeNet-modell konkrét teljesítményét, észrevehető különbséget látunk a teljesítményben, ha adatbővítést nélkül vagy adatbővítéssel alkalmazzuk. Adatbővítés nélkül a GoogLeNet 50 epoch után 18,04%-os top-1 pontosságot és 44,28%-os top-5 pontosságot ér el, 5,1184-es validációs veszteséggel. Ezzel szemben az adatok bővítésével a modell 30 epoch után 11,74%-os top-1 pontosságot és 37,25%-os top-5 pontosságot ér el, 3,6701-es csökkent validációs veszteséggel. Ez azt sugallja, hogy bár az adatbővítés segít a túlillesztés csökkentésében - amit a veszteség is jelez -, nem mindig eredményez nagyobb pontosságot ugyanannyi epoch alatt. Ez annak tudható be, hogy a modellnek több epochra van szüksége ahhoz, hogy teljes mértékben kihasználja a bővített adatok általánosításra vonatkozó változékonyságát.

A GoogLeNet teljesítménye a VGGNettel szemben, különösen az előtanított VGGNet modell (adatbővítéssel), rávilágít a mélység és a számítási hatékonyság közötti kompromisszumokra. A VGGNet mélyebb rétegei és nagyszámú paramétere előnyhöz segíti a részletgazdag jellemzők tanulását, ami a jobb top-1 és top-5 pontosságban is megmutatkozik. A GoogLeNet azonban a paraméterek hatékonyabb felhasználása miatt működőképes alternatívát jelent, különösen olyan esetekben, amikor a számítási erőforrások problémát jelenthetnek.
Összefoglalva, a GoogLeNet modell megismerése és összevetése a LeNet-5-tel, az AlexNet-tel és a VGGNet-tel összehasonlítva alátámasztja az architekturális innovációk fontosságát a mélytanulás területén. Mindegyik modell hozzájárult ahhoz, hogy megértsük, hogyan lehet hatékonyan felépíteni a neurális hálózatokat képi feladatokhoz. Különösen a GoogLeNet inception moduljai nyitották meg az utat az összetettebb és számítási szempontból hatékonyabb architektúrák kifejlesztéséhez, amelyek továbbra is kitolják a képfelismerés és osztályozás terén lehetséges határokat.

## 5.6 ResNet

A ResNet (He et al., 2015a), vagyis „Residual Network” volt a következő mérföldkő a számítógépes látás területén. A ResNet modell megnyerte az ImageNet 2015-ös versenyét, egy évvel a GoogLeNet által aratott siker után, 3,6%-os hibaaránnyal. Ez nem csak a teljesítménye miatt volt kiemelkedő, hanem azért is mivel rendkívül mély volt, 152 réteget tartalmazott.

A ResNet alapkoncepciója és az újítása a kihagyásos kapcsolatokban, más néven rövidített kapcsolatokban rejlik. Ez a módszer lehetővé teszi, hogy egy réteg bemenete közvetlenül hozzáadódjon egy másik, a hálózatban tovább haladó réteg kimenetéhez. Ez a megközelítés segíti a hálózatot abban, hogy $h(x)$helyett $f(x) = h(x) - x$ függvényt modellezzen, amit reziduális tanulásnak nevezünk. (He et al., 2015a)

Azáltal, hogy hatékonyan lehetővé teszik a hálózat számára, hogy kezdetben az identitásfüggvényt modellezze, gyorsabb konvergenciát tesznek lehetővé, különösen akkor, ha a célfüggvény hasonló az identitásfüggvényhez. (He et al., 2015a)

A ResNet felépítését az egyszerűség és a mélység jellemzi. (Géron, 2019; He et al., 2015a) Hasonlóan kezdődik és végződik, mint a GoogLeNet, de az architektúra magja, a reziduális egységek (RU-k) mély egymásra épülése. (Géron, 2019)

A hálózat előrehaladtával a jellemzőtérképek száma periodikusan megduplázódik, miközben dimenzióik a felére csökkennek. (Géron, 2019) Ahhoz, hogy a modell alkalmazkodni tudjon ezekhez a változásokhoz, a bemenetek egy 1x1-es konvolúciós rétegen keresztül kerülnek kiigazításra 2 lépésközzel, hogy megfeleljenek a RU kimeneti dimenzióinak. (He et al., 2015a)

A ResNet-nek különböző változatai vannak, például ResNet-34, ResNet-50, ResNet-101 és ResNet-152, amelyek mindegyike a rétegek számában különböztethető meg. (He et al., 2015a)

A mélyebb változatok, mint például a ResNet-152, a jobb hatékonyság érdekében három konvolúciós réteggel rendelkező ún. szűk keresztmetszetet alkalmaznak. (He et al., 2015a)

A ResNet forradalmasította a mélytanulást azáltal, hogy megmutatta, hogy a nagyon mély hálózatok hatékonyan képezhetők. (He et al., 2015a) Az ILSVRC 2015-ös kihíváson elért sikere, hogy a top 5-ös hibaarányt 3,6% alá csökkentette, jelentős lépés volt a számítógépes látás területén. (Géron, 2019; He et al., 2015a) A ResNet koncepciója hatással volt a későbbi modellek, például a Google Inception-v4 modelljének kifejlesztésére. (Géron, 2019)

A ResNet mérföldkőnek számít a mélytanulásban, mivel bevezetett egy hatékony módszert a különösen mély neurális hálózatok tanítására. (He et al., 2015a) A kihagyásos kapcsolatainak, egyszerű, mégis mély struktúrájának és a különböző változatok kezelésére való képességének köszönhetően kulcsfontosságú modellé vált a számítógépes látás területén és azon kívül is. (Géron, 2019) A ResNet által lefektetett alapelvek továbbra is befolyásolják a neurális

hálózatok architektúrájának tervezésében a jelenlegi és a jövőbeli fejlesztéseket is. (Géron, 2019)

A modell tesztelésére a ResNet50 változatot használtam. A választás azért erre a modellre esett mivel a mélység és a hatékonyság közötti egyensúlyt próbáltam megtalálni. A ResNet50 az 50 rétegével egy stabil és kellően mély, robosztus architektúrát kínál a képosztályozási feladatokhoz, valamint számítási kapacitás szempontjából megvalósítható. Ahogyan az előzőekben itt is a TensorFlow és Keras könyvtárakat használtam. A Keras előre definiált modelleket kínál, emiatt nem szükséges a rétegeket nekünk felépíteni. A két modellt teszteltem, az első az nem előre tanított (nulláról tanuló) volt, a második viszont az ImageNet adathalmazon előre tanított. A transzfer tanulás választása azért volt itt főként kritikus, mivel drasztikusan csökkenti a képzési időt, amiatt mert a modell már eleve a jellemzők széles körét ismeri. Az előfeldolgozás során, a képeket 224 x 224 pixelesre méreteztem, valamint az adatbővítést is végrehajtottam. Ezek voltak a véletlenszerű tükrözés, forgatás, nagyítás.

A modellemben új technikákat is használtam, hogy javítsam a modell teljesítményét. Beépítettem úgynevezett „callback”-eket.
A „callback” hatékony eszköz, amelyet a képzési folyamat ellenőrzésére és irányítására használnak. A visszahívások lényegében olyan függvények, amelyek a képzési folyamat bizonyos szakaszaiban, például egy batch vagy epoch végén alkalmazhatók. Különböző célokra használhatók, például a modell mentésére a különböző szakaszokban, a tanulási sebesség beállítására vagy a képzési folyamat korai leállítására, ha bizonyos feltételek teljesülnek. A visszahívások alkalmazásának elsődleges oka a CNN-ekben a képzési folyamat hatékonyságának és eredményességének növelése.
A ModelCheckpointhoz (*Tf.Keras.Callbacks.ModelCheckpoint | TensorFlow v2.14.0*, n.d.) hasonló callbackek például lehetővé teszik a legjobb modell megőrzését a képzés során kézi beavatkozás nélkül (Géron, 2019). Az EarlyStopping (*Tf.Keras.Callbacks.EarlyStopping | TensorFlow v2.14.0*, n.d.) a túlillesztés megelőzésére szolgál - ez a mélytanulásban gyakori kihívás, amikor a modell a képzési adatokon kivételesen jól, de a nem látott adatokon rosszul teljesít (Géron, 2019). Ezt úgy éri el, hogy leállítja a képzést, amikor a modell teljesítménye egy validációs készleten egy meghatározott számú epoch alatt nem javul. Végül, az olyan visszahívások, mint a ReduceLROnPlateau (*Tf.Keras.Callbacks.ReduceLROnPlateau | TensorFlow v2.14.0*, n.d.) segítenek a tanulási folyamat finomhangolásában azáltal, hogy a

modell teljesítménye alapján módosítják a tanulási sebességet, ezáltal elősegítve a jobb konvergenciát és a modell teljesítményének általános javulását (Géron, 2019).
A callback-ek tehát nélkülözhetetlenek a mély neurális hálózatokban, mivel automatizálják a képzési folyamat kritikus aspektusait, lehetővé téve a modell hatékonyabb képzését, a validációs és tesztadatokon nyújtott jobb teljesítményt, valamint ahogy a modellünk egyre erőforrás igényesebb segít azt optimalizálni.
Továbbá a verziók tesztelése során beépítettem az alap ResNet modell után egy globális átlagos összevonó („global avergae pooling”) réteget, valamint egy sűrű („dense”) réteget is, mivel ahogy a tesztek során is kiderült nagyban befolyásolta a modell teljesítményét. A „global average pooling” réteg a jellemzőtérképek térbeli dimenzióinak csökkentésére szolgált, ezáltal csökkentve a modellben lévő paraméterek teljes számát (Lin et al., 2014). Ez a komplexitáscsökkentés nemcsak a túlillesztés megelőzésében segített, hanem a modell számítási hatékonyságát is növelte. A „global average pooling” réteget követően sűrű rétegeket alkalmaztam dropout regularizációval. A „dense” rétegek, mivel teljesen összekapcsoltak, biztosítani tudják az osztályozáshoz szükséges számítási teljesítményt, míg a „dropout” rétegek a neuronok egy részét véletlenszerűen deaktiválják a képzés során (I. Goodfellow et al., 2016; Srivastava et al., 2014). Ez a kiesési technika tovább segítette a túlillesztés megelőzését, mivel biztosította, hogy a modell ne támaszkodjon túlságosan egyetlen jellemzőre sem, elősegítve az általánosító tanulást, ami elengedhetetlen a robusztus teljesítményhez a változatos adatokon.

A különböző neurális hálózati modellekkel kapcsolatos korábbi tárgyalásokat átgondolva egyértelmű, hogy a ResNet bevezetése jelentős előrelépést jelent a mélytanulás és a számítógépes látás területén. A projektem során a ResNet50, ami egy előzetesen betanított modellt használt, lenyűgöző, 76%-os pontossággal rendelkezik a top-1 és 96%-os pontossággal a top-5 osztályozásnál. Ez a magas pontosság azt tükrözi, hogy a modell képes hatékonyan megragadni és általánosítani az ImageNet-adatkészlet komplex jellemzőit, valamint az új adatokon is képes folyatni a tanulást. Az olyan visszahívások, mint a ModelCheckpoint, EarlyStopping és ReduceLROnPlateau stratégiai integrálása a modell architektúrájával együtt jelentősen finomította a képzési folyamatot. Ezek a callbackek a modell optimális modellmegőrzését, a túlillesztés megelőzését és a tanulási sebesség hatékony beállításának megkönnyítésével fokozták a modell képzési hatékonyságát.

Összefoglalva, a ResNet egy kellően fejlett architektúra, ami képes magas pontosságot elérni a felépítéséből adódóan. De a modell teljesítménye még tovább fejleszthető az olyan technikai

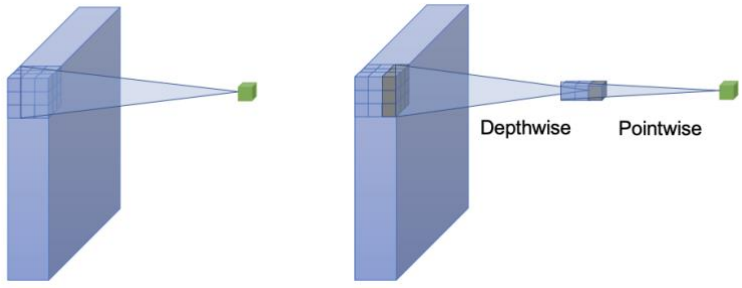
megoldások, mint a „callbackek”, egy-egy jól megválasztott réteg, valamint az adatbővítés felhasználásával. Ezek az esetek rámutatnak arra is, hogy a ResNet architektrúra eredendően egymagában is jó megoldás, viszont képesek vagyunk optimalizálni különleges adathalmazokra is, mint a Stanford Dogs dataset.

## 5.7 Xception (Extreme Inception)

Az Xception, az „Extreme Inception” rövidítése. Ezt a modellt (Chollet, 2017) kifejezetten képosztályozási feladatokra tervezte Francois Chollet a Keras (Chollet, 2015) megalkotója. A Xception a GoogLeNet architektúra módosítása, amely jelentősen felülmúlta az Inception V3-at a nagyméretű képosztályozási feladatokban, amikor 350 millió képet és 17.000 osztályt vizsgáltak (Chollet, 2017).

A modell azon a hipotézisen alapul, hogy a konvolúciós neurális hálózatok jellemzőtérképének a térbeli és a csatornák közti korrelációi teljesen szétválaszthatók. (Chollet, 2017) Ez a hipotézis az Inception architektúra alapjául szolgáló hipotézis kiterjesztett változata. (Szegedy et al., 2014)

Az architektúra a korábbi Inception modellekben központi szerepet játszó Inception modulokat mélységben szeparálható konvolúciós rétegekkel helyettesíti. A mélység szerint szeparálható konvolúció egy fejlett technika a neurális hálózatok tervezésében, amely a 2014 körüli bevezetése óta vált ismertté, mivel a TensorFlow-ban elérhető modul. (Abadi et al., 2016) Ez a módszer kulcsfontosságú az olyan modellek hatékonyságának és teljesítményének növelésében, mint az Xception, és kétlépcsős folyamatot foglal magában: a mélységi konvolúciót és a pontszerű konvolúciót.




*ábra 25 Mélységi és pontonkénti konvolúció szemléltetése (forrás: (Guo et al., 2019) )*

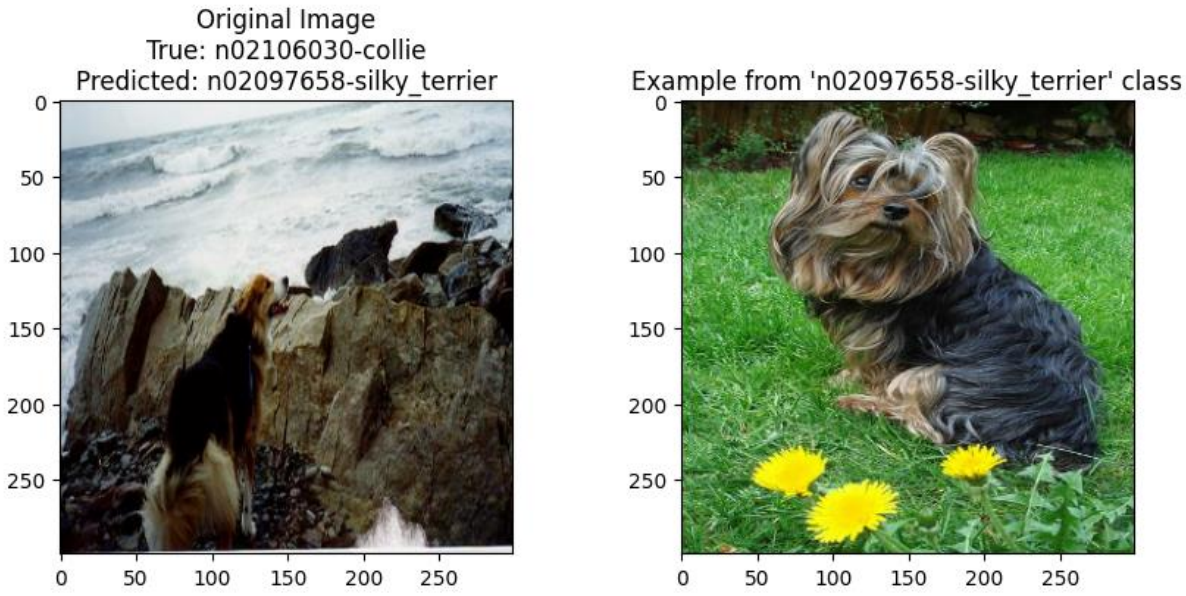
A mélységi konvolúció a hagyományos konvolúciós művelet módosított változata. A standard konvolúcióval ellentétben, amely egyetlen szűrőt alkalmaz az összes bemeneti csatornára, a mélységi („depthwise”) konvolúció minden egyes bemeneti csatornára egyedi szűrőt alkalmaz.

Például egy három csatornával - vörös, zöld és kék - rendelkező RGB-kép esetében a mélységi konvolúció három különböző szűrőt alkalmaz, amelyek mindegyike a saját csatornáján működik. Ez a megközelítés lehetővé teszi a modell számára, hogy a térbeli dimenziókat, például a szélességet és a magasságot, minden csatornán belül külön-külön képezze le. Így a mélység szerinti konvolúció az egyes csatornákból származó térbeli jellemzők kinyerésére összpontosít, anélkül, hogy az információkat összekeverné a csatornák között. (Chollet, 2017)
A mélység szerinti konvolúció után következik a pontszerű („pointwise") konvolúció, amely lényegében egy hagyományos konvolúció, de 1x1 szűrőt használ. A "pontszerű" elnevezés onnan ered, hogy ezek az 1x1-es konvolúciók a jellemzőtérkép minden egyes pontját vagy pixelét külön-külön dolgozzák fel. A pontonkénti konvolúció a mélységi konvolúció kimenetén hat, amely az egyes bemeneti csatornáknak megfelelő jellemzőtérképek halmazát tartalmazza. Szerepe az, hogy ezeket a jellemzőtérképeket az összes csatornára vonatkozó 1x1-es konvolúció alkalmazásával kombinálja. Ez a szakasz kulcsfontosságú, mivel lehetővé teszi a csatornák közötti információk keveredését, ezáltal a neurális hálózat megtanulja és integrálja a csatornákon átívelő mintákat és jellemzőket. (Abadi et al., 2016; Chollet, 2017)
Lényegében a mélység szerint szeparálható konvolúció a konvolúciós műveletek hatékonyabb és eredményesebb elvégzési módját jelenti. A konvolúciós folyamat térbeli (mélységi) és csatornákon átívelő (pontszerű) komponensekre történő szétválasztásával ezek a konvolúciók csökkentik a számítási komplexitást, miközben fenntartják, sőt gyakran fokozzák a hálózat azon képességét, hogy a bemeneti adatok fontos jellemzőit megragadja és integrálja. (Chollet, 2017; Géron, 2019)
Az Xception 36 konvolúciós rétegből áll, amelyek a jellemzőkinyerés alapját képezik. (Chollet, 2017) Ezek a rétegek 14 modulba vannak csoportosítva, amelyek az első és az utolsó modul kivételével lineáris reziduális kapcsolatokkal rendelkeznek. (Chollet, 2017) Az architektúra két hagyományos konvolúciós réteggel kezdődik, de aztán elsősorban szeparálható („separable") konvolúciókat használ, valamint max pooling rétegeket és végső rétegekként, egy globális átlag gyűjtő („global average pooling) réteget és egy sűrű („dense") kimeneti réteget. (Chollet, 2017)
Az architektúra egyszerűségé, hogy a mélységi szeparálható konvolucók és ehhez tartozó reziduális kapcsolatok egymásra épülnek. Emiatt sokkal könnyebb a modell definiálása, az olyan összetettebb architektúrákhoz képest, mint az Inception v2 vagy v3.
Összefoglalva, az Xception jelentős mérföldkőnek számít a konvolúciós neurális hálózatok fejlődésében, mivel egyesíti a GoogLeNet és a ResNet elveit, miközben innovatív, mélységben elválasztható konvolúciós rétegeket vezet be központi elemként. Ez az architektúra nemcsak a modelltervezést egyszerűsíti elődjeihez képest, hanem a hatékonyságot és a teljesítményt is

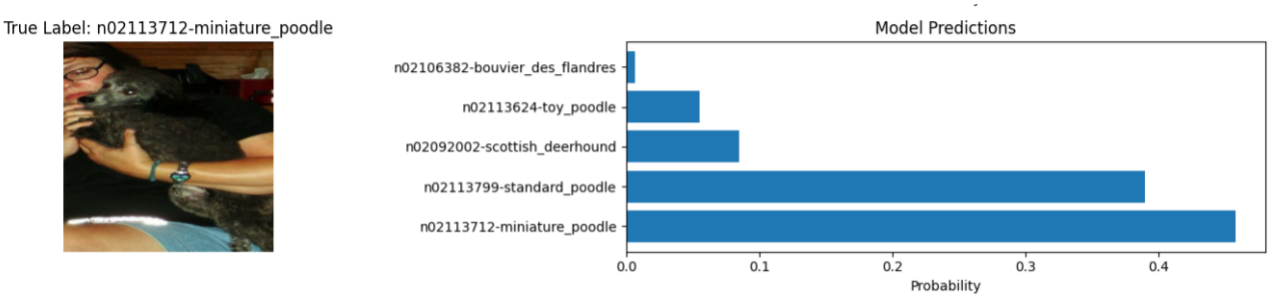
növeli, különösen a nagyméretű képosztályozási feladatok kezelésében. Ahogy (Géron, 2019) megjegyzi, a kevesebb paramétert és számítást használó szeparálható konvolúciós rétegek alkalmazása gyakran jobb teljesítményt eredményez, ami alátámasztja az Xception gyakorlati jelentőségét a számítógépes látás területén történő fejlődésben. Az egyszerűség, hatékonyság és eredményesség egyensúlya az Xceptiont kulcsfontosságú modellé teszi a mélytanulási architektúrák között.

A képosztályozáshoz használt neurális hálózati architektúrák vizsgálata során az Xception modell képességeit vizsgáltam, amely egy rendkívül fejlett architektúra. Az Xception modell teljesítménye figyelemre méltó volt, amely 89,7%-os top-1 pontosságot és 99,27%-os top-5 pontosságot ért el. Ez a fejlett felépítéséről és megoldásairól tanúskodik. Az eredmények jelentősen felülmúlják a korábban implementált ResNet50 modell eredményeit, amely 76%-os és 96%-os top-1 és top-5 pontosságot mutatott. Ez a számottevő mértékű javulás a pontosságban alátámasztja az Xception fejlett képességeit. Ennél a modellépítésnél is kihasználtam a Kerasban elérhető előre elkészített Xception modellt (Team Keras, n.d.). Ez a változat az ImageNet adathalmazból történő transfer learninggel kombinálva jelentősen csökkentette a képzési időt, és gazdag jellemzőkészletet biztosított a modell számára a tanuláshoz. A modell felépítésénél azokat a megoldásokat alkalmaztam, amiket ResNet50 modell esetében is. A ResNet50 modell megközelítéséhez hasonlóan az Xception modellt is „global average pooling” és „dense” rétegekkel, valamint kiesés-szabályozással láttam el.

Összességében elmondható, hogy az Xception architektúra kiemelkedően alkalmas a bonyolult képosztályozási kihívások megoldására, és felülmúlja a korábbi modellek képességeit. Viszont a modern megoldásoknál nem feltétlenül csak egy modell lehet mindig a legalkalmasabb. Azonban ezeknek a modelleknek a bemutatása jól példázza a konvolúciós neurális hálózatok (CNN) architektúráinak folyamatos fejlődését és finomodását, kiemelve a mélytanulásban rejlő lehetőségeket és alkalmazhatóságát a számítógépes látás és a képfelismerés, képosztályozás területén. Ezek a folyamatos újítások jelentős előre lepéseket jeleznek a területen, rámutatva a neurális hálózatok tervezésének innovációira.




*ábra 26 Helytelen klasszifikáció példája és a célosztály egy mintaképe (saját forrás)*




*ábra 27 Képklasszifikáció valószínűségi előrejelzései egy adott képre (saját forrás)*

```
Top Misclassification Pairs:
True Label: 'n02109961-eskimo_dog' - Predicted as: 'n02110185-siberian_husky' - Count: 19
True Label: 'n02106030-collie' - Predicted as: 'n02106166-border_collie' - Count: 9
True Label: 'n02093256-staffordshire_bullterrier' - Predicted as: 'n02093428-american_staffordshire_terrier' - Count: 8
True Label: 'n02093428-american_staffordshire_terrier' - Predicted as: 'n02093256-staffordshire_bullterrier' - Count: 8
True Label: 'n02097209-standard_schnauzer' - Predicted as: 'n02097047-miniature_schnauzer' - Count: 8
```

*ábra 28 Leggyakoribb téves besorolási párok a modellben (saját forrás)*

# 6. Összefoglalás

## 6.1 Az eredmények értelmezése, a kutatási kérdések megválaszolása

A dolgozatom eredményei rávilágítanak a képosztályozási feladatokhoz használt neurális hálózati architektúrák jelentős fejlődésére, a LeNet-5, AlexNet, VGGNet, GoogLeNet, ResNet és Xception modellek összehasonlításával, ezenfelül átfogó áttekintést nyújt a különböző

neurális hálózati megoldások fejlődéséről és hatékonyságáról. A munkám során ezeknek a modelleknek a mélyére ásva az innovációk egyértelmű pályát mutatnak a mélyebb és összetettebb architektúrák felé, amelyek mindegyike az elődök fejlesztéseire és hiányosságaira épül. Ezeknek a modelleknek a növekedő mélysége és összetettsége jelentős javulást eredményezett a képosztályozási pontosságban, ami bizonyítja az architektúra tervezésének kritikus jelentőségét a mélytanulás és a számítógépes látás területén. A fő kutatási kérdés a különböző neurális hálózati architektúrák hatékonysága volt, a nagyméretű képosztályozási feladatok megoldásában. Az eredmények egyértelműen azt mutatják, hogy az újabb architektúrák, mint a ResNet és az Xception, pontosság és hatékonyság tekintetében felülmúlják a régebbi modelleket, mint a LeNet-5 és az AlexNet.

## 6.2 Az egyes technikák elemzése, erősségei és gyengeségei

Az egyes modellek kritikai elemzése során nyilvánvalóvá válik, hogy a neurális hálózatok architektúrájában nincs mindenre egyformán alkalmas megoldás. Minden modell egyedi erősségeket és gyengeségeket mutat, amelyek alkalmassá teszik őket a képosztályozási feladatok különböző típusaira. A VGGNet például egyszerűsége és mélysége miatt nagy teljesítményű a képek összetett jellemzőinek megragadására, ugyanakkor számításigényes is. Másrészt a GoogLeNet és az Xception, az inception és a mélység szerint szeparálható konvolúciós rétegekkel egyensúlyt kínál a számítási hatékonyság és a teljesítmény között.
A ResNet jelentős előrelépést jelentett a reziduális kapcsolatok bevezetésével, amely az eltűnő gradiens problémájának kezelésével sokkal mélyebb hálózatok tanítását tette lehetővé. Az Xception erre épül tovább azáltal, hogy hatékonyan szétválasztja a térbeli és csatornaszintű korrelációkat a konvolúciós rétegeiben, ami nagyméretű képosztályozási feladatokban rendkívül hatékonynak bizonyult.

## 6.3 Az üzleti és az ipari vonatkozásai

Ezek az eredmények számos lehetőséget nyújtanak a képosztályozásra és -elemzésre támaszkodó vállalkozások és iparágak számára. A neurális hálózati architektúrákban elért fejlődés azt jelenti, hogy a bonyolultabb és változatosabb képi adathalmazok minden eddiginél pontosabban és hatékonyabban feldolgozhatók. Ez a fejlődés új lehetőségeket nyit meg olyan területeken, mint az orvosi képalkotás, az autonóm járművek vagy esetleg a balesetvédelem, ahol a pontos képosztályozás kulcsfontosságú.

A tanulmány rávilágít a neurális hálózatok architektúrájával kapcsolatos folyamatos kutatás és fejlesztés fontosságára annak érdekében, hogy a képfeldolgozási technológiákra támaszkodó iparágakban megőrizzék versenyelőnyüket a vállalatok.

## 6.4 Összegzés

Ez a szakdolgozat a különböző neurális hálózati architektúrák fejlődésének és hatékonyságának mélyreható vizsgálatát mutatta be a képosztályozás területén. A képfeldolgozás, valamint architektúrák bemutatása során betekintést engedett a tervezési elvekbe, működési mechanizmusokba és alkalmazási kontextusokba. A tanulmány legfontosabb megállapításai kiemelik a neurális hálózati modellek evolúcióját az egyszerű struktúráktól a komplex struktúrákig, rávilágítva a mélytanulásban elért technológiai előre lepésekre. Az úttörő és egyben legrégebbi LeNet-5-től a kifinomult Xception-ig minden egyes modell egyedülálló módon járult hozzá a képosztályozás területéhez. A tanulmány azt is kiemeli, hogy a neurális hálózatok mélysége és komplexitása mennyire fontos a képfelismerési lehetőségek javítása szempontjából.
Figyelemre méltó felfedezés az olyan architekturális újítások hatása a képosztályozás hatékonyságára és pontosságára, mint például a GoogLeNet-ben az inception rétegek és a ResNet-ben a reziduális kapcsolatok. Ezek az újítások megnyitották az utat az újabb modellek előtt, amelyek figyelemre méltó pontosságot érnek el, még az egyre összetettebb képi adathalmazok esetében is.
Ezen modellek összehasonlító elemzése árnyalt képet adott a modellek erősségeiről és korlátairól, és világos képet adott a különféle képosztályozási feladatokra való alkalmasságukról.

Az egyszerűbb modellektől a bonyolultabbak felé való fejlődés nem pusztán a rétegek vagy paraméterek növekedését jelenti, hanem annak mélyebb megértését, hogy hogyan optimalizálható a hálózati architektúra az adott feladatokhoz. Fontos hangsúlyozni, hogy a folyamatos kísérletezés és az eredmények vizsgálata létfontosságú ezen a területen.

Irodalomjegyzék

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., … Zheng, X. (2016). *TensorFlow: A system for large-scale machine learning* (arXiv:1605.08695). arXiv. http://arxiv.org/abs/1605.08695

Amidi, S. (n.d.). *CS 230—Convolutional Neural Networks Cheatsheet*. CS 230 - Deep Learning. Hozzáférés 12 November 2023, from https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks

Ayto, J., & Crofton, I. (2010). *Brewer's dictionary of modern phrase & fable* (2. ed., repr). Weidenfeld & Nicolson.

Ballard, D. H., Ballard, D. H., Brown, C. M., & Brown, C. M. (1982). *Computer vision*. Prentice-Hall.

Boureau, Y.-L., Ponce, J., & Lecun, Y. (2010). A Theoretical Analysis of Feature Pooling in Visual Recognition. *ICML 2010 - Proceedings, 27th International Conference on Machine Learning*, 111–118.

Boyd, S. P., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

*Building powerful image classification models using very little data*. (n.d.). Hozzáférés 11 December 2023, from https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html

Chollet, F. (2015). *Keras*. https://keras.io

Chollet, F. (2017). *Xception: Deep Learning with Depthwise Separable Convolutions* (arXiv:1610.02357). arXiv. http://arxiv.org/abs/1610.02357

Chollet, F. (2018). *Deep learning with Python*. Manning.

Coates, A., Ng, A., & Lee, H. (2011). An Analysis of Single-Layer Networks in Unsupervised Feature Learning. In G. Gordon, D. Dunson, & M. Dudík (Eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics* (Vol. 15, pp. 215–223). PMLR. https://proceedings.mlr.press/v15/coates11a.html

*CS231n Convolutional Neural Networks for Visual Recognition*. (n.d.-a). Hozzáférés 26 November 2023, from https://cs231n.github.io/classification/

*CS231n Convolutional Neural Networks for Visual Recognition*. (n.d.-b). Hozzáférés 29 October 2023, from https://cs231n.github.io/neural-networks-1/

*Dachshund Dog Stock Photo*. (n.d.). Shutterstock. Hozzáférés 11 December 2023, from https://www.shutterstock.com/image-photo/dachshund-sausage-dog-1-year-old-1335585956

Deng, J., Dong, W., Socher, R., Li, L.-J., Kai Li, & Li Fei-Fei. (2009). ImageNet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–255. https://doi.org/10.1109/CVPR.2009.5206848

*Dogphotographer.co.uk Kurt Pas*. (2023, January 12). iStock. https://www.istockphoto.com/hu/fot%C3%B3/teckel-vagy-tacsk%C3%B3-sz%C3%B3rakozik-gm1455075216-490534681

Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems* (Second edition). O'Reilly Media, Inc.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. The MIT Press.

Goodfellow, I. J., Warde-Farley, D., Mirza, M., Courville, A., & Bengio, Y. (2013). *Maxout Networks* (arXiv:1302.4389; Issue arXiv:1302.4389). arXiv. http://arxiv.org/abs/1302.4389

Grewal, D., Roggeveen, A. L., & Nordfält, J. (2017). The Future of Retailing. *Journal of Retailing*, *93*(1), Article 1. https://doi.org/10.1016/j.jretai.2016.12.008

Guo, Y., Li, Y., Feris, R., Wang, L., & Rosing, T. (2019). *Depthwise Convolution is All You Need for Learning Multiple Visual Domains* (arXiv:1902.00927). arXiv. http://arxiv.org/abs/1902.00927

He, K., Zhang, X., Ren, S., & Sun, J. (2015a). *Deep Residual Learning for Image Recognition* (arXiv:1512.03385). arXiv. http://arxiv.org/abs/1512.03385

He, K., Zhang, X., Ren, S., & Sun, J. (2015b). *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification* (arXiv:1502.01852; Issue arXiv:1502.01852). arXiv. http://arxiv.org/abs/1502.01852

Ioffe, S., & Szegedy, C. (2015). *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift* (arXiv:1502.03167). arXiv. http://arxiv.org/abs/1502.03167

Khosla, A., Jayadevaprakash, N., Yao, B., & Fei-Fei, L. (n.d.). *Stanford Dogs dataset*. Hozzáférés 21 November 2023, from http://vision.stanford.edu/aditya86/ImageNetDogs/

Krizhevsky, A. (2009). *Learning multiple layers of features from tiny images*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90. https://doi.org/10.1145/3065386

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998a). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998b). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791

Leeuwen, J. van (Ed.). (1998). *Algorithms and complexity* (1. MIT paperback ed., 2. print). Elsevier [u.a.].

Lin, M., Chen, Q., & Yan, S. (2014). *Network In Network* (arXiv:1312.4400). arXiv. http://arxiv.org/abs/1312.4400

Louridas, P. (2020). *Algorithms*. The MIT Press.

Minsky, M., & Papert, S. A. (2017). *Perceptrons: An Introduction to Computational Geometry*. The MIT Press. https://doi.org/10.7551/mitpress/11301.001.0001

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). *PyTorch: An Imperative Style, High-Performance Deep Learning Library* (arXiv:1912.01703). arXiv. http://arxiv.org/abs/1912.01703

Razavian, A. S., Azizpour, H., Sullivan, J., & Carlsson, S. (2014). *CNN Features off-the-shelf: An Astounding Baseline for Recognition* (arXiv:1403.6382). arXiv. http://arxiv.org/abs/1403.6382

Simonyan, K., & Zisserman, A. (2015). *Very Deep Convolutional Networks for Large-Scale Image Recognition* (arXiv:1409.1556). arXiv. http://arxiv.org/abs/1409.1556

*Sleeping dachshund*. (n.d.). Pinterest. Hozzáférés 11 December 2023, from https://www.pinterest.com/pin/sleeping-beauty--53269208082423091/

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.*, *15*(1), 1929–1958.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2014). *Going Deeper with Convolutions* (arXiv:1409.4842). arXiv. http://arxiv.org/abs/1409.4842

Szeliski, R. (2022). *Computer vision: Algorithms and applications* (Second edition). Springer.

Team Keras. (n.d.). *Keras documentation: Xception*. Hozzáférés 7 December 2023, from https://keras.io/api/applications/xception/

*Tf.keras.callbacks.EarlyStopping | TensorFlow v2.14.0*. (n.d.). TensorFlow. Hozzáférés 6 December 2023, from https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/EarlyStopping

*Tf.keras.callbacks.ModelCheckpoint | TensorFlow v2.14.0*. (n.d.). TensorFlow. Hozzáférés 6 December 2023, from https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/ModelCheckpoint

*Tf.keras.callbacks.ReduceLROnPlateau | TensorFlow v2.14.0*. (n.d.). TensorFlow. Hozzáférés 6 December 2023, from https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/ReduceLROnPlateau

*What are Convolutional Neural Networks? | IBM*. (n.d.). Hozzáférés 4 December 2023, from https://www.ibm.com/topics/convolutional-neural-networks

*Wire Haired Dachshunds*. (2021, January 7). https://formydachshund.com/do-long-smooth-and-wire-haired-dachshunds-have-different-personalities/

Yu, F., & Koltun, V. (2016). *Multi-Scale Context Aggregation by Dilated Convolutions* (arXiv:1511.07122). arXiv. http://arxiv.org/abs/1511.07122

# Ábrák jegyzéke