Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs Rendszerek Tanszék

FedMOD: Keretrendszer Federatív Multitaszk Objektum Detekció problémájára

TDK DOLGOZAT

Készítette
Kádár Attila

Konzulens
Hadházi Dániel

2022. november 1.

Tartalomjegyzék

Kivonat

Napjainkban az objektum detekciós algoritmusok használata egyre elterjedtebbé válik úgy vállalati, mint végfelhasználói környezetben. Alkalmazási területei közé tartoznak többek között az önvezető rendszerek, automatizált CCTV felügyelet, automatizált minőségellenőrzési folyamatok, tűzveszély monitorozás, orvosi döntéstámogatás. Sok esetben az ilyen jellegű szolgáltatások teljesítményének növelésében ma már a lokálisan rendelkezésre álló, felcímkézett adathalmaz mennyisége jelenti a szűk keresztmetszetet. Mivel a szigorú adatvédelmi korlátozások miatt - főként nagyvállalati környezetben, vagy például egészségügyben GDPR elvek miatt - ezen tanító adatok nem oszthatók meg és nehezen bővíthetők, így a teljesítmény növelése az egymással korreláló taszkokat tanuló machine learning alapú megoldások közötti kollaboráció során valósulhat meg. Különböző privát, lokális adathalmazokkal rendelkező kliensek kollaborációjára alkalmas a klasszikus federált tanulás aparátusa, ennek viszont fő korlátja, hogy az esetek többségében csak egy globális modell kollaboratív betanítása lehetséges, nem pedig több, korreláló taszkot tanuló modell teljesítményének növelése. Továbbá a létező federált multitaszk eljárások privacy-t sértő módon modell paraméterek vagy lokális adatminták megosztása által valósítják meg a kollaborációt, nem beszélve arról, hogy a publikált eljárások nagy többsége nem is alkalmazható hatékonyan olyan komplex, deep learning modellek esetében, melyek a képi objektum detekció feladatát hivatottak megoldani.

Ebben a dolgozatban javasolt FedMOD eljárás képes felülkerekedni az előbbiekben említett, szakirodalomban máig megoldatlan problémákon. Eljárásunk képes megbecsülni a résztvevő kliensek által tanult taszkok közötti kereszthasznosságokat. Ezen információ alapján valósít meg hatékony tudás disztillációt, melyhez publikusan rendelkezésre álló, címkézetlen adatmintákat használ fel. A FedMOD adatok védelmét megőrző, elosztott tanulást valósít meg, melynek során a privát adatok és a lokális modell paraméterek sem kerülnek megosztásra a kliensek között. Empirikus vizsgálatok alapján a FedMOD eljárás hatása ekvivalens a kollaboráló kliensek privát adathalmazainak bővítésével (melynek mértéke akár 50% feletti is lehet) azaz ennek mértékével emeli az adathalmaz értékét a kollaboráció. A javasolt FedMOD eljárást átfogóan kiértékeltük és elemeztük, továbbá kidolgoztuk a FedAvg algoritmusnak egy, a cél problémához illeszkedő adaptációját, mely teljesítményével összevetettük a FedMOD sémát is.

Abstract

Nowadays, the use of object detection algorithms is becoming increasingly common both in industry and among end-users. Applications include self-driving systems, automated video surveillance and alarm systems, automated quality control processes, fire risk monitoring, medical decision support, etc. In many cases, the bottleneck to increasing the performance of these services is the amount of locally available, labelled data. Due to strict privacy restrictions, this learning data is not shareable (especially in enterprise environments or for example in healthcare due to GDPR principles) and difficult to scale up, therefore performance improvement can only be achieved through collaboration between machine learning algorithms learning correlated tasks. The classical federated learning paradigm is suitable for the collaboration of clients with different private, local datasets, but its main limitation is that in most cases it can only train one global model collaboratively, it is not able to increase the performance of distributed models learning correlated tasks. Furthermore, existing federated multitask methods implement collaboration through sharing model parameters or private data samples in a privacy-violating manner, not to mention that the vast majority of published methods are not even efficiently applicable to complex deep learning models that are intended to solve the task of image object detection.

The proposed FedMOD method is able to overcome the aforementioned limitations, which are still unsolved based on literature. Our procedure is able to estimate the cross-utilities between tasks learned by the participating clients. Based on this information, it implements efficient knowledge distillation using publicly available unlabeled data samples. FedMOD realises privacy-preserving distributed learning, where private data and local model parameters are not shared with any other participant. Empirical studies have shown that the effect of the FedMOD procedure is equivalent to an increase in the private datasets of the collaborating clients (up to 50% or more), i.e., the value of the private dataset is increased thanks to the collaboration. We have comprehensively evaluated and analyzed the proposed procedure. Additionally we proposed and implemented an adaptation of the popular FedAvg algorithm to the target problem, and compared it with the FedMOD scheme.

1. fejezet

Bevezetés

Napjainkban az intelligens algoritmusok által működtetett szolgáltatások használata egyre elterjedtebbé válik úgy az iparban, mind a végfelhasználók körében. Ezt igazolja az intelligens, viselhető eszközök fokozott használata, az autonóm funkciókkal, például vezetéstámogató, önparkoló rendszerekkel felszerelt járművek megjelenése vagy akár a nagyobb vállalatokon belül alkalmazott, gépi tanuláson alapuló szolgáltatások [1]. Sok esetben az ilyen jellegű szolgáltatások teljesítményének növelésében ma már a lokálisan rendelkezésre álló, felcímkézett adathalmaz mennyisége jelenti a szűk keresztmetszetet. Mivel a szigorú adatvédelmi korlátozások miatt - főként nagyvállalati környezetben - ezen tanító adatok nem oszthatók meg, így a teljesítmény növelése az egymással korreláló taszkokat tanuló machine learning algoritmusok közötti kollaboráció során valósulhat meg úgy a végfelhasználók, mint a nem konkurens nagyvállalatok körében. Erre példa a telekommunikációs hálózatok operátorai és a szolgáltatók közötti együttműködés, vagy önvezető rendszerek (ADAS) kollaborációja [41].

Az autonóm vezetés egyik legfontosabb feladata a kamerák és LIDAR-ok [26] által rögzített közlekedési felvételeken az objektumok detektálása. ADAS rendszereket kínáló vállalatok a demo adatokon kívül a termékeiket használó járművektől gyűjtenek adatokat, azonban a fogyasztók nem biztos, hogy beleegyeznek vezetési adataik (pl. az általuk bejárt helyekről vagy lakóhelyükről készült fényképek) megosztásába. Ilyen jellegű problémákra kínál megoldást a federált tanulás (FL) [33] aparátusa.

A szakirodalomban már publikált federált képi objektum detekciós algoritmusok [31] többsége viszont egy globális modellt tanít elosztottan, és figyelmen kívül hagyja, hogy a kollaborációban résztvevő kliensek esetlegesen eltérő eloszlású tanító adathalmazzal (non-iid) rendelkezhetnek, vagy korreláló, de nem teljes mértékben megegyező feladatokat (taszkokat) tanulhatnak lokálisan. Ezen probléma megoldása már a federált multitaszk [13] eljárásoknál is megtalálható ötletek alkalmazását követelik, lehetővé téve a kliensek számára több, kisebb-nagyobb mértékben korreláló taszkot tanuló lokális modell betanítását.

Adatvédelem szempontjából fontos, hogy a kollaboráció alapjául ne a lokális, privát adathalmazaikból kinyert információ szolgáljon, hanem egy globálisan elérhető, publikus adathalmaz. A félig ellenőrzött tanulás metodológiája lehetőséget biztosít, hogy a kliensek közötti együttműködéshez és a klienspárok közötti kereszthasznosságok feltérképezéséhez felhasználhatóak legyenek a publikusan, nagy mennyiségben, könnyen elérhető címkézetlen minták.

Legjobb tudomásunk szerint a számos gyakorlati alkalmazás ellenére (például elosztott CCTV rendszerek, kollaboráló ADAS rendszerek, tűzveszély monitorozása, tüdő szegmentáció [3], multi-organ medical imaging [14]) szakirodalomban jelenleg nincs publikálva olyan elosztott objektum detekciós algoritmus, mely az előbb említett három terület (federált tanulás, multitaszk tanulás, félig ellenőrzött tanulás) metszetében helyezkedik el. Ezen dolgozat eredménye a javasolt FedMOD eljárás, mely a federált, multitaszk és félig ellenőrzött tanítás előnyeit kovácsolja össze egy új, komplex eljárás-blokk formájában. Eljárásunk több (akár eltérő, ám korreláló objektum detekciós feladatot tanuló) lokális modell között valósít meg hatékony tudás disztillációt a modell reprezentációk transzformálása és regularizálása által, melyhez publikusan rendelkezésre álló, címkézetlen adatmintákat használ fel. A FedMOD adatvédelmet megörző elosztott tanulást valósít meg, melynek során a privát adatok és a lokális modell paraméterek sem kerülnek megosztásra a kliensek között. Megjegyezzük, hogy eljárásunk főként kevés, ám megbízható és nagy adatmennyiséggel rendelkező kliens kollaborációjára biztosít megoldást, azaz tipikusan nagyvállalatok vagy több végfelhasználó által közösen használt nagyobb modellek (például több, régionálisan definiált és tanított ADAS rendszer) közötti együttműködést valósítja meg.

A dolgozat felépítése a következő: a második fejezetben bemutatom a szakirodalomban már publikált federált, multitaszk, félig ellenőrzött tanulási eljárásokat, kiemelve előnyeiket, illetve hiányosságaikat, melyet a FedMOD hivatott pótolni. A harmadik fejezetben az objektum detekciós feladat megoldására gyakran használt architektúrák, illetve a felhasznált Pascal VOC és COCO adathalmazok bemutatása, valamint ezeknek elosztott környezetbe történő definiálása kerül kifejtésre. A negyedik fejezetben a konzulensemmel közösen fejlesztett FedMOD eljárást és ennek komponenseit mutatom be részletesen, ezt követi az ötödik fejezetben az eljárás kiértékelése és összehasonlítása a nem kooperatív tanulással és az ismert FedAvg eljárásnak egy szintén általunk javasolt multitaszk adaptációjával. Végül pedig a levont konklúziókat a hatodik fejezetben ismertetem.

2. fejezet

Kapcsolódó irodalom

Ebben a fejezetben először bemutatom a szakirodalomban publikált federált, multitaszk és félig ellenőrzött témakörökbe tartozó, népszerű eljárásokat, melyek a vizsgált elosztott képi objektum detekciós probléma megoldása szempontjából relevánsak lehetnek. Kiemelem ezen eljárások előnyeit, illetve megindoklom a szakirodalom alapján a FedMOD eljárás szükségességét. A fejezet második felében a létező objektum detekciós megoldások csoportjainak rövid bemutatása következik.

2.1. Federált tanulás

Az egyik legnépszerűbb és a gyakorlatban is előszeretettel alkalmazott federált tanulást megvalósító eljárás a FedAvg [33]. Ez a megközelítés olyan tipikus elosztott tanulási problémákra nyújt megoldást, ahol a cél egyetlen globális modell federált tanítása a résztvevő kliensek lokális (privát) adathalmazain. Minden tanulási körben az összes kliens először az aktuális globális modell egy példányát optimalizálja lokálisan, a saját adathalmazán, majd a frissített paraméterek átlagolásával a koordináló szerver meghatározza a frissített globális modellt. A FedAvg főbb lépéseit az 1. algoritmus foglalja össze.

1. Algorithm Federated Averaging (FedAvg)

Szerver hajtja végre:

x_0 globális modell inicializálása
for t = 1, 2, ..., T **do**
    $S_t \leftarrow$ (random kiválasztott M kliens halmaza)
    for $\forall i \in S_t$ párhuzamosan **do**
        $x_{t+1}^i \leftarrow ClientUpdate(i, x_t)$
    end for
    $x_{t+1} \leftarrow \sum_{k=1}^{M} \frac{1}{M} x_{t+1}^k$
end for

ClientUpdate(i, x)**:**

for j = 1, ..., K **do**
    $x \leftarrow x - \eta \nabla_x f(x; z)$, ahol $z \sim P_i$
end for
return x

A FedAvg általánosításának tekinthető a FedProx [25], mely képes non-iid lokális adathalmazok hatékony kezelésére is. Ezek az algoritmusok számos cross-device [21] probléma esetén igen hasznosnak bizonyultak, viszont nem használhatóak olyan kollaboráció esetében, ahol a lokális adathalmazok eltérő lokális modellek tanítását követelik. Egy lehetséges megoldást kínál az ilyen jellegű problémákra a FedMD [24] módszer. Az eljárás során a kliensek először lokális modelljeiket konvergenciáig tanítják a privát adathalmazaikon, majd tudás transzfer [18] valósul meg több tanulási körön keresztül. A kollaboráció során egy minden kliens számára publikus, globális adathalmazon minden kliens kiszámolja közvetlenül a kimeneti réteg előtt létrejövő aktivációkat, és az eredményt minden egyes tanulási körben elküldi a koordináló szervernek. Az aktivációk átlagát (más néven konszenzust) a kliensek további finomhangolásra használják lokális tanítás során. A FedMD főbb lépéseit a 2. algoritmus foglalja össze. Az eljárás lehetővé teszi több lokális modell kollaboratív betanítását, mely non-iid adathalmazok esetén hatékony lehet, viszont nem használható olyan federált multitaszk problémák megoldására, ahol a résztvevő kliensek taszkjai közötti hasonlóságok varianciája nagy, vagy akár az egyes kliensek eltérő számú taszkot tanulnak.

2. Algorithm FedMD

Bemenet: D_0 publikus adathalmaz, D_k privát adathalmazok, $f_k,\ k = 1, ..m$ függetlenül tervezett modellek

Kimenet: f_k betanított modellek

1: **Előtanítás:** Minden résztvevő tanítja konvergenciáig saját f_k modelljét D_0-n és saját D_k adathalmazán
2: **for** t = 1, 2, ..., P **do**
3: **Kommunikáció:** A kliensek meghatározzák $f_k(D_0)$ kimenet előtti aktiváció értékeket, és elküldik a szervernek
4: **Aggregálás:** A szerver kiszámolja: $\overline{f}(x_i^0) = \frac{1}{m}\sum_k f_k(x_i^0),\ \forall x_i^0 \in D_0$ értékét.
5: **Megosztás:** Minden kliens letölti az új $\overline{f}(x_i^0)$ "konszenzust"
6: **Globális tanítás:** Minden f_k tanítása úgy, hogy közelítse $\overline{f}$-et D_0 adathalmazon.
7: **Lokális tanítás:** Minden f_k tanítása a privát D_k adathalmazon egy pár epochig.
8: **end for**

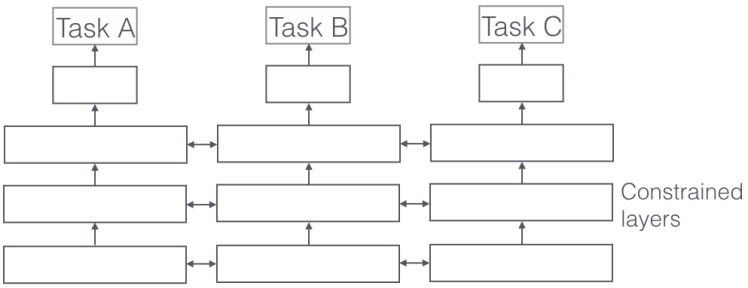
2.2. Elosztott multitaszk tanulás

Olyan elosztott tanulási problémák megoldására, amelyekben a résztvevő kliensek által lokálisan tanult taszkok nem megegyezőek (viszont köztük korreláció van), a klasszikus federált tanuló algoritmusok önmagukban nem elegendőek (a multitaszk tanulási módszerek felhasználása is szükséges). Mivel a FedMOD célja képi objektum detekció, ezért jelen alfejezetben a gépi látás témakörében előszeretettel alkalmazott multitaszk módszerekre fókuszálok.

2.2.1. Multitaszk tanulás

A multitaszk machine learning (MTL) módszereknél használt architektúrákat két fő komponensre bonthatjuk: jellemző kinyerő (feature extractor), illetve taszk specifikus rétegek halmaza. Ezen felosztás úgy történik, hogy az információ megosztása javítsa az egyes taszkokat megoldó modellek általánosító képességét és közben minimalizálja a negatív tudás transzfert a taszkok között. A [9] tanulmány több, erre az elvre épülő megközelítést mutat be, melyek közül az első az irodalomban shared trunk vagy hard parameter sharing néven ismert. A módszer egyetlen, közös jellemző kinyerő komponenst használ az összes taszk megoldására. Erre a komponensre épülnek a különböző, taszk specifikus rétegek. A shared trunk megközelítés (hasonlóan az említett tanulmányban bemutatott Predictive Distillation vagy Task Routing eljárások) főként olyan feladatokra lett kifejlesztve, ahol ugyanazon bemeneti adatok kerülnek felhasználásra a különböző taszkok megoldására és a közös, jellemző kinyerő komponens betanítására. Ez alapján a módszer önmagában nem alkalmazható olyan elosztott tanuló környezetekben, ahol a különböző taszkokat megoldó kliensek lokális adathalmaza eltér, valamint a lokális modell paraméterek és tanító minták megosztását adatvédelmi elvek gátolják.

Ellentétben a shared trunk megközelítéssel, a [9] tanulmányban bemutatott Cross-talk (soft parameter sharing) konstrukció taszkonként teljesen külön architektúrát javasol, melyek jellemző kinyerő komponensei között infromáció áramlás (cross-talk) megy végbe. A 2.1. ábrán egy példa látható Cross-talk architektúrára, a [35] cikk pedig egy lehetséges gyakorlati implementációját mutatja be. Látható, hogy ez a konstrukció átvihető elosztott környezetbe is, a 4. fejezetben részletesen bemutatott FedMOD eljárás is tartalmaz a jellemző kinyerő komponensek közötti kommunikációt.



2.1. ábra. Cross-talk konstrukció [45]

További elosztott környezetben alkalmazható multitaszk eljárásokat a [48] tanulmány foglal össze. Az ott leírt Feature Learning megközelítések a következő hipotézisen alapulnak: ha a modellek megosztják egymás között a belső reprezentációjukat, és együtt dolgoznak egy közös reprezentáció kialakításán, amely az összes taszkra vonatkozó információt hordozza magában, akkor ez potenciálisan javíthatja az egyes modellek általánosítási képességét a lokális feladataikat tekintve (az adathalmaz pontosabb modellezésével). A federált tanulási környezetben ezt az elképzelést a korábban említett FedMD algoritmus valósítja meg. A Feature Learning impliciten feltételezi, hogy minden taszk hasonló egymáshoz, ami gyakran nem így van, különösen az elosztott rendszerekben. A Feature Leraning megközelítésekhez hasonlóan több MTL eljárás is arra épít, hogy a taszkok közötti hasonlósági információ

rendelkezésre áll, mint priori ismeret, ami a gyakorlati problémák túlnyomó többségénél nem igaz. Ezzel szemben a Task Relation Learning megközelítések (pl. [6]) a feladatok közötti hasonlóságok feltérképezésére törekszenek, és ezt az információt használják fel a hatékony információ áramláshoz. Az [48] tanulmányban áttekintett összes módszer a hozzájuk tartozó adathalmazok alapján becsüli az egyes taszkok kereszthasonlóságát. Ez azonban adatvédelmi okokból nem mindig kivitelezhető, így az ilyen jellegű megközelítések a leírt formájukban nem is alkalmazhatóak elosztott környezetben.

2.2.2. Federált multitaszk tanulás

A Federált multitaszk tanulás területe az elmúlt néhány évben vált népszerűvé. A szakirodalomban több publikáció is született [13], [29], ezek közül kiemelném a "state-of-the-art"-nak számító MOCHA eljárást. A MOCHA algoritmus úgy optimalizálja az elosztott környezetben tanuló modelleket, hogy közben becsüli a modellek közötti kereszthasznosságokat is, ezáltal biztosítva a hatékony kollaboratív tanulást. Az eljárás korlátai a következők:

- A módszer $h(w_i^T X_i)$ alakú modelleken alkalmazható, $w_i \in R^d$ az i. modell paramétervektora, X_i a hozzá tartozó tanító adathalmaz, $h()$ pedig egy nemlinearitás. A képi objektum detekció megoldására viszont ennél nagyságrendekkel komplexebb, Deep Learning architektúrák használata szükséges, melyekhez a publikált MOCHA eljárás adatptációja nem triviális és ismereteink szerint szakirodalomban sem publikálták még. A szerzők az eljárást kizárólag szenzor adatokon értékelték ki.

- A taszkok és modellek közötti hasonlóságok feltérképezésére a MOCHA a modell paramétereket használja, viszont ezen paraméterek számos gyakorlati alkalmazás esetén (főként nagyvállalatok közötti kollaboráció során) nem megoszthatóak privacy okokból kifolyólag. Továbbá az egymáshoz hasonló paraméterű modellek bonyolultabb (sok nemlineáris réteget tartalmazó) architektúrákban már nem biztos, hogy hasonló leképezéseket is valósítanak meg.

A [3] és [37] publikációk mutatnak néhány példát a federált multitaszk tanulásnak az orvosi képdiagnosztikában történő alkalmazására. Ezek mindegyike egy globális modell betanítására törekszik, mely ellehetetlenítheti a hatékony együttműködést olyan esetekben, amikor a kliensek álal tanult taszkok kereszthasznossága, vagy akár a száma is eltér. Legjobb tudomásom szerint a szakirodalomban még nem elérhető olyan publikált federált multitaszk tanulási eljárás, mely képes több, változó mértékben korreláló taszkot tanuló kliens között szigorú adatvédelmi elvek mellett is hatékonyan kollaborálni olyan komplex problémák megoldása során is, mint a képi objektumdetekció.

2.3. Félig ellenőrzött federatív tanító algoritmusok

Elosztott multitaszk környezetben a kollaboráció hatékonyságát legfőképp a kliensek közötti tudás transzfer hatásfoka határozza meg. Ezen tudás transzferhez szükséges,

hogy fel tudjuk mérni a résztvevő kliens párok közötti kereszthasznosságokat, melyekkel arányos mértékben valósul meg köztük a kollaboráció. A kereszthasznosságok becslésére a szakirodalomban publikált megoldások többnyire a tanító adatok vagy a modell paraméterek megosztására épülnek [48]. Számos elosztott tanulásra épülő gyakorlati probléma esetén viszont ezek megosztása adatvédelmi okokból / üzleti megfontolásokból nem lehetséges. A publikusan nagy mennyiségben, könnyen hozzáférhető címkézetlen adatok viszont hatékonyan felhasználhatóak az adatok titkosságát is megőrző tudás transzfer megvalósítására, ezáltal teret adva a félig ellenőrzött, federatív, multitaszk tanulási eljárásoknak. Az előbbi motiváció és számos gyakorlati alkalmazás ellenére a szakirodalomban legjobb tudomásom szerint még nem található ezen három terület nyújtotta lehetőségeket kihasználó megoldás, mely alkalmazható olyan komplex problémákon is, mint a képi objektumdetekció.

Egy példa félig ellenőrzött federált eljárása a FedSem [4], melynek szerzői pseudo-labeling technika alkalmazását javasolják a lokális adathalmazok méretének növelése érdekében a kliens oldalon, így egy robusztusabb globális modell létrehozását elérve. Egy másik megközelítés [32] lokálisan optimalizálja a címkézett és a címkézetlen adatokból számított közös veszteségfüggvényt a jól ismert Mean Teacher [44] SSL technika segítségével. A FedSemhez hasonlóan ez a módszer is aggregálja a tanult paramétereket kliensek fölött egyetlen betanított globális modell kialakítása érdekében, mely az előzőek alapján nem hatékony multitaszk környezetben. Végül kiemelném a Distillation-based DS-FL módszert [19], amely a FedMD-hez hasonlóan globális logitokat számol ki a kliensek által a címkézetlen adatokon meghatározott logitok aggregálásával. A szerzők egy új logit-aggregációs módszert javasolnak, amelyet Entropy Reduction Aggregation (ERA) eljárásnak neveznek. A DS-FL megközelítés feltételezi, hogy a kliensek azonos taszkokat tanulnak a nem feltétlenül i.i.d. eloszlású adathalmazokon. Ez a feltételezés sok gyakorlati probléma esetén nem adekvát.

A javasolt FedMOD eljárás a címkézetlen adatok felhasználásának módjában különbözik az előbbi megközelítésektől: egy publikus, címkézetlen adathalmazt használunk a kliensek közötti kereszthasznosságok feltérképezésére, majd az ez alapján történő tudás transzfer megvalósítására. Továbbá a FedMOD még azt sem követeli meg, hogy a különböző kliensek által tanult feladatok azonos bemeneten azonos kimenetet produkáljanak. Ez kifejezetten hasznos a vizsgált elosztott, multitaszk objektum detekciós problémát tekintve, ahol akár ugyanazon a bemeneti képen két különböző kliens eltérő objektumok felismerését tanulhatja.

2.4. Képi objektum detekció

Az objektum detekciós probléma két alfeladatból tevődik össze: a bemeneti képen levő objektumok lokalizálásából és a lokalizált objektumok osztályokba sorolásából. A lokalizálás az objektumot a lehető legpontosabban határoló téglalap (bounding box) meghatározásával történik (például bal felső és jobb alsó csúcspontjának koordinátáinak felírásával). Tehát egy adott bemeti képre a modell kimenetként egy vagy több bounding box-ot, illetve az ezekhez tartozó osztálycímkéket adja vissza. A betanított modellek teljesítményét a legjobban illeszkedő bounding box-okhoz tartozó predikciók átlagos pontosságából és precizitásából határozzuk meg (lásd 3.1.3. alfejezet)

A létező képi objektumdetekciós eljárásokat két nagy csoportba sorolhatjuk ezek az Egy fokozatú (Single-stage object detection), illetve a Két fokozatú (Two-stage object detection) objektum detekciós eljárások.

2.4.1. Két fázisú objektum detekciós eljárások

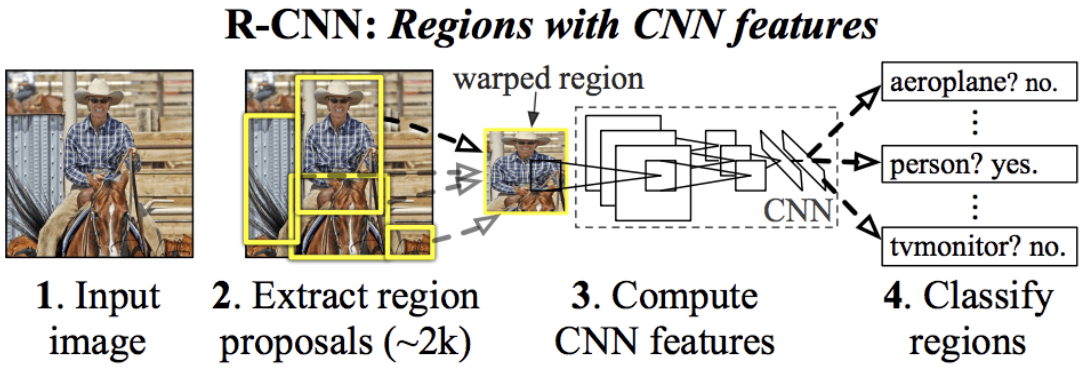
Ezen családba tartozó modellek esetén két különböző lépésben történik az objektumok lokalizációja és klasszifikálása. A modell család legnépszerűbb eljárása a 2014-ben publikált R-CNN (Region-Based Convolutional Neural Network) [16] architektúra, mely az elsők között alkalmazta sikeresen a konvolúciós hálókat az objektum lokalizáció, szegmentálás és detektálás problémáira. Az R-CNN modell három fő modulból épül fel:

1. **Az objektumokat tartalmazó lehetséges régiók meghatározása (Region Proposal):** Kategóriafüggetlen régiójavaslatok (bounding box jelöltek) generálása. A publikált megoldásban ez a gépi látásban használt Selective Search algoritmus használatával történik.

2. **Jellemzők kinyerése (Feature extractor):** Mély konvolúciós neurális hálózat segítségével a régió jelöltekben található jellemzők kinyerése. A szerzők ehhez a területen gyakran alkalmazott AlexNet [22] architektúrát használták fel.

3. **Osztályozó (Classifier):** A kinyert jellemzők alapján meghatározza, hogy mely javasolt régióban milyen objektum található. Erre a célra az ismert lineáris SVM klasszifikációs eszköz is jól használható. Az előző modullal együtt az Osztályozó valósítja meg a két fokozatú objektum detekció második fokozatát.

Az R-CNN architektúra komponensei a 2.2 ábrán láthatóak. Az R-CNN mellett ebbe a modell családba tartoznak még ennek továbbfejlesztett változatai, többek között a Fast R-CNN [15] és Faster-RCNN [40].



2.2. ábra. R-CNN architektúra bemenete és három komponense [16]

2.4.2. Egy fázisú objektum detekciós eljárások

Az objektum detektáló eljárások másik népszerű családja olyan modelleket tartalmaz, melyek az objektum lokalizációt (lehetséges régiók meghatározását) és az ob-

jektumok klasszifikálását egy lépésben végzik el. Ide tartozik többek között az SSD (Single Shot MultiBox Detector) [30] és a YOLO (You Only Look Once) [39] architektúra. Ellentétben az előbb bemutatott R-CNN modellekkel, a Single-stage detektorok képesek valós idejű objektum detekcióra is. A YOLO akár 45 FPS (frames per second) sebességet, míg a kisebb verziója (Fast YOLO [43]) pedig 155 FPS-t is elérhet NVIDIA Titan X videókártyán futtatva [34]. Többek között gyoraságának és alacsony predikciós hibájának köszönhetően napjaink egyik legnépszerűbb objektum detekciós modellje lett, így a javasolt FedMOD eljárásban (lásd 4. fejezet) is a kliens modellek YOLO architektúrák. A YOLO modell felépítésének és működésének részletes bemutatása a következő fejezet tárgya. A modell családba tartoznak még többek között a YOLOv2, YOLOv3, YOLOv4, YOLOv5, CenterNet [11] architektúrák is.

3. fejezet

Federált objektum detekció

A federált objektumdetekció területén a bevezetőben említett számos gyakorlati alkalmazás ellenére kevés publikáció létezik, ezek legnagyobb hányada is olyan megoldásokat taglal, melyek egy globális modell elosztott tanítását valósítják meg. Erre példa a FedVision [31] eljárás, mely a FedAvg módszernek objektumdetekcióra történő alkalmazását mutatja be. Hasonlóan az önvezető autók elosztott tanítására ad lehetséges megoldást a [20] tanulmány, mely szintén egy globális modell betanítását teszi lehetővé. Látható, hogy ezen megoldások nem alkalmazhatóak multitaszk környezetben, amikor egyes kliensek eltérő taszkokat is tanulhatnak, például régiónként (pl. környezet hasonlósága alapján) definiált modellek kollaborációja során. A következő fejezetben bemutatott FedLinked (FedMOD) eljárás az ilyen jellegű problémákra ad egy lehetséges megoldást. Ezen fejezet pedig a kollaborációban részt vevő modellek architektúráját, illetve a kiértékeléshez használt adathalmazt és ennek elosztott környezetbe történő definiálást ismerteti.

3.1. YOLOv1

A FedLinked eljárást három résztvevőt tartalmazó elosztott tanulási környezetben értékelem ki, ahol az egyes résztvevő kliensek YOLOv1 modellt tanítanak. A YOLOv1 modell felépítését az alábbiakban részletezem.

3.1.1. Architektúra

Az eredetileg publikált YOLOv1 architektúra a 3.1 ábrán látható. Ezen modellben 24 konvolúciós réteget követ 2 teljesen összekötött réteg (fully connected layer). Bemenetként 448 x 448 x 3 méretű képeket vár, végső kimenete pedig egy 7 x 7 x 30 méretű tenzor. Mivel a YOLO a Single-Stage Detector családba tartozik, ezért a 7 x 7 x 30 méretű kimenet tartalmazza úgy a bounding box-okra, mint az azonosított osztályokra vonatkozó információt a következő megfontolások alapján:

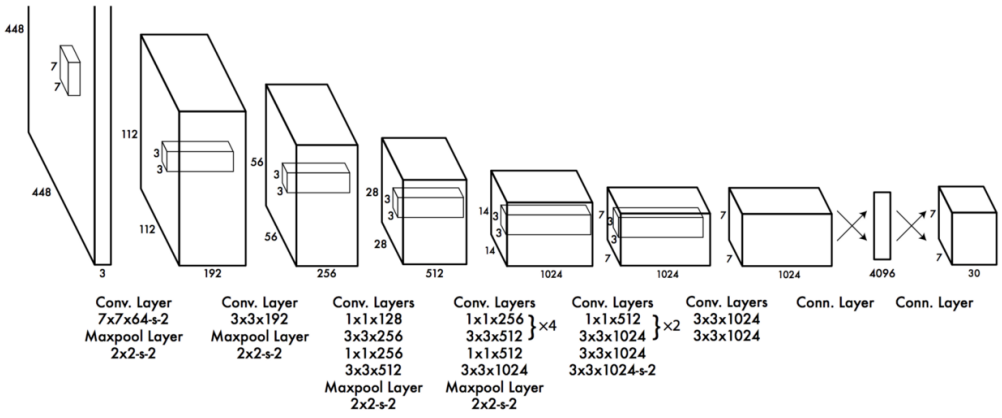
- A YOLOv1 modell a bemeneti képet 7 x 7 cellára bontja, ezen cellákban próbál azonosítani egy-egy objektumot.
- Az eredeti YOLOv1 20 különböző osztályba tanulta besorolni az azonosított objektumokat, ezért minden egyes kimeneti cella első 20 csatornája az osztályokon értelmezett diszkrét valószínűségi eloszlást tartalmazza. Elosztott kör-

nyezetben az egyes kliensek eltérő számú taszkot tanulnak, így esetükben ez a csatorna szám változni fog.

- Cellánként 2 bounding box-ot próbál illeszteni a modell az ott esetlegesen felismert objektumra. Egy bounding box-ot 5 érték jellemez: milyen valószínűséggel van objektum az adott tartományban (*objectness score*), a téglalap 2 átellenes csúcsának koordinátái (4 érték). A költségfügvényben objektum azonosítása esetén csak a nagyobb *objectness score*-al rendelkező bounding box szerepel.

- Az osztályokhoz tartozó valószínűségekből (20) és a két bounding box-hoz tartozó értékekből (10) áll össze az egy cellát jellemző 30 hosszú vektor.

A modell csak azon cellákban jelzi azonosított objektum jelenlétét, ahol a legnagyobb *objectness score* értéke meghalad egy hiperparaméterként megadott küszöb értéket (a szerzők 0.4 értéket használtak erre a célra).



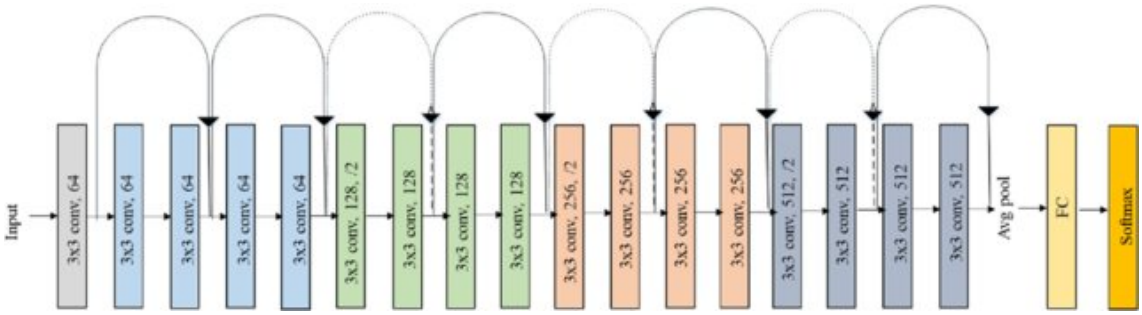
3.1. ábra. YOLOv1 architektúra [39]

A modell teljesítményének növelése és az elosztott környezetben történő használhatóság érdekében a következő módosításokat eszközöltem a a publikált YOLOv1 architektúrán:

- A felső 4 konvolúció alatti rétegeket az ismert ResNet-18 [17] architektúrára cseréltem (fully connected layer és softmax nélkül). Ez az architektúra a 3.2 ábrán látható. Inicializációnál az ImageNet adathalmazon előtantott súlyokat használtam a ResNet-18 backbone esetében. A teljesen random inicializációhoz képest ez a lépés az eredmények 30%-os javulását eredményezte.

- A legfelső két teljesen összekötött réteget 1 x 1-es konvolúciókra cseréltem. Ennek motivációja a modell paraméterek csökkenése, továbbá a tisztán konvolúciós rétegek alkalmazása megengedi, hogy a tudás transzfer során csak azon cellákhoz tartozó jellemző vektorokat vegyünk figyelembe, melyek az azonosított objektumhoz tartoznak (lásd következő fejezet).

- A felső négy konvolúciós rétegek között a YOLOv1 után publikált Batch Normalizáció használatát is bevezettem, mely Deep Learning architektúrák tanításánál hasznos eszköznek bizonyult.



3.2. ábra. ResNet-18 architektúra [38]

3.1.2. Célfüggvény

Az alábbiakban a YOLOv1 modell tanításánál használt költségfüggvényeket mutatom be, ezek a publikációban definiált Sum of Squared Error (SSE) alapú, illetve az objektum detekciónál gyakran használt Focal Loss [27] költségvüggvény.

3.1.2.1. SSE alapú költségfüggvény

A célfüggvény definiálásához vezessük be az alábbi jelöléseket:

- λ_{coord} : a bounding box-ok koorinátáira vonatkozó költség súlya
- λ_{noobj} : az objektumot nem tartalmazó cellákhoz tartozó *objectness score*-ok költségének súlya
- S^2: Cellák száma, esetünkben 7 x 7
- B: Predikált bounding box-ok száma cellánként, esetünkben $B = 2$
- $\mathbb{1}_{ij}^{\text{obj}}$: Indikátor függvény, értéke azon i, j cellák esetében egy, ahol van objektum a kimeneti címkék alapján. Hasonlóan definiálható $\mathbb{1}_{ij}^{\text{noobj}}$ is.
- $\hat{x}_i, \hat{y}_i, x_i, y_i$: Az i. cellában a legnagyobb *objectness score*-al rendelkező bounding-box prediktált és tényleges bal felső koordinátái
- $\hat{w}_i, \hat{h}_i, w_i, h_i$: Az i. cellában a legnagyobb *objectness score*-al rendelkező bounding-box prediktált és tényleges szélessége, magassága
- $\hat{C}_i$: Az i. cellában a prediktált *objectness score*-ok maximuma
- C_i: Az i. cella tényleges *objectness score* értéke
- $\hat{p}_i(c)$: Az a prediktált valószínűség, hogy az i. cellában azonosított objektum a c osztályba tartozik. $p_i(c)$ az ennek megfelelő elvárt érték.

A YOLOv1 tanításához a szerzők által definiált költségfüggvényt a 3.1 kifejezés írja le.

$$
\begin{aligned}
&\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
&\quad + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
&\quad + \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&\quad + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
&\quad + \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{ classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2
\end{aligned}
\tag{3.1}
$$

A célfüggvény 5 tagból áll, melyek az alábbi interpretációval bírnak:

- Első tag: prediktált és elvárt bounding box koordináták közti négyzetes eltérések összege, azon cellák esetében, ahol van objektum. A koordináták normalizáltak.
- Második tag: prediktált és elvárt bounding box méreteinek négyzetgyökei közti négyzetes eltérések összege. A gyökvonás motivációja, hogy a kis eltérések nagyobb bounding box-ok esetén kevésbé számítanak, mint a kis bounding box-ok esetében. Az első két tag $\lambda_{\text{coord}} = 5$ súllyal szerepel a teljes költségben, ezáltal hangsúlyozva a bounding box-ok pontos meghatározásának fontosságát.
- Harmadik és negyedik tag: az *objectness score*-ok pontos meghatározásáért felelnek. A képek celláinak nagy része az esetek többségében nem tartalmaz semmilyen objektumot. Emiall a cellák többségének *objectness score*-ját nulla környezetébe kényszeríti, gyakran felülírva az objektumokat tartalmazó cellákból származó gradienseket, és instabillá téve a modellt.Ezt elkerülendő, külön tagban szerepelnek az objektumokat nem tartalmazó cellák, melyek *objectness score*-ját befolyásoló költség $\lambda_{\text{noobj}} = 0.5$ értékkel van súlyozva.
- Ötödik tag: biztosítja, hogy a detektált objektumot helyes osztályba sorolja a modell, hozzáadva a teljes költséghez az elvárt és prediktált valószínűségi eloszlás vektorok négyzetes eltérését.

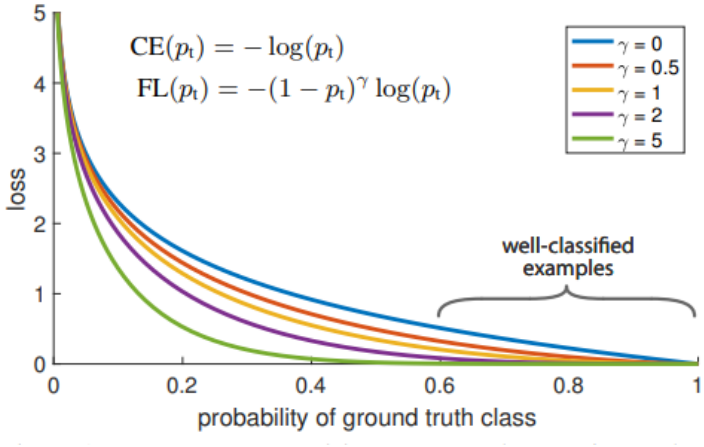
3.1.2.2. Focal Loss

A YOLOv1 megjelenését követő években publikálták a Focal Loss [27] költségfüggvényt, mely igencsak hasznosnak bizonyult az objektum detekció probléma megoldásában. A Focal Loss a kereszt entrópia költségfüggvény általánosított változata, amely a nehezebben osztályozható tanítómintákra fókuszál biztosítva ezáltal, hogy a tanítás előrehaladtával a predikció minősége javul a nehéz mintákon is, ahelyett, hogy csak a könnnyű minták felismerésében legyen egyre magabiztosabb a modell. Ezt egy úgynevezett "lesúlyozás" (Down weighting) technikával éri el, mely a gyakor-

latban a kereszt entrópiához egy moduláló tényező hozzáadását jelenti a 3.2 kifejezés alapján.

$$\text{FocalLoss} = -\sum_{i=1}^{i=n} (i - p_i)^{\gamma} \log_b (p_i) \tag{3.2}$$

γ a fókuszálás intenzitását befolyásoló paraméter, melynek hatását a Focal Loss függvény struktúrájára a 3.3 ábra szemlélteti.



3.3. ábra. Gamma értékének hatása a Focal Loss függvényre [27]

Ahogy a Focal Loss-al tanított modell predikcióihoz tartozó valószínűségek nőnek ($p_i \to 1$), a moduláló tényező nullához tart, így a jól osztályozott mintákhoz tartozó költség csökken, míg a hibásan osztályozott minták költsége nő. Minél nagyobb γ értéke, a függvény a könnyű példákat annál jobban lefelé súlyozza, csökkentve ezzel a veszteségfüggvényre gyakorolt hatásukat. A szerzők kísérletei alapján $\gamma = 2$ bizonyult a legelőnyösebb választásnak. $\gamma = 0$ esetben a klasszikus kereszt entrópiát kapjuk. Az előbbi leírás alapján módosított YOLOv1 architektúrát betanítottam és kiértékeltem a PascalVOC (lásd 3.2 alfejezet) adathalmazon mindkét költségfüggvénnyel. Az eredményekben nem mutatkozott jelentős eltérés, így az 5. fejezetben bemutatott kiértékelés során minden esetben a YOLOv1 publikációban definiált SSE alapú költségfüggvényt használom.

3.1.3. Teljesítmény kiértékelése

Az objektumdetekciós modell kiértékeléséhez a területen gyakran használt mean average precision (mAP) metrikát használom. Az ehhez szükséges komponensek, illetve a teljes kiértékelési algoritmus bemutatása ezen alfejezet témája.

3.1.3.1. Intersection over Union

Annak eldöntésére, hogy a modell által prediktált bounding box megfelelő helyen található-e, azaz a predikció helyes-e, az egyszerű Intersection Over Union (IoU) metrika alkalmazható.

Legyen B_p egy prediktált bounding box, és B_g a hozzá legközelebbi tényleges bounding box. Ekkor a B_p-hez tartozó IoU érték az alábbi módon számolható:

$$IoU(B_p) = \frac{B_p \cap B_g}{B_p \cup B_g} \tag{3.3}$$

Az egyenletben $B_p \cap B_g$ a két bounding box metszetét képező téglalap területét, $B_p \cup B_g$ pedig az uniójuk területét jelenti. Amennyiben a kapott IoU érték egy megadott küszöbérték fölött van, akkor azt mondjuk, hogy a vizsgált bound box helyesen lokalizált egy objektumot és további vizsgálatok eszközölhetőek a klasszifikáció helyességét illetően.

3.1.3.2. Non-maximum suppression

Előfordulhat, hogy egy adott objektumra több, IoU alapján helyesen prediktált bounding box is illeszkedik, ekkor ezek közül a "legjobb" kiválasztása és a többi eldobása a cél. A Non-maximum suppression (NMS) algoritmus pontosan ennek megoldásában segít az alábbi lépések végrehajtásával:

1. Adott a prediktált bounding box-ok listája B, az ezekhez tartozó *objectness score*-ok listája S, a szűrt bounding box-ok lisátja D (mely kezdetben üres), valamint egy küszöbérték N.

2. Kiválasztjuk az B listában levő legmagasabb *objectness score*-hoz tartozó bounding box-ot és áthelyezzük a D listába.

3. Töröljük a B listából az összes olyan bounding box-ot, melyeknek az áthelyezett bounding box-al vett IoU értéke nagyobb, mint N.

4. Folytatjuk a 2. lépéssel, ameddig B nem üres

Egy példa az NMS algoritmus által megvalósított szűrésre a 3.4 ábrán látható.

3.4. ábra. Példa Non-maximum Suppression alkalmazására [42]

3.1.3.3. Mean Average Precision

A betanított objektum detekciós modellek teljesítményének numerikus kiértékeléséhez a Mean Average Precision (mAP) metrikát használom, mely az NMS algoritmussal történő szűrés után a következőképpen számolandó:

1. Legyen C a tanult objektum osztályok száma, ekkor minden $c \in C$ osztályra végrehajtjuk az alábbi lépéseket:
   (a) Kiválasztjuk a c osztályhoz tartozó prediktált bounding box-okat
   (b) A kiválasztott bounding box-okat True Positive (TP), vagy False Positive (FP) kategóriába soroljuk, aszerint, hogy egy tényleges bounding box-al vett IoU értékük elegendően magas-e vagy sem
   (c) A TP, FP értékekből származtatjuk a c osztályra vonatkozó Average Precision értéket
2. Az osztályonként számolt Average Precision értékeket átlagoljuk, így kapjuk a végső mAP értéket.

Megjegyzés: A szakirodalomban a fenti algoritmust általában kiegészítik azzal, hogy több különböző IoU küszöbérték mellett is kiszámolják a mAP értékeket és ezek átlagából nyerik a végső mAP értéket. Ezen kiegészítést én a munkám során a korlátozottan rendelkezésre álló számítási kapacitás miatt nem alkalmaztam.

3.2. Pascal VOC adathalmaz

A fejlesztett FedLinked (FedMOD) eljárást a Pascal VOC [12] adathalmazon értékelem ki. Az adathalmaz bemutatása ezen alfejezet tárgya.

3.2.1. Adathalmaz felépítése

A Pascal VOC adathalmaz több mint 20.000, objektum detekciós modell tanítására alkalmas, annotált képet tartalmaz 20 különböző osztályból. Ezek rendre: repülőgép, bicikli, madár, hajó, palack, busz, autó, macska, szék, tehén, asztal, kutya, ló, motorbicikli, személy, cserepes virág, júh, fotel, vonat, TV/monitor. Minden egyes felcímkézett mintához tartozik egy szöveges, annotációs fájl, melynek annyi sora van, ahány objektum található a képen, egy sorban pedig a következő adatok találhatóak meg az adott objektumról: osztály sorszáma, amelybe az objektum tartozik, a bounding box bal felső koordinátái, szélessége és hosszúsága a teljes kép szélességével/hosszúságával normalizálva. Az adathalmazt úgy osztottam fel tanító, illetve teszt mintákra, hogy ez teljes mértékben megegyezzen a szakirodalomban található kiértékeléseknél használt felosztással [36].

3.2.2. Augmentálás

A tanító adatok augmentációja fontos szerepet játszik bármely gépi látáshoz kapcsolódó feladat esetében. Augmentálás során minden egyes epochban a tanító adatokat random transzformációknak vetjük alá, ilyen például az eltolás, forgatás, skálázás vagy pedig zaj hozzáadása.

Az egyszerű klasszifikációs feladatokkal szemben az objektumdetekciónál egy plusz nehézséget okoz az augmentáció alkalmazásánál, hogy a képekhez rendelt bounding box-okra is alkalmazni kell affin trnaszformációkat. Ennek megvalósítására az Albumentation [8] Python könyvtárat használom, mely automatikusan elvégzi az adatok augmentálásával a bounding box-ok megvelelő transzformációját is. Egy képnek az augmentálás előtti, illetve Albumentation modul segítségével augmentált változata és a hozzájuk tartozó bounding box-ok a 3.5 ábrán láthatóak.

(a) Augmentáció előtt **(b)** Augmentáció után

3.5. ábra. Példa Albumentation-el történő augmentációra

A tanítómintákon tükrözés, eltolás, forgatás, skálázás és kontraszt módosítás alkalmazása 10%-os teljesítmény növekedést eredményezett a teljes adathalmazon betanított, előbbi fejezetben leírt YOLOv1 architektúra esetében.

3.2.3. Adathalmaz felosztása kliensek között

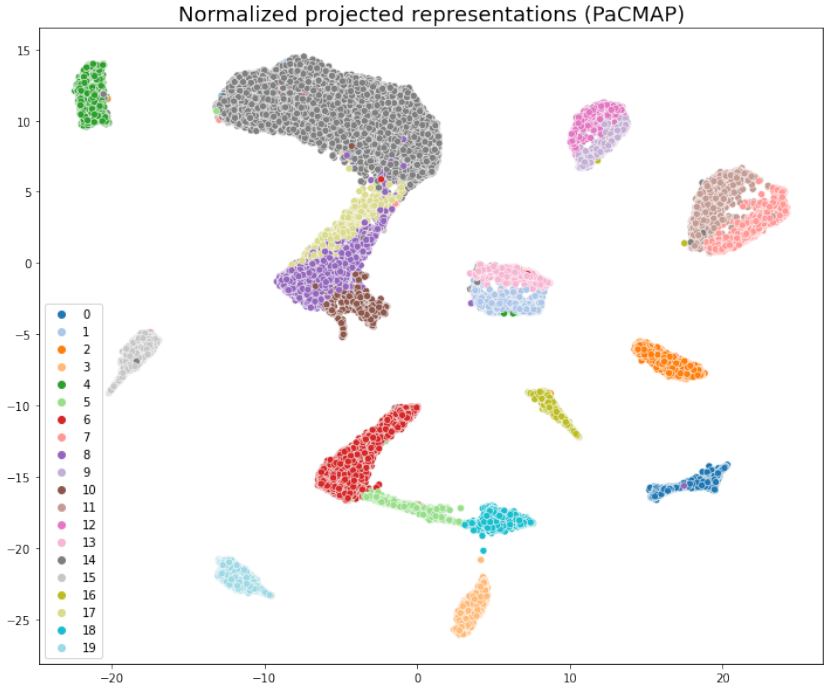
A Pascal VOC adathalmaz federált, multitaszk környezetbe történő transzformálásához és a kliensek közötti felosztásához először szükséges az egyes osztályok/taszkok között a betanított YOLOv1 modell által értelmezett hasonlóságok feltérképezése, hogy tudjuk, mely taszk csoportok között van esély tudás transzfer megvalósulására. A hasonlóságok hatékony felmérését követően előállítható a tanítómintáknak és taszkoknak egy olyan felosztása, mely megengedi, hogy a hasonló feladatokat tanuló kliensek között ténylegesen történjen tudás transzfer.

3.2.3.1. Taszkok közötti hasonlóság mérése

Az elvégzett kísérletek során a Pascal Voc adathalmazban található 20 osztály mindegyikére külön taszkokként tekintünk. Ezek közti hasonlóságok mérésére két különböző módszert valósítottam meg. Mindkét esetben egy YOLOv1 modellt tanítottam be a teljes tanító adathalmazon és a tanító mintáknak a modell által értelmezett reprezentációit vizsgáltam. Reprezentációként az 1×1-es konvolúciós réteg bemeneti aktivációit definiáltam. Ezen 7x7x1024 dimenziós reprezentációknak is azon 1024

méretű feature vektoraival dolgoztam, melyek a kimeneti réteg alapján objektumot tartalmazó cellához tartoznak (szűrt reprezentációs vektorok).

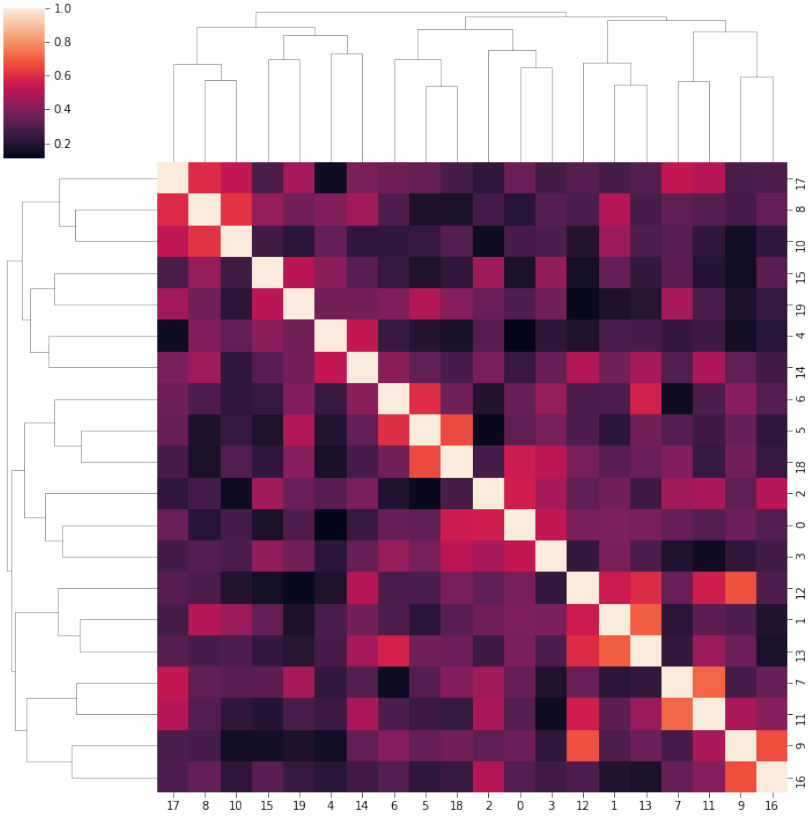
Az első módszer esetében a szűrt reprezentációs vektorokat egy távolságtartó projekciós eljárással, a PacMAP-el [46] két dimenziós altérbe vetítettem a 3.6 ábrán látható módon. A levetített 2D-s vektorokat aszerint színeztem, hogy milyen objektum detektálásáért felelnek.



3.6. ábra. Két dimenziós altérbe vetített, szűrt reprezentációs vektorok megjelenítése

Az ábrán látható, hogy a betanított modell taszkok szerint jól elkülöníthető, homogén csoportokba tudta rendezni az egyes adatpontokhoz tartozó reprezentációs vektorokat. A levetített reprezentációs vektorok közelsége alapján megfigyelhető továbbá erősebb hasonlósági viszony például a 17 (fotel) - 8 (szék); 1 (bicikli) - 13 (motorbicikli); 9 (tehén) - 12 (ló) ; 7 (macska) - 11 (kutya) taszkpárok között, mely az emberi intuició alapján meghatározható hasonlósági viszonyoknak is megfelel.

Annak érdekében, hogy elkerüljem a két dimenzióra történő vetítés során keletkező információvesztésből adódó hamisan detektált hasonlósági viszonyokat, egy másik módszerrel is ellenőriztem az előbb látott eredményeket. A szűrt, magas dimenziós reprezentációs vektorokat taszkonként átlagoltam és az így kapott 20 vektor között vizsgáltam páronként a koszinusz hasonlóságok értékeit. Az eredményül kapott heatmap a hasonlósági értékek alapján hierarchikusan klaszterezett taszkokkal a 3.7 ábrán látható.



3.7. ábra. Taszkpárokhoz tartozó reprezentációs vektorok közötti koszinusz hasonlóságok heatmap-je

Megfigyelhető, hogy ezen vizsgálat megerősítette az előbbi eljárás által feltárt hasonlósági viszonyok jelentős részét, így például a 17 (fotel) - 8 (szék); 1 (bicikli) - 13 (motorbicikli) ; 7 (macska) - 11 (kutya) párok között. Továbbá új hasonlóságokat is sikerült felfedezni példáué a 9 (tehén) - 16 (júh) taszkpár esetében úgy, hogy közben az előbbi módszer által kiemelt 9 (tehén) - 12 (ló) hasonlóság is fennáll.

A két módszer összességében egymással korreláló eredményt adott, mely alapján magabiztosan következtethetünk bizonyos taszkpárok közötti (modell által értelmezett) hasonlóságokra és különbségekre.

3.2.3.2. Felosztások

Felhasználva az előbbiekben igazolt, modellek számára is felismerhető hasonlósági információkat, elkészíthetjük a taszkoknak és adatpontoknak olyen felosztásait, melyek a tudás transzfernek általunk szabályozható fokozatait engedi meg és ezáltal mérhetjük tisztán a FedMOD kollaboratív tanító eljárás által megvalósított tudás transzfert.

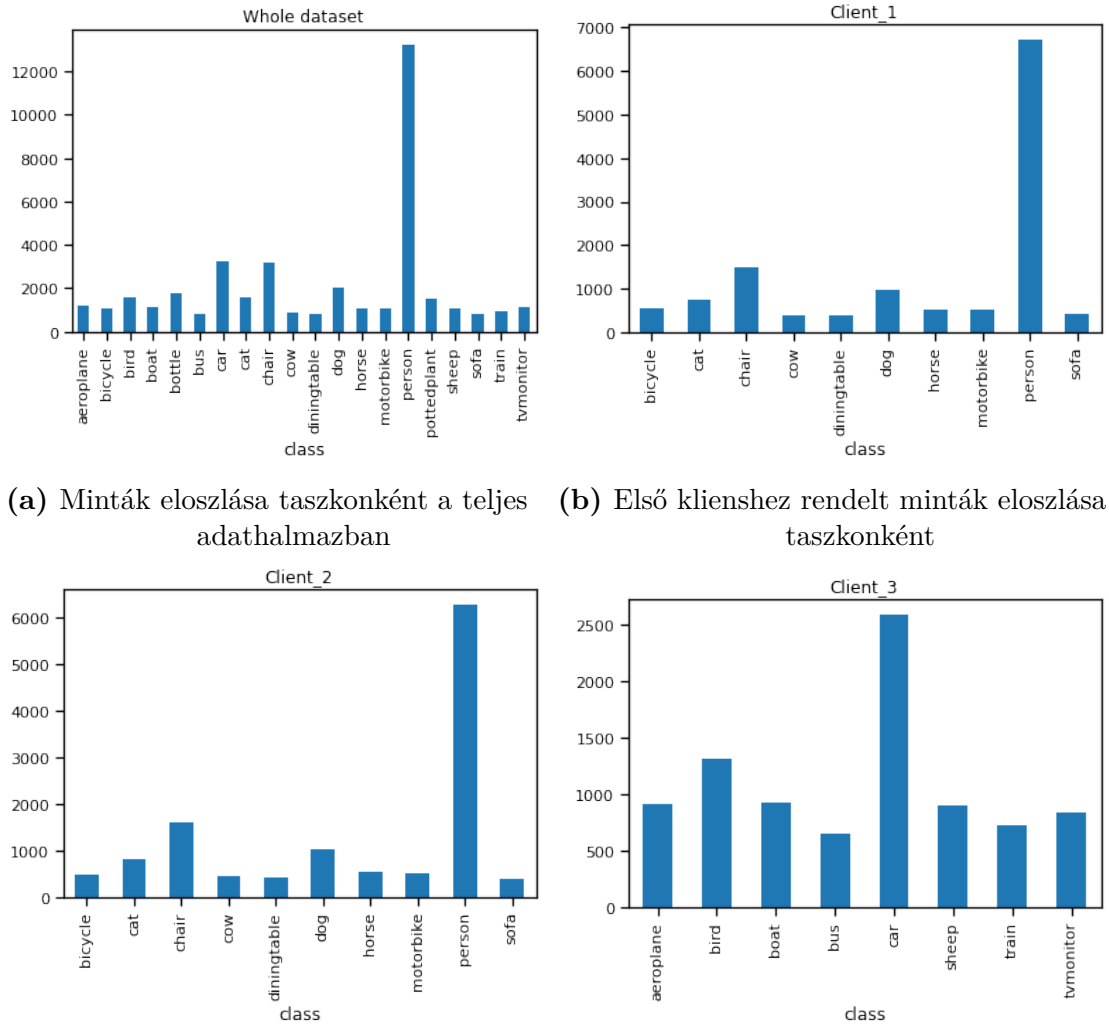
Ennek érdekében 3 kliens taszkjai esetében 3 különböző felosztást definiáltam:

- **Teljes átfedés:** Két kliens teljesen ugyanazon taszkokat tanulja (természetesen különböző tanító adathalmazokon), a harmadik kliens ezektől eltérő taszkokat tanul.
- **Részleges átfedés:** Két kliens részlegesen átfedő (megegyező) és hasonló taszkhalmazt tanul, míg a harmadik kliens mindkettőtől eltérő taszkokat.

- **Nincs átfedés:** Minden kliens teljesen diszjunkt taszkhalmazt tanul, viszont két kliens az előbbi vizsgálatok alapján hasonlónak definiált taszkokat tanul.

Sejtésem alapján mindhárom esetben a megegyező vagy hasonló taszkokat tanuló kliensek között hangsúlyosabb lesz a kollaboráció pozitív hatása, a kevésbé hasonló vagy eltérő taszkokat tanuló klienspárok között pedig kisebb mértékben lesz kimutatható a tudás transzfer. A javasolt FedMOD eljárást ezen felosztások mentén fogom kiértékelni.

A "teljes átfedés" esetben a taszkoknak és adatpontoknak a kliensek között egy lehetséges szétosztását a 3.8 ábrázolja.



(a) Minták eloszlása taszkonként a teljes adathalmazban

(b) Első klienshez rendelt minták eloszlása taszkonként

(c) Második klienshez rendelt minták eloszlása taszkonként

(d) Harmadik klienshez rendelt minták eloszlása taszkonként

3.8. ábra. Adatpontok és taszkok eloszlása a három kliens között

A bal felső ábra mutatja a teljes adathalmaz mintáinak számát taszkonként, míg a maradék három ábra a három klienshez rendelt minták eloszlását jeleníti meg taszkonként a "teljes átfedés" esetben. A felosztás során egy kép kerülhet több klienshez is, viszont egy objektum maximum egy klienshez rendelhető. Az ábrákon

látható, hogy a felosztás úgy jött létre, hogy mindhárom kliensnél az egyes taszkokhoz tartozó mintaszámok viszonya arányosan megegyezzen a teljes adathalmazban látott taszkonkénti mintaszámok arányaival.

3.2.4. Publikus adathalmaz

A javasolt FedMOD algoritmus, mint részben félig ellenőrzött tanulási eljárás, a kollaboráció során címkézetlen, bármely kliens számára publikusan hozzáférhető adathalmazt is használ (további részletek a következő fejezetben).

A Pascal VOC adathalmaz alapvetően több, mint 40.000 képet tartalmaz, de mivel ezek közül "csak" 20.000 van felcímkézve, ezért a maradék, kb. 20.000 elemből álló címkézetlen halmazt használom publikus adathalmazként. Ezen minták felhasználhatóságának tesztelése érdekében őket random mintavételeztem és egy betanított YOLO modell-el felcímkéztem. Az így kapott címkék helyességét manuálisan ellenőriztem és ez alapján feltételeztem a teljes, címkézetlen adathalmaz megfelelő minőségét, illetve megbizonyosodtam arról, hogy az ebben található képek eloszlása nem tér el túlzott mértékben a felcímkézett adatokétól.

3.3. COCO adathalmaz

A javasolt eljárás átfogó kiértékelése érdekében a PascalVOC mellett a COCO (Common Objects in Context) [28] adathalmazon is kipróbáljuk a FedMOD keretrendszert. Amellett, hogy következtetéseinket így nem csak egy adathalmaz alapján vonjuk le, kipróbálhatjuk, hogyan teljesít a FedMOD eljárás nem ugyanazon eloszlásból származó (non-i.i.d.) privát adathalmazokkal rendelkező kliensek esetén (például, ha néhány kliens privát mintái a PascalVOC halmazból, a többi résztvevőjé pedig a COCO adathalmazból származik). Ilyen jellegű vizsgálatot az 5.10 fejezet taglal.

A COCO egy nagyméretű, több mint 200.000 felcímkézett mintát tartalmazó objektum detekciós, szegmentációs és feliratozási (captioning) adathalmaz. Az egyes képekhez tartozó annotációs fájlok jó minőségben tartalmazzák az előbb említett három problémához szükséges címkék mindegyikét. Az általam használt COCO 2017 adathalmaz objektum detekció szempontjából 90 osztályt különböztet meg, melyeknek egy, a PascalVOC adathalmaz osztályaival is átfedő részhalmazát használtam fel a kiértékelések során. A képek augmentálása és felosztása a résztvevők között hasonló megfontolások mentén történt, mint a PascalVOC esetében.

A következő fejezetben részletesen bemutatom a javasolt FedMOD eljárást és enek főbb komponenseit.

4. fejezet

A FedMOD eljárás

A FedMOD federált multitaszk obejtum detekciós eljárás alapötletét a tavalyi TDK dolgozatomban [5] bemutatott, illetve a [23] cikkben is publikált, szintén saját tervezésű FedLinked módszer adja.

Az ismert federált tanuló algoritmusokkal szemben a FedMOD képes különböző számú, hasonlóságú taszkokat tanuló modellek között is kollaborációt megvalósítani. A résztvevő modellek nem kell megegyező architektúrával rendelkezzenek ahhoz, hogy részt vegyenek az adatvédelmet megőrző kollaboratív tanításban.

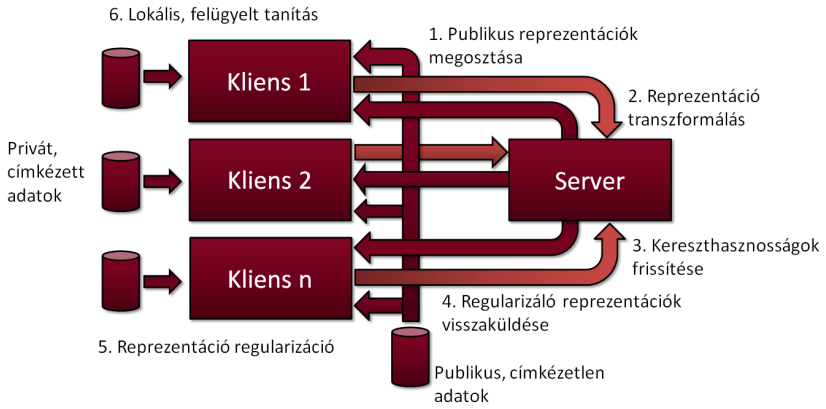
Az eljárás megbecsli a kliensek közötti kereszthasznosságokat, mely alapján a résztvevők számára kölcsönösen előnyős reprezentáció regularizációt valósít meg. A regularizáció során az egymással korreláló taszkokat tanuló kliensek nagyobb súllyal befolyásólják egymás reprezentációit, ezáltal intenzív tudás transzfert eredményezve köztük. Ez az együttműködés úgy valósul meg, hogy sem a kliensek privát adatait, sem a paramétervektoraikat, sem a tanult taszkok listáját nem kell megosztani a többi résztvevővel.

A FedMOD algoritmus bemenetét a kliensek képezik, melyek mindegyike rendelkezik saját modellel, taszkokkal és egy ezekre illeszkedő, privát, címkézett adathalmazzal. Továbbá minden kliens számára rendelkezésre áll egy címkézetlen, publikus adathalmaz is.

A kollaboráció megkezdése előtt minden kliens konvergenciáig tanítja a saját modelljét a privát adathalmazán. Ennek motivációja, hogy erősebb, előtanított reprezentációkkal rendelkező kliensek esetén kevesebb tanulási ciklus (alább definiálva) végrehajtása is elegendő a kollaboratív tanítás konvergenciájához. A tanulási ciklusok és ezáltal a kliensek és szerver közötti kommunikáció redukálása több szempontból is előnyös: amellett, hogy így kevesebb számításigényt és erőforrást vesz igénybe a kollaboráció (főként, ha a kliensek eltérő hálózaton vannak), adatvédelmi szempontból is fontos, hogy minél kevesebbszer osszanak meg információt a szerverrel a kliensek. Ilyen megfontolásból az 5. fejezetben bemutatott eredményeket is előtanítást tartalmazó futtatásokból generáltam. Az előtanítást követően minden kliens kiszámítja a publikus adathalmaznak a saját modellje által értelmezett reprezentációs vektorait[1], és ezen vektorok közül elküldi a szervernek azokat, melyek valamely detektált objektum leíróit tartalmazzák. A reprezentációk közvetlen összehasonlítása nem feltétlenül elég informatív, ezért második lépésként a szerver lineáris transzfor-

[1]Reprezentációs vektor alatt az előző fejezetben bemutatott, módosított YOLOv1 architektúra kimenet előtti 1x1-es konvolúciójának aktivációs vektorait értjük.

mációt hajt végre a kliens párok reprezentációi között, így a módszer robosztussá válik az olyan jelenségekkel szemben, mint például: 2 kliens ugyanazon jellemzőket tanulja eltérő sorrendben. A transzformált reprezentációs vektorok alapján a szerver becslést ad a kliens párok kereszthasznosságára, a közelebb eső reprezentációk magasabb kereszthasznosságot eredményeznek. A következő lépésben a szerver a kereszthasznossági együtthatókat felhasználva minden kliens számára személyre szabott, regularizáló reprezentációt készít, és azt elküldi nekik. Egy kliens regularizáló reprezentációját a többi résztvevő reprezentációinak súlyozott összegeként kapjuk, ahol a súlyok a kereszthasznossági együtthatók. Ezt követően a kliensek néhány epoch-on keresztül tanulják a regularizáló reprezentációkat a publikus, címkézetlen adatokon. Így valósul meg a tudásátadás. Utolsó lépésként lokális, felügyelt tanulást hajtunk végre a kliensek privát adatain. Ezen lépések egy tanulási ciklust alkotnak, melyet a 4.1 ábrán is megfigyelhetünk. A FedMOD-al történő kollaboratív tanulás során több egymást követő tanulási ciklus kerül végrehajtásra.



4.1. ábra. Egy tanulási ciklus lépései a FedMOD eljárásban

Az alábbiakban az algoritmus főbb komponenseinek részletes bemutatása következik.

4.1. Súlyozott reprezentáció regularizáció

A FedMOD eljárás központi eleme a kliensek reprezentációinak kereszthasznosságát becslő és regularizációját megvalósító komponens. A módszer alapfeltevése a következő: egymással korreláló taszkokat tanuló modellek hasonló tulajdonságok kiemelését, ezáltal hasonló belső reprezentációt kell tanulniuk a jobb általánosító képesség elérése érdekében.

Legyen $f^{(r)}(x, \theta_k) \in \mathbb{R}^{S^2 \times C}$ a θ_k súlyokkal rendelkező k. modell belső reprezentációja az $x \in X^{(0)}$ publikus adathalmazból származó bemenet esetén. Ezen mátrix a 3.1.1 fejezetben leírtak alapján cellaszámnyi (S^2) C elemű jellemzővektorból áll, ahol C a reprezentációkat előállító konvolúciós réteg szűrőinek száma. Legyen továbbá $I_x \in (\{0..S\} \times \{0..S\})^{Ob}$ egy adott x publikus bemenethez tartozó index halmaz, mely azon cellák koordinátáit tartalmazza, ahol legalább egy résztvevő modell objektumot detektált, azaz a kimeneten az objectness score meghaladott egy előre definiált T_o küszöbértéket. Az I_x vektor mérete a bemeneti kép függvényében változhat. A modellek az általuk előállított reprezentációknak azon (r, c) koordinátájú

celláiban található C elemú jellemzővektorokat küldik el a szervernek, amelyekre teljesül, hogy $(r, c) \in I_x$.

A reprezentációk ilyen jellegű szűrésének motivációja, hogy elkerüljük a zajos vektorok dominanciáját a kollaboráció során. Hipotézisünk alapján az üres cellákhoz (melyek nem tartalmaznak objektumot) tartozó jellemző vektorok nem informatívak a taszkok szempontjából, zajt adnak a kereszthasznossági együtthatók becslésének, illetve a tudás transzfer folyamatához. Továbbá a kliensek és szerver közötti adatátvitel szempontjából is előnyös, hogy kevesebb reprezentációs vektor kerül megosztásra, ezáltal gyorsítva az eljárást. A könnyebb áttekinthetőség miatt a továbbiakban a k. kliens egy adott x bementéhez tartozó reprezentáció r. sorának c. cellájában található jellemző vektort jelöljük $f_{(r,c)}(x, \theta_k)$-val.

Az előbbiek ismeretében definiálhatjuk a kereszthasznossági együtthatók meghatározásáért és a reprezentáció regularizációért felelős célfüggvényt, melyet a 4.1 egyenlet ír le.

$$L_R(A, \theta_j) = \sum_{k \neq j} \sum_{x \in X^{(0)}} \sum_{(r,c) \in I_x} A_{k,j} \cdot \| f_{(r,c)}(x, \theta_k) - f_{(r,c)}(x, \theta_j) \|_2^2 \quad \text{feltéve, hogy: } \sum_{k \neq j} A_{k,j}^2 = \eta : \forall j \tag{4.1}$$

A célfüggvény minimalizálása egy adott j modell esetében θ_j súlyok és $\{A_{k,j}\}_{k \neq j}, \forall j, k \in 1, K$ kereszthasznossági együtthatók mentén történik. A függvény intuitív értelmezése a következő: abban az esetben amikor a k. és j. modell reprezentációi távol esnek egymástól, azaz a különbségvektor L_2 normájának négyzete nagy, akkor ezen modellpárhoz kis $A_{k,j}$ együtthatót rendel az optimalizációs eljárás a költség minimalizálása érdekében. Hasonlóképpen belátható az is, hogy azokhoz a modellpárokhoz fog nagyobb $A_{k,j}$ együttható tartozni, melyek között kisebb a különbség, azaz valószínűsíthetőleg a két modell olyan taszkokat tanul, melyekhez hasonló reprezentáció szükséges, így közöttük nagyobb súllyal valósítható meg tudás transzfer.

A célfüggvény egy triviális optimuma lenne $A_{k,j} = 0$, $\forall j, k \in 1, K$, ezért a $\sum_{k \neq j} A_{k,j}^2 = \eta : \forall j$ kényszert is bevezetjük, ahol $\eta > 0$ egy hiperparaméter, mely amellett, hogy a kereszthasznossági együtthatók értékét szabályozza, képes általánosan a kollaboráció erősségét is befolyásolni: nagyobb η intenzívebb kollaborációt eredményez a modellek között.

Mint ahogy több más federált tanuló eljárás is, a FedMOD alternáló optimalizációt használ a 4.1 célfüggvény optimalizálásához: először a kereszthasznossági együtthatókat tartalmazó mátrix optimalizációja történik az előző iterációban meghatározott θ paramétereket felhasználva, majd θ változókat optimalizáljuk az újonnan meghatározott kereszthasznossági együtthatómátrix alapján.

Összefoglalva az eddigieket, a súlyozott reprezentáció regularizációt megvalósító komponens lépései a következők:

1. Minden kliens meghatározza a publikus, címkézetlen adathalmaz összes elemére a belső reprezentációkat ($f^{(r)}(x, \theta_k)$), illetve cellánként az objectness score-okat.

2. Az objectness score-okat a kliensek elküldik a szervernek, mely ezek alapján meghatározza az I_x index vektorokat és elküldi minden résztvevőnek

3. A kliensek elküldik a szervernek a potenciálisan objektumokat tartalmazó cellák reprezentációs vektorait $(f_{(r,c)}(x,\theta_k), \forall (r,c) \in I_x)$

4. A publikus reprezentációk alapján a szerver végrehajtja az alternáló optimalizáció első lépését, frissíti a kereszthasznossági együtthatókat:

$$\begin{aligned} \{A_{k,j}\}^* &= \arg\min_{A_{k,j}} \sum_{k\neq j} \sum_{x\in X^{(0)}} \sum_{(r,c)\in I_x} A_{k,j} \cdot \|f_{(r,c)}(x,\theta_k) - f_{(r,c)}(x,\theta_j)\|_2^2 \\ \text{feltéve, hogy: } & \sum_{k\neq j} A_{k,j}^2 = \eta : \forall j \end{aligned} \tag{4.2}$$

A reprezentációk távolságának L_2 norma négyzetének minimalizálása jól közelíthető a belső szorzat maximalizálásával (4.3 egyenlet), ha a reprezentációs vektorok normájának szórása kicsi. Ezen feltétel biztosítására a modellek súlyainak L2 regularizációját is bevezettük a veszteségfüggvénybe (lásd 4.17 egyenlet).

$$\begin{aligned} \min . & - \sum_{k\neq j} \sum_{x\in X^{(0)}} \sum_{(r,c)\in I_x} A_{k,j} \cdot \langle f_{(r,c)}(x,\theta_k), f_{(r,c)}(x,\theta_j)\rangle \\ \text{feltéve, hogy: } & \sum_{k\neq j} A_{k,j}^2 = \eta \ , \forall j \end{aligned} \tag{4.3}$$

A fenti megfigyelés alapján, a kereszthasznossági együtthatók optimális értéke $(\{A_{k,j}\}^*)$ zárt alakban meghatározható Lagrange duális optimalizációt [7] alkalmazva. Figyelembe véve a $\sum_{k\neq j} A_{k,j}^2 = \eta : \forall j$ kényszert, az optimalizációs probléma megoldása:

$$\{A_{j,k}\}^* = \frac{\sum_{x\in X^{(0)}} \sum_{(r,c)\in I_x} \langle f_{(r,c)}(x,\theta_j), f_{(r,c)}(x,\theta_k)\rangle}{\sqrt{(\sum_{u\neq v}(\sum_{x\in X^{(0)}} \sum_{(r,c)\in I_x} \langle f_{(r,c)}(x,\theta_u), f_{(r,c)}(x,\theta_v)\rangle)^2)/\eta}} \tag{4.4}$$

5. A következő lépés a kliensek reprezentációs csonkjainak regularizálása (az alternáló optimalizálás második lépése), azaz bármely j kliens θ_j modell paramétereinek frissítése a 4.5 egyenlet alapján, az előbbiekben kiszámolt $A_{k,j}$ együtthatókat felhasználva.

$$\{\theta_j\}^* = \arg\min_{\theta_j} \sum_{k\neq j} \sum_{x\in X^{(0)}} \sum_{(r,c)\in I_x} A_{k,j} \cdot \|f_{(r,c)}(x,\theta_k) - f_{(r,c)}(x,\theta_j)\|_2^2 \tag{4.5}$$

Annak érdekében, hogy ne sérüljön a privacy, a szerver a kereszthasznossági együtthatókkal súlyozott, aggregált reprezentációkat küldi el az egyes klienseknek, nem pedig külön a kereszthasznossági együtthatókat és a többi klienshez tartozó reprezentációs vektorokat. Legyen az x bemenetnek a j. kliens számára

küldött aggregált reprezentációja $f_j^*(x) \in \mathbb{R}^{S^2 \times C}$. Ekkor az r. sor c. cellájához tartozó vektort a következőképp kapjuk:

$$f_j^*(x)[(r,c)] = \begin{cases} f_{(r,c)}(x, \theta_j), \text{ ha } (r,c) \notin I_x \\ \sum_{k \neq j}(A_{k,j} \cdot f_{(r,c)}(x, \theta_k), \text{ ha } (r,c) \in I_x \end{cases} \tag{4.6}$$

Azaz, ha semelyik résztvevő szerint sincs objektum a cellában, akkor nem módosítjuk a jellemző vektort, ellenkező esetben a többi kliens reprezentációját tanulja a j. modell, kereszthasznosságokkal súlyozva. Ekkor a 4.5 egyenletet optimumát közelíthetjük a 4.7 megoldásával, mely iteratívan, gradient descent alapú módszerrel történik. A 4.6 definícióból látható, hogy nullától különböző gradiens csak azon cellák receptív területeire fog visszaterjesztődni, ahol potenciálisan objektum található.

$$\{\theta_j\}^* = \arg\min_{\theta_j} \|f^{(r)}(X^{(0)}, \theta_j) - f_j^*(X^{(0)})\|_2^2 \tag{4.7}$$

Ezen 5 lépés valósítja meg a kollaborációt a kliensek között. Látható, hogy a FedLinked eljáráshoz képest a FedMOD alkalmazása során a kliensek a szerver tudtára adják, hogy a publikus bemeneti képeken hol vélnek objektumot felfedezni, ami bizonyos mértékben sérti a tanulandó taszkok privát jellegét, viszont a kliensek egymás között továbbra sem osztják meg ezt az információt. Megfigyelhető viszont az is, hogy a kollaboráció során csak a címkézetlen, publikus adathalmazat használjuk fel, a privát adatok és a modell paraméterek is teljesen védve maradnak, ami számos gyakorlati alkalmazásnál (pl. orvosi döntéstámogatás) kritikus fontosságú lehet.

4.2. Reprezentáció átképzés kliensek között

A gyakorlatban előfordulhat az, hogy a hasonló taszkokat tanuló kliensek ugyanazon jellemzőket a reprezentációnak eltérő aktivációiba kódolják (például megtörténhet, hogy ugyanazon jellemzőt az egyik kliens a reprezentációjának i. aktivációjában, míg a másik kliens az ő reprezentációjának j. aktivációjában tárolja). Ebből az okból kifolyólag a különböző modellek reprezentációs vektorai közötti különbség számolása direkt módon nem feltétlenül elégséges a köztük lévő hasonlóság meghatározásához. A FedMOD keretrendszer ezen komponensében egy új algoritmust javaslunk, amely képes a kliens párok reprezentációi között olyan transzformációt végrehajtani, mely a fent említett eltéréseket feloldja a tudás transzfer előtt.

Lineáris transzformációt alkalmazunk a reprezentációs vektorokon, ahol $B_{k,j}$ mátrixxal történő szorzás jelenti a k. kliens reprezentációjának j. kliens alterébe történő transzformációját. Ez a transzformáció tovább gyorsítja a konvergenciát, és megkönnyíti a kliensek számára, hogy más kliensek belső reprezentációiból tanuljanak úgy, hogy közben saját, önálló reprezentációt tudnak fenntartani. Az optimális transzformációs mátrix egy lépésben, analitikusan számolható az alábbi származtatás szerint:

A komponens bevezetével a 4.1 célfüggvény a következőképp módosul:

$$L_R(A, \theta_j) = \sum_{k \neq j} \sum_{x \in X^{(0)}} \sum_{(r,c) \in I_x} A_{k,j} \cdot \| B_{k,j} \cdot f_{(r,c)}(x, \theta_k) - f_{(r,c)}(x, \theta_j) \|_2^2$$
$$\text{feltéve, hogy: } \sum_{k \neq j} A_{k,j}^2 = \eta : \forall j \ ; \ B_{k,j}^T \cdot B_{k,j} = I : \forall k \neq j \tag{4.8}$$
$$\text{ahol } I \text{ az egységmátrix}$$

A $B_{k,j}^T \cdot B_{k,j} = I$ ortogonalitási feltétel biztosítja, hogy a transzformáció során a reprezentációk normája nem változik. Az optimális mátrix a 4.9 egyenlet megoldásából nyerhető, hiszen arra törekszünk, hogy a transzformációt követően a reprezentációk a lehető leghasonlóbbak legyenek, azaz belső szorzatukat maximalizáljuk. Mint azt az alábbi összefüggés is mutatja, a kereszthasznossági együtthatókhoz hasonlóan az átképző mátrixok meghatározása is kizárólag az objektumot tartalmazó jellemző vektorok figyelembe vételével történik.

$$\{B_{k,j}\}^* = \arg\max_{B_{k,j}} \sum_{k \neq j} \sum_{x \in X^{(0)}} \sum_{(r,c) \in I_x} \langle B_{k,j} \cdot f_{(r,c)}(x, \theta_k), f_{(r,c)}(x, \theta_j) \rangle \tag{4.9}$$

Az egyenletben a $\langle \cdot, \cdot \rangle$ operátor jelöli a vektorok belső szorzatát. A mátrix nyomának definícióját felhasználva a fenti kifejezés egy adott (k, j) klienspárra tovább alakítható:

$$\begin{aligned} &\sum_{x \in X^{(0)}} \sum_{(r,c) \in I_x} \langle B_{k,j} \cdot f_{(r,c)}(x, \theta_k), f_{(r,c)}(x, \theta_j) \rangle \\ &= \sum_{x \in X^{(0)}} \sum_{(r,c) \in I_x} trace(B_{k,j} \cdot f_{(r,c)}(x, \theta_k) \cdot f_{(r,c)}(x, \theta_j)^T) \\ &= trace(B_{k,j} \cdot \sum_{x \in X^{(0)}} \sum_{(r,c) \in I_x} f_{(r,c)}(x, \theta_k) \cdot f_{(r,c)}(x, \theta_j)^T) \end{aligned} \tag{4.10}$$

Legyen $R_{k,j} = \sum_{x \in X^{(0)}} \sum_{(r,c) \in I_x} f_{(r,c)}(x, \theta_k) \cdot f_{(r,c)}(x, \theta_j)^T$, továbbá legyenek $(\Gamma_{k,j} \cdot \Lambda_{k,j} \cdot \Phi_{k,j}^T)$ és $(U_{k,j} \cdot S_{k,j} \cdot V_{k,j}^T)$ a $B_{k,j}$ és $R_{k,j}$ mátrixok SVD felbontásai. Ez alapján a 4.10 egyenlet tovább írható a következő alakban:

$$\begin{aligned} &trace(B_{k,j} \cdot \sum_{x \in X^{(0)}} \sum_{(r,c) \in I_x} f_{(r,c)}(x, \theta_k) \cdot f_{(r,c)}(x, \theta_j)^T) = trace(B_{k,j} \cdot R_{k,j}) \\ &= trace((\Gamma_{k,j} \cdot \Lambda_{k,j} \cdot \Phi_{k,j}^T) \cdot (U_{k,j} \cdot S_{k,j} \cdot V_{k,j}^T)) \end{aligned} \tag{4.11}$$

Felhasználva a nyom operátor ciklikus tulajdonságát és az SVD felbontásból származó mátrixok valamint $B_{k,j}$ ortogonalitását, az optimalizációs probléma ekvivalens az alábbival:

$$\begin{aligned} &\arg\max_{\Gamma_{k,j}, \Lambda_{k,j}, \Phi_{k,j}} trace(S_{k,j} \cdot V_{k,j}^T \cdot \Gamma_{k,j} \cdot \Lambda_{k,j} \cdot \Phi_{k,j}^T \cdot U_{k,j}) \\ &\text{subject to: } \Gamma_{k,j}^T \cdot \Gamma_{k,j} = I, \Phi_{k,j}^T \cdot \Phi_{k,j} = I, \Lambda_{k,j} = I) \end{aligned} \tag{4.12}$$

Megjegyzés 1. Ha egy adott S diagonális mátrix elemei nem-negatívak, a $trace(S \cdot X) = \sum_i S_{(i,i)} \cdot X_{(i,i)}$ kifejezés értéke pontosan akkor lesz maximális, ha X főátlója mentén minden elem maximális értéket vesz fel. ▪

Megjegyzés 2. Mivel SVD felbontás során 2 ortogonális mátrxxal diagonalizálunk és ortogonális mátrixok szorzata szintén ortogonális, ezért $V_{k,j}^T \cdot \Gamma_{k,j} \cdot \Lambda_{k,j} \cdot \Phi_{k,j}^T \cdot U_{k,j}$ mátrix is ortogonális. ▪

Felhasználva 1 és 2 állításokat belátható, hogy azon $\Gamma_{k,j}$ és $\Phi_{k,j}$ mátrixokat keressük, amelyekre igaz, hogy $V_{k,j}^T \cdot \Gamma_{k,j} \cdot \Lambda_{k,j} \cdot \Phi_{k,j}^T \cdot U_{k,j} = I$. Mivel tudjuk, hogy $V_{k,j}$ és $U_{k,j}$ ortogonálisak, valamint $\Lambda_{k,j} = I$, ezért a keresett mátrixok: $\Gamma_{k,j} = V_{k,j}$, $\Phi_{k,j} = U_{k,j}$.

Tehát az optimális átképző mátrix egy lépésben, az alábbi módon számolható:

$$\begin{aligned} &B_{k,j} = V_{k,j} \cdot I \cdot U_{k,j}^T = V_{k,j} \cdot U_{k,j}^T \\ &\text{ahol } V_{k,j} \text{ és } U_{k,j} \text{ mátrixok } R_{k,j} \text{ SVD felbontásából kaphatóak meg} \end{aligned} \tag{4.13}$$

A reprezentációk közötti átképzés folyamatát megvalósító komponens könnyedén beilleszthető a 4.1 fejezet végén bemutatott lépés sorozatba, a következőképpen:

1. Minden kliens meghatározza a publikus, címkézetlen adathalmaz összes elemére a belső reprezentációkat ($f^{(r)}(x, \theta_k)$), illetve cellánként az objectness score-okat.

2. Az objectness score-okat a kliensek elküldik a szervernek, mely ezek alapján meghatározza az I_x index vektorokat $\forall x \in X^0$ -ra és elküldi minden résztvevőnek

3. A kliensek elküldik a szervernek a potenciálisan objektumokat tartalmazó cellák reprezentációs vektorait ($f_{(r,c)}(x, \theta_k)$, $\forall (r, c) \in I_x$)

4. A publikus reprezentációk alapján a szerver meghatározza klienspáronként a transzformációs mátrixokat és elvégzi az átképzést ($f_{(r,c)}^{(k,j)}(x)$ jelöli a k. kliens $f_{(r,c)}(x, \theta_k)$ jellemzővektorának j. kliens számára átképzett változatát):

$$\begin{aligned} &R_{k,j} = \sum_{x \in X^{(0)}} \sum_{(r,c) \in I_x} f_{(r,c)}(x, \theta_k) \cdot f_{(r,c)}(x, \theta_j)^T = U_{k,j} \cdot S_{k,j} \cdot V_{k,j}^T \\ &B_{k,j} = V_{k,j} \cdot U_{k,j}^T \\ &f_{(r,c)}^{(k,j)}(x) = B_{k,j} \cdot f_{(r,c)}(x, \theta_k) \ , \forall x \in X^{(0)}, \forall (r, c) \in I_x, \forall k, j \ , k \neq j \end{aligned} \tag{4.14}$$

5. A szerver frissíti a kereszthasznossági együtthatókat az átképzett reprezentációk hasonlósága alapján:

$$\{A_{j,k}\}^* = \frac{\sum_{x \in X^{(0)}} \sum_{(r,c) \in I_x} \langle f_{(r,c)}(x, \theta_j), f_{(r,c)}^{(k,j)}(x) \rangle}{\sqrt{(\sum_{u \neq v} (\sum_{x \in X^{(0)}} \sum_{(r,c) \in I_x} \langle f_{(r,c)}(x, \theta_u), f_{(r,c)}^{(v,u)}(x) \rangle)^2)/\eta}} \tag{4.15}$$

6. A frissített kereszthasznossági együtthatókat és a transzformált reprezentációkat felhasználva, minden kliens számára előállítjuk az aggregált, regularizáló reprezentációkat:

$$f_j^*(x)[(r,c)] = \begin{cases} f_{(r,c)}(x,\theta_j), \text{ ha } (r,c) \notin I_x \\ \sum_{k \neq j}(A_{k,j} \cdot f_{(r,c)}^{(k,j)}(x)), \text{ ha } (r,c) \in I_x \end{cases} \tag{4.16}$$

7. A kliensek pár epoch-on keresztül tanulják a számukra küldött regularizáló reprezentációt a 4.7 célfüggvény szerint.

Ezen lépéseket kiegészítve kliensenként a privát adatokon történő lokális, felügyelt tanulással kapjuk a fejezet elején leírt és a 4.1 ábrán is szemléltetett tanulási ciklus egy lehetséges megvalósítását. A teljes tanulási ciklushoz tartozó költségfüggvényt a j. kliens esetében a 4.17 egyenlet írja le. Ennek második tagja a kliens $X^{(j)}$ privát adathalmazán történő, SSE költségű lokális tanításáért felelős, a harmadik tag pedig a modellparamétereknek egy γ súlyozású L_2 regularizációja.

$$\begin{aligned} L(A,\theta_j) = &\sum_{k \neq j} \sum_{x \in X^{(0)}} \sum_{(r,c) \in I_x} A_{k,j} \cdot \|B_{k,j} \cdot f_{(r,c)}(x,\theta_k) - f_{(r,c)}(x,\theta_j)\|_2^2 \\ &+ SSE(\theta_j, X^{(j)}) + \gamma \cdot \|\theta_j\|_2^2 \\ &\text{feltéve, hogy: } \sum_{k \neq j} A_{k,j}^2 = \eta : \forall j \ ; \ B_{k,j}^T \cdot B_{k,j} = I : \forall k \neq j \end{aligned} \tag{4.17}$$

4.3. További komponensek

A teljesítmény növelése érdekében az előbbi fő komponensek mellett a teljes FedMOD keretrendszerbe további komponensek beillesztését is javasoljuk, ez az alfejezet ezek részletes bemutatását tartalmazza.

4.3.1. Finetuning

Legyen egy modell *reprezentációs csonkja* azoknak a rétegeknek a halmaza, amelyek a kollaboráció során használt reprezentációkat állítják elő, és hasonlóképpen legyenek a *taszkspecifikus rétegek* azok, amelyek a reprezentációból kimenetet állítanak elő. Közvetlenül a reprezentáció regularizáció után a taszkspecifikus rétegek még mindig az utolsó lokális tanulás utáni állapotot tükrözik, ellentétben a reprezentációs csonkkal. Az ilyen jellegű ellentmondás feloldása érdekében javasoljuk a *finomhangolás (fine-tuning)* alkalmazását a taszkspecifikus rétegeken a reprezentáció regularizálása után. A finomhangolás során a reprezentációs csonkot befagyasztjuk, és csak a felső, taszkspecifikus rétegeket tanítjuk privát adathalmazon, majd a tanulási ciklus utolsó lépése - a lokális felügyelt tanulás - a teljes hálón történik.

4.3.2. Intra-round rollback

Neurális hálózatok tanításánál bevált szokás a korai leállás (early stopping) alkalmazása a túltanulás elkerülése érdekében. Ezen komponens az early stopping-nak

egy federált multitaszk környezetben jól alkalmazható adaptációját, a javasolt *Rollback* módszert valósítja meg. A Rollback használata a kollaboratív tanulás során a következő lépések végrehajtását jelenti: minden tanulási ciklusban a modelleket kiértékejük a lokális validációs adathalmazokon a reprezentáció regularizáció, finetuning és lokális tanítás után. A tanulási ciklus végén visszatöltjük azon modellparamétereket, melyek az összes eddigi kiértékelés közül a legjobb eredményt adták. A Rollback fő célja, hogy, ha egy kliens számára bizonyos számú tanulási ciklus után már nem előnyös az együttműködés, akkor is megőrizheti a federált tanulás során elért legjobb teljesítményt szolgáló paramétereit anélkül, hogy ténylegesen kilépne az együttműködésből (ami negatív hatással lehet a többi résztvevőre). Továbbá kis adathalmazzal rendelkező, könnyen túltanuló klienseknél előfordulhat, hogy az első néhány tanulási ciklusban számukra nagyobb teljesítmény növekedést biztosít a reprezentáció regularizáció a lokális tanítás nélkül, mint ez utóbbi alkalmazásával. Ebben az esetben a klienskenek lehetőségük van a lokális tanítás eredményét figyelmen kívül hagyni az adott tanulási ciklusban.

4.3.3. Publikus adatok augmentációja

Az adatok augmentálása által jelentős teljesítmény növekedést tudunk elérni az objektum detekció problémáját tekintve, ezért a kollaboráció során is bevezetjük a publikus, címkézetlen adatok augmentálását (kontraszt, fényerő, szaturáció módosítása, képek eltolása, skálázása, forgatása). Az eljárás során a szerver minden tanulási ciklusban véletlenszerű transzformációkat hajt végre a publikus adathalmaz minden elemén, és az így módosított mintákat juttatja el a klienseknek (minden résztvevő számára ugyanazt). Egy adott kép augmentált verziója változik minden tanulási ciklusban, ezáltal segítve, hogy a kliensek transzformáció-invariáns kollaboratív reprezentáció tanulásra legyenek képesek.

4.3.4. Kliensenkénti adaptív kollaboráció

Ahogyan arról a fejezet elején is szó esett, az η hiperparaméter beállításával határozhatjuk meg a kollaboráció erejét a teljes eljárásban. Ezen paraméter értéke a 4.1 célfüggvényben szerplő feltétel szerint minden kliens esetében ugyanakkora. Gyakorlatban viszont jelentősen eltérhet, hogy az egyes résztvevőknek mennyire előnyös a kollaboráció, milyen súllyal érdemes nekik ebben résztvenni. Ilyen megfontolásból javaslunk egy egyszerű komponenst, mely az egyes klienseknél a kollaboráció eredményessége szerint a tanulási folyamat során módosítja a hozzájuk rendelt η_j paramétereket. A 4.1 egyenletben is szereplő, kereszthasznossági együtthatókat szabályozó kényszer tehát a következőképp módosul: $\sum_{k \neq j} A_{k,j}^2 = \eta_j : \forall j$. A komponens alkalmazása az alábbi lépések végrehajtását jelenti:

1. Bármely j. klienshez rendelt η_j paramétert ugyanazon η értékkel inicializáljuk.

2. Minden tanulási ciklus végén a validációs halmazon kiértékeljük a modelleket és, amennyiben valamely kliens teljesítménye javult az előző tanulási ciklushoz képest, a hozzá tartozó η_j paraméter értékét a következőképp módosítjuk: $\eta_j \leftarrow \eta_j \cdot m$, ahol $m > 1$ előre definiált multiplikátor. Ha valamely modell teljesítménye romlik, akkor esetében az $\eta_j \leftarrow \eta_j / m$ műveletet hajtjuk végre

Látható, hogy ezen komponens a kollaboráció eredményességének függvényében képes kliensenként személyre szabottan változtatni a reprezentáció regularizáció súlyát. Belátható, hogy az η paraméterek ilyen jellegű módosítása ekvivalens azzal, mintha a reprezentáció regularizációhoz tartozó iteratív optimalizációs eljárás (esetünkben Adam) tanulási tényezőjét módosítanánk ugyanezen $m, 1/m$ multiplikátorokkal. Ezért az egyszerűség miatt az η paraméterek helyett a tanulási tényezők módosítását implementáltam a fenti lépések mentén.

A két központi (súlyozott reprezentáció regularizáló és reprezentációk között átképző) és ezen négy opcionális komponens összessége alkotja a javasolt FedMOD keretrendszert. Az összes komonenst tartalmazó eljárás pszeudokódot a 3 algoritmus mutatja be.

3. Algorithm FedMOD

Bemenet: Publikus adathalmaz $X^{(0)}$, privát adatok X_i, random inicializált θ_i modellek , $i = 1, ..K$, ahol K a résztvevők száma, $\eta_1 = \eta_2 = .. = \eta_K = \eta$, m

Kimenet: Betanított modellek: $\{\theta_i\}$

$best\_val\_mAP_i \leftarrow 0, \forall i = 1, 2, ..., K$

Minden kliens konvergenciáig tanítja saját θ_i modelljét a privát X_i adathalmazán

for r = 1, 2, ..., rounds **do**

  $round\_mAP_i \leftarrow best\_val\_mAP_i, \forall i \in \{1, ..., K\}$

  $X^{(*)} \leftarrow augment(X^{(0)})$

  for $x \in X^{(*)}$ **do**

    $I_x^k = get\_cells\_containing\_object(x, \theta_k), \forall k \in \{1, ..., K\}$

    $I_x = \bigcup_{k=1}^{K} I_x^k$

  end for

  for $j, k = 1, 2, ..., K, j \neq k$ **do**

    $U_{k,j} \cdot S_{k,j} \cdot V_{k,j}^T \leftarrow SVD(\sum_{x \in X^{(*)}} \sum_{(r,c) \in I_x} f_{(r,c)}(x, \theta_k) \cdot f_{(r,c)}(x, \theta_j)^T)$

    $B_{k,j} \leftarrow V_{k,j} \cdot U_{k,j}^T$

    $f_{(r,c)}^{(k,j)}(x) \leftarrow B_{k,j} \cdot f_{(r,c)}(x, \theta_k)$, $\forall x \in X^{(*)}, \forall (r,c) \in I_x$

    $\{A_{j,k}\}^* = \frac{\sum_{x \in X^{(*)}} \sum_{(r,c) \in I_x} \langle f_{(r,c)}(x, \theta_j), f_{(r,c)}^{(k,j)}(x) \rangle}{\sqrt{(\sum_{u \neq v} (\sum_{x \in X^{(*)}} \sum_{(r,c) \in I_x} \langle f_{(r,c)}(x, \theta_u), f_{(r,c)}^{(v,u)}(x) \rangle)^2)/\eta_j}}$

  end for

  for $(j, x) \in \{1, 2, ..., K\} \times X^{(*)}$ **do**

    $f_j^*(x)[(r,c)] = \begin{cases} f_{(r,c)}(x, \theta_j), \text{ ha } (r,c) \notin I_x \\ \sum_{k \neq j} (A_{k,j} \cdot f_{(r,c)}^{(k,j)}(x)), \text{ ha } (r,c) \in I_x \end{cases}$

  end for

  $\{\theta_j\}^* \leftarrow \arg\min_{\theta_j} \sum_{x \in X^{(*)}} \|f^{(r)}(x, \theta_j) - f_j^*(x)\|_2^2, \forall j \in \{1, ..K\}$

  $round\_mAP_i \leftarrow Rollback(\theta_i, round\_mAP_i), \forall i \in \{1, ..K\}$

  Kliensek taszkspecifikus rétegeinek **finomhangolása** a privát adatokon

  $round\_mAP_i \leftarrow Rollback(\theta_i, round\_mAP_i), \forall i \in \{1, ..K\}$

  Lokális tanítás SSE költség alapján a privát adatokon

  for k = 1,2, .., K **do**

    $round\_mAP_k \leftarrow Rollback(\theta_k, round\_mAP_k)$

    if $round\_mAP_k \geq best\_val\_mAP_k$ **then**:

      $\eta_k \leftarrow \eta_k \cdot m$

      $best\_val\_mAP_k \leftarrow round\_mAP_k$

    else:

      $\eta_k \leftarrow \eta_k / m$

    endif

  end for

  $\theta_i \leftarrow load\_saved\_model(i), \forall i \in \{1, ..K\}$

Rollback$(\theta_i, round\_mAP_i)$**:**

  $mAP \leftarrow evaluate(\theta_i, X_i)$

  if $mAP \geq round\_mAP_i$ **then**:

    $save\_model(\theta_i)$

  endif

  return $\max(round\_mAP_i, mAP)$

5. fejezet

Eredmények

Ebben a fejezetben először bemutatom a FedMOD eljárás kiértékeléséhez használt környezetet (hiperparaméterek, kliensek közötti taszk felosztás) melyet a futási eredmények részletezése követ.

5.1. Hiperparaméter optimalizáció

A FedMOD eljárásnak a következő hiperparaméterei vannak: η, disztillációs epoch-ok száma, finetuning epoch-ok száma, lokális epoch-ok száma, lokális tanításnál használt tanulási tényező, reprezentáció regularizációnál használt tanulási tényező, batch méret. Ezen paraméterek optimális értékeit a PascalVOC adathalmazara az Optuna [2] automatikus hiperparaméter optimalizáló eljárás segítségével határoztam meg. Egy adott paraméter konstrukciót a résztvevő kliensek validációs mAP értékeinek átlaga alapján minősítettem. 60 különböző FedMOD futtatás alapján a legjobb 5 értéket eredményező paraméter értékeket az 5.1 táblázat tartalmazza. Ezek közül az első sorban levő értékeket használtam fel a FedMOD eljárás kiértékelésénél mindhárom taszkfelosztás (lásd 5.2) esetében.

5.1. táblázat. Optimális hiperparaméterek

mAP	η	**#Dist epoch**	**#Finetune epoch**	**#Local epoch**	**Repr LR**	**Local LR**	**Batch size**
0.518	1.23	2	2	7	7.8E-5	1.68E-4	16
0.516	1.99	2	5	7	8.8E-5	1.64E-4	16
0.514	0.47	2	1	3	6.7E-5	1.73E-4	16
0.513	0.79	2	1	7	6.4E-5	1.51E-4	16
0.512	1.11	3	2	7	2.5E-5	2.25E-4	16

5.2. Taszk felosztások

Ahogyan a 3.2.3.2 fejezetben már említésre került, 3 különböző taszk felosztás mentén vizsgáljuk a tudás transzfer mértékét egy 3 résztvevős kollaboratív tanulási környezetben. A 5.2 táblázat mutajta a konkrét Pascal VOC taszkok (objektum osztályok) felosztását a kliensek között úgy, hogy (a 3.2.3.1 fejezetben bemutatott

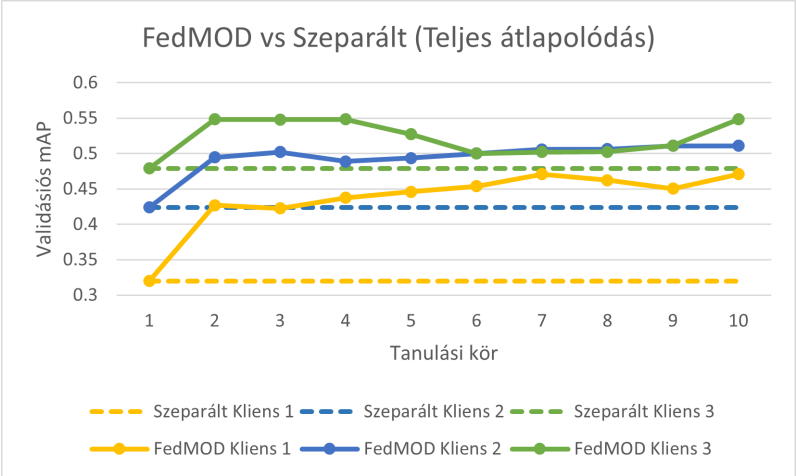
eredményeket felhasználva) minden felosztás esetében a *Kliens 1* és *Kliens 2* lehetőleg korreláló taszkokat tanuljanak. Látható, hogy a változó hasonlóságok mellett kihasználjuk a FedMOD eljárás azon előnyét is, hogy a résztvevő kliensek akár eltérő számú taszkkal is rendelkezhetnek.

5.2. táblázat. Három eltérő taszkfelosztás a három résztvevő között

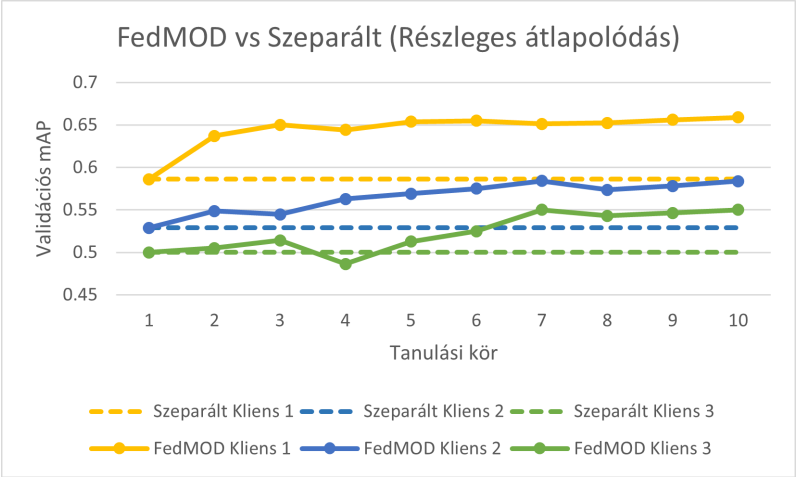
	Kliens 1	**Kliens 2**	**Kliens 3**
Teljes átlapolódás	Person Sofa Chair Table Bicycle Motorbike Cat Dog Horse Cow	Person Sofa Chair Table Bicycle Motorbike Cat Dog Horse Cow	Plane Bird Boat Bus Car Sheep Train TVmonitor
Részleges átlapolódás	Train Sofa Car Bicycle Dog	Chair Car Bus Motorbike Cat	Plane Bird Horse Cow Sheep TVmonitor
Nincs átlapolódás	Person Chair Cow Bicycle Cat	Sofa Table Sheep Motorbike Dog Car	Plane Bird Boat Bottle Bus Horse Pottedplant Train TVmonitor

5.3. Nem kooperatív tanítás és FedMOD összehasonlítása

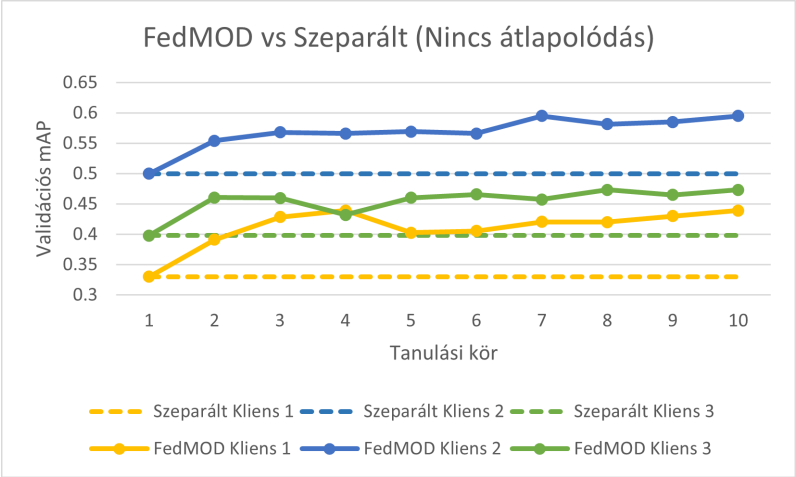
A javasolt FedMOD eljárást összehasonlítottam a kliensek teljesen szeparált (nem kooperatív) tanításával is annak érdekében, hogy megvizsgáljam a kollaboráció hozzáadott értékét. Az összeahsonlítás eredménye a 5.1 ábrán látható mindhárom taszk felosztás esetén. A vizsgálat során először mindhárom klienst konvergenciáig tanítottam a privát adathalmazaikon (ennek eredményét az ábrákon szaggatott vonalak jelölik), ezt követően kezdődött köztük a kollaboráció 10 tanulási cikluson keresztül.



(a) Teljes átlapolódás



(b) Részleges átlapolódás



(c) Nincs átlapolódás

5.1. ábra. FedMOD és nem-kooperatív (szeparált) tanítás teljesítményének összehasonlítása

Látható, hogy mindhárom fajta taszk felosztás esetében eredményesnek bizonyult a kollaboráció, főként a hasonló (vagy esetenként átfedő) taszkokat tanuló első és második kliens tekintetében. A teljesen áltapolódó felosztásnál megfigyelhetjük,

hogy a 4. tanulási ciklust követően a harmadik kliens teljesítménye nem javult tovább, a kollaborációból nem profitált ezt követően, az eljárás végén a Rollback komponens miatt lett a teljesítménye az addigi maximum. Ezen eredmény is igazolja, hogy annál előnyösebb a kollaboráció egy adott résztvevő számára, minél hasonlóbb reprezentációkat igénylő taszkot tanul a többi klienssel. A grafikonok többsége továbbá azt is mutatja, hogy az első néhány tanulási ciklus már elég a kliensek számára a kollaborációból kinyerhető teljesítmény növekedés közel maximumához.

5.4. FedAvg alapú federált multitaszk objektum detekciós eljárás

Legjobb tudomásom szerint a szakirodalomban jelenleg nem elérhető a javasolt FedMOD eljáráson kívül más, federált multitaszk objektum detekció tanulására alkalmas algoritmus, ezért ezen dolgozatban javaslom a, már ismertetett, FedAvg eljárásnak is egy multitaszk adaptációját, mely technikailag hasonló problémák megoldására alkalmas, mint a FedMOD. A javasolt FedAvg alapú eljárás fő lépései az alábbiak:

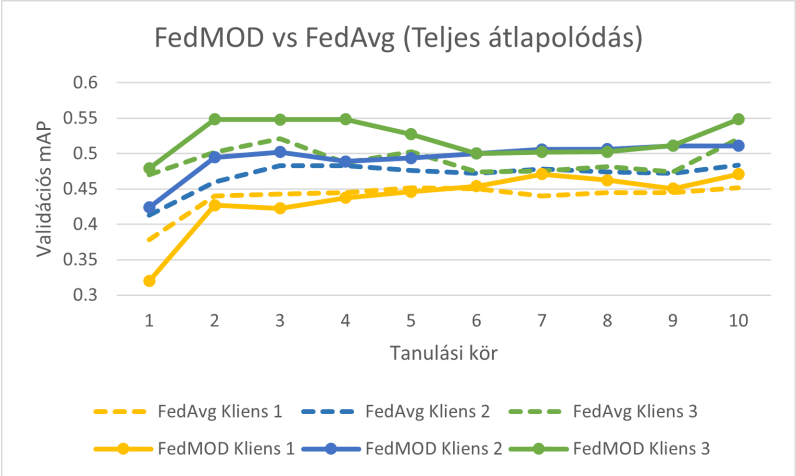
0. Kliensek előtanítása konvergenciáig a privát adathalmazaikon.
1. Minden kliens elküldi a szervernek a reprezentációs csonkjukat alkotó paramétereit[1]
2. A szerver átlagolja a kliensektől érkező paramétereket és visszaküldi minden résztvevőnek az aggregált reprezentációs csonkot, mellyel a kliensek felülírják saját, aktuális csonkjukat.
3. Reprezentációs csonk befagyasztása és felső, taszkspecifikus rétegek finomhangolása
4. Teljes architektúra lokális, felügyelt tanítása a privát adathalmazon
5. Tanulási cikluson belül elért legjobb eredményt adó paraméterek visszatöltése kliensenként
6. Folytatás az 1. lépéssel a meghatározott számú tanulási cikluson keresztül

A felvázolt eljárás előnye a FedMOD-hoz képest, hogy a tanítás kisebb számításigényű, ezáltal gyorsabb, valamint nagy számú résztvevőnél is hatékonyan alkalmazható, hiszen nem szükséges a klienspárok reprezentációinak átképző mátrixait kiszámítani. Hátrányai viszont, hogy egyáltalán nem veszi figyelembe a különböző mértékű hasonlóságokat a kliensek között. Továbbá nem alkalmazható olyan esetekben, amikor eltérő méretű/komplexitású reprezentációs csonkkal rendelkezik minden kliens, végül pedig a megosztott model paraméterek visszafejtéséből egy támadó információhoz juthat a kliensek privát adathalmazairól [10], mely komoly privacy kérdéseket vet fel.

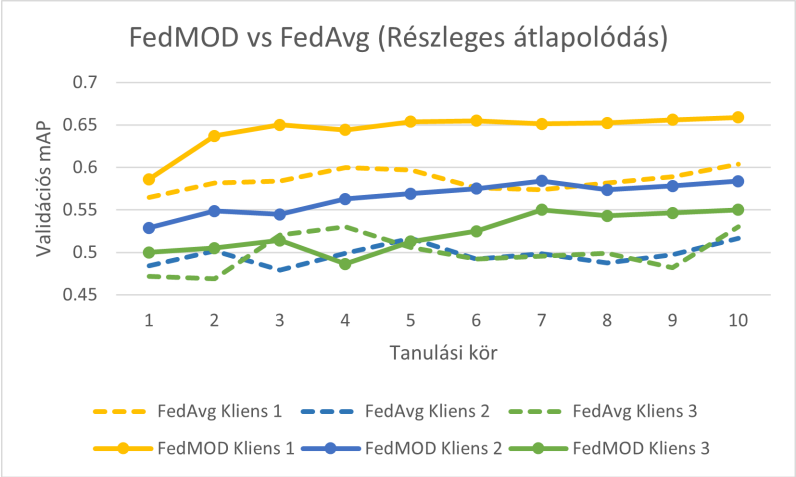
[1]Ennek feltétele, hogy minden kliens pontosan ugyanolyan architektúrával számolja a reprezentációkat, ami jelentős megkötés a FedMOD-nál említett architektúrális szabadsághoz képest

A FedMOD és a FedAvg alapú eljárás összehasonlításának eredményét az 5.2 ábra mutatja mindhárom fajta taszkfelosztásra. Látható, hogy a teljes átlapolódásnál közel azonos validációs mAP értékeket eredményezett a két eljárás, hiszen a megegyező taszkokat tanuló résztvevők esetében a reprezentációs csonkok átlagolása előnyös. Az eltérő taszkokat tanuló harmadik kliens esetében a FedAvg alapú módszernél is megfigyelhetjük a teljesítmény esést a harmadik tanulási ciklust követően. Amikor a kliensek már nem teljesen megegyező, ám korreláló taszkokat tanulnak (b) és c) grafikonok), a kereszthasznosságokkal súlyozott reprezentáció regularizáció előnye egyértelműen megmutatkozik az egyszerű átlagolással szemben. Látható, hogy 10 tanulási ciklus után mindkét taszkfelosztás esetében mindhárom kliens magasabb validációs mAP értéket ér el, mint a FedAvg alapú módszert alkalmazva.

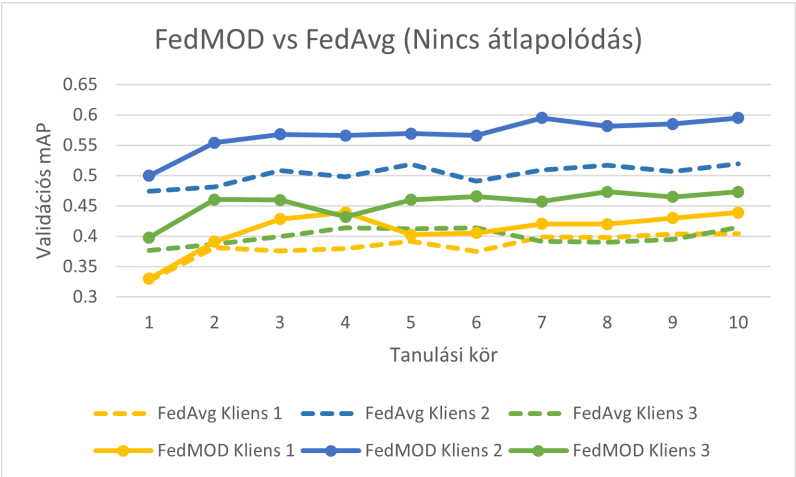
Az 5.3 ábra mutatja a FedMOD (folytonos vonalak) és FedAvg alapú módszer (szaggatott vonalak) alkalmazása során közvetlenül a reprezentáció regularizációt (disztilláció), illetve a reprezentációs csonkok átlagolását követően az egyes kliensek validációs mAP értékét teljes taszkátlapolódás esetén. Látható, hogy míg a FedAvg során alkalmazott átlagolás a kliensek többségénél közel nulla mAP értéket eredményezett, addig a FedMOD esetében a súlyozott reprezentáció regularizáció minden tanulási ciklusban magas értéket tudott fenntartani, esetenként az előző tanulási ciklus maximmális pontosságát is képes volt meghaladni a teljesen megegyező taszokat tanuló kliensek esetében. Ezen eredmény igazolja, hogy valóban a korreláló taszkokat tanuló kliensek számára kölcsönösen előnyös tudás transzfer jön létre a FedMOD alkalmazása során, nem csak a tanulási cikluson belüli lokális, felügyelt tanulás felelős a teljesítmény növekedéséért. Az ábrán az is látható, hogy a többiektől eltérő taszkokat tanuló harmadik kliens teljesítménye a FedMOD folyamat elején jelentősen alacsonyabb, mint a másik két kliensé, majd a tanulási ciklusok előrehaladtával a résztvevők olyan reprezentációt tanulnak, mely ezen kliens számára is egyre hasznosabb. Ezt igazolja a növekvő trend a harmadik klienshez tartozó függvényben.



(a) Teljes átlapolódás

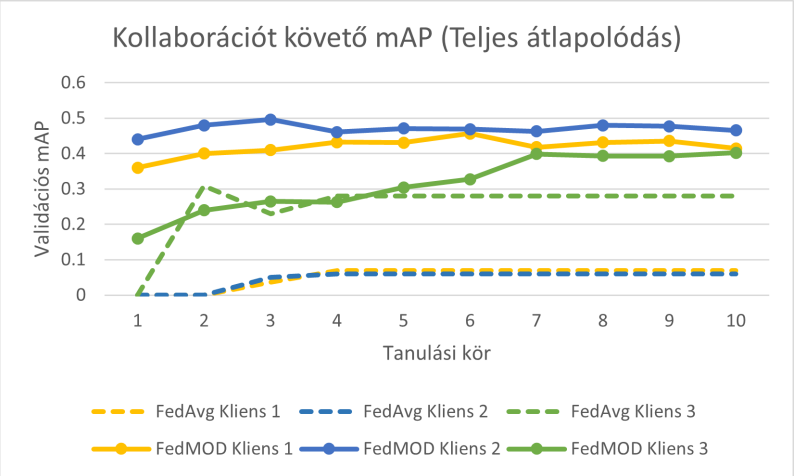


(b) Részleges átlapolódás



(c) Nincs átlapolódás

5.2. ábra. FedMOD és FedAvg alapú eljárás teljesítményének összehasonlítása előtanított kliensek esetén



5.3. ábra. Validációs mAP értékek közvetlenül a tudásmegosztás után a FedMOD, illetve a FedAvg alapú eljárás esetében

5.5. Kereszthasznossági együtthatók

A FedMOD eljárás kiértékelése során vizsgáltam, hogy a kereszthasznossági együtthatók értéke mennyire tükrözi a 3.2.3.1 fejezetben is bemutatott, taszkok között mért hasonlósági viszonyokat. A 5.3, 5.4, 5.5 táblázatok mutatják rendre a teljesen, részlegesen és egyáltalán nem átlapolódó taszk felosztásokra a kereszthasznossági mátrixokat az előtanítást követő első (vastagított értékek) és az utolsó (záróeles értékek) tanulási ciklusban. Látható, hogy a kollaboráció kezdetekor mindhárom esetben a *Kliens 1* és *Kliens 2* közti kerszthasznosság a legmagasabb, ami jól korrelál a felosztásokban szereplő hasonlósági viszonyokkal. A mátrixokban valamilyen szintű szimmetria is megfigyelhető, az együthatók alapján a *Kliens 3* számára a *Kliens 1* és *Kliens 2* szinte teljesen ugyanannyira bizonyul hasznosnak és fordítva: *Kliens 3* is ugyanolyan mértékben hasznos a másik két résztvevőnek. Ezen eredmény is alátámasztja az együtthatók robosztus viselkedését. A kollaboráció utolsó tanulási ciklusában mért értékek azt mutatják, hogy a ciklusok előrehaladtával az együtthatók egyre inkább kiegyenlítetté válnak, mivel a disztilláció során a résztvevők olyan reprezentációkat tanulnak, mely egymás számára egyre hasznosabb és könnyebben átképzehető. Ezen jelenséget mutatja az 5.3 ábrán is a három klienshez tartozó függvény közeledése.

5.3. táblázat. Kereszthasznossági mátrix az első (vastagított értékek) és az utolsó (zárójeles értékek) tanulási ciklusban - teljes átfedés

	Kliens1	Kliens2	Kliens3
Kliens1	-	**0.7750** (0.7369)	**0.6320** (0.6760)
Kliens2	**0.7743** (0.7329)	-	**0.6328** (0.6804)
Kliens3	**0.7063** (0.7029)	**0.7079** (0.7115)	-

5.4. táblázat. Kereszthasznossági mátrix az első (vastagított értékek) és az utolsó (zárójeles értékek) tanulási ciklusban - részleges átfedés

	Kliens1	Kliens2	Kliens3
Kliens1	-	**0.7248** (0.7169)	**0.6890** (0.6971)
Kliens2	**0.7281** (0.7104)	-	**0.6855** (0.7038)
Kliens3	**0.7105** (0.7005)	**0.7037** (0.7137)	-

5.5. táblázat. Kereszthasznossági mátrix az első (vastagított értékek) és az utolsó (zárójeles értékek) tanulási ciklusban - nincs átfedés

	Kliens1	Kliens2	Kliens3
Kliens1	-	**0.7309** (0.7264)	**0.6825** (0.6873)
Kliens2	**0.7305** (0.7118)	-	**0.6829** (0.7024)
Kliens3	**0.7067** (0.6921)	**0.7075** (0.7218)	-

5.6. Belső reprezentációk vizualiálása

Annak érdekében, hogy jobban beleláthassunk a kollaboráció működésébe és átfogóan elemezhessük annak hatását, a belső reprezentációk térbeli eloszlását is vizsgáltam az alábbi lépések mentén:

1. A nem kooperatív előtanítás végén mindhárom résztvevő meghatározta a publikus halmaz minden mintájának potenciálisan objektumokhoz tartozó reprezentációs vektorait ($f_{(r,c)}(x, \theta_k), \forall (r,c) \in I_x, \forall x \in X^{(0)}$) és ezen 1024 elemű vektorokból álló listát elmentettem

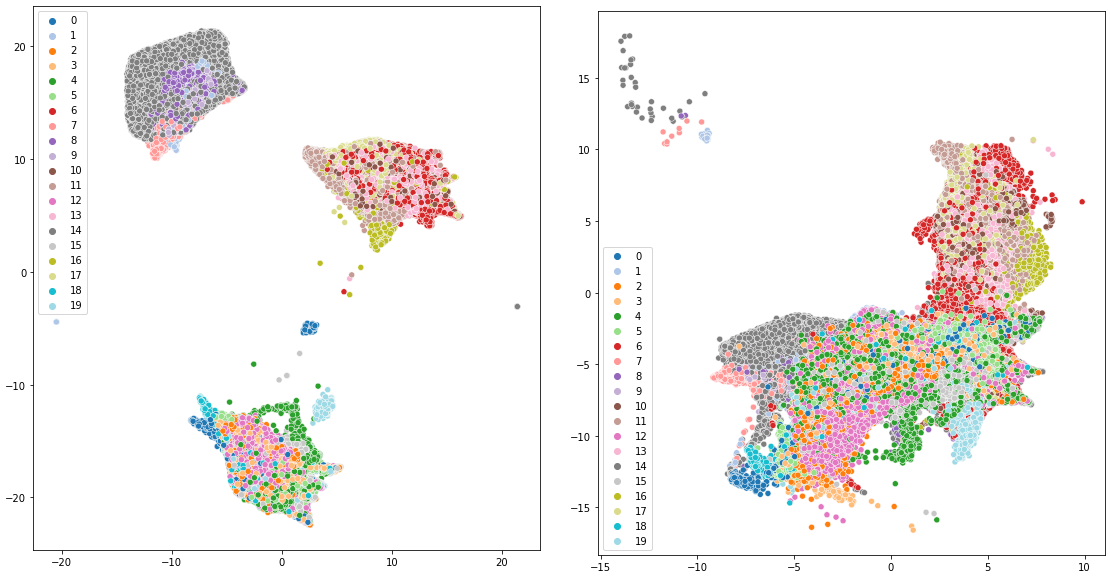
2. Hasonlóan a nem kooperatív esethez, a FedMOD és FedAvg eljárás végén is elmentettem a reprezentációs vektorok halmazát.

3. Az így kapott 1024 dimenziós vektorokkal egy PacMAP [46] távolságtartó projekciós modellt tanítottam, mely minden reprezentációs vektort 2 dimenzióra vetített le. A projekció eredménye a 5.4 ábrán látható rendre a nem kooperatív

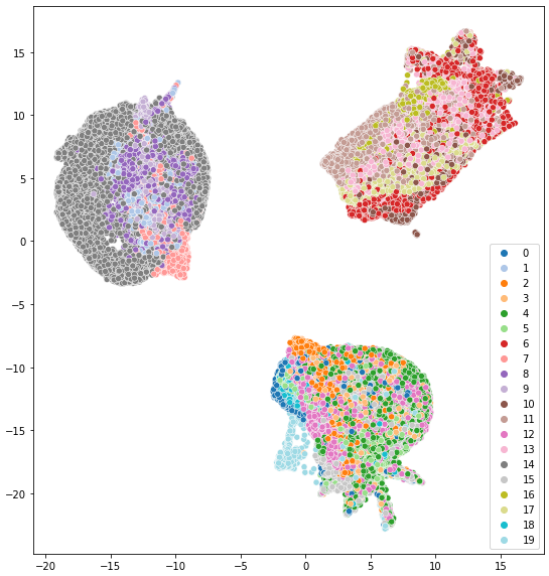
tanulás, FedAvg alapú kollaboráció és FedMOD eljárás végén az átlapolódás nélküli taszkfelosztás esetében.

A levetített reprezentációk az ábrán aszerint vannak színezve, hogy az egyes kliensek az általuk tanult taszkok közül milyen típusú objektumhoz rendelték hozzá az adott jellemzővektort. Mivel a publikus adathalmaz nem címkézett, ezért ezen predikciók helyességét nem tudjuk validálni, viszont a helyes működést erősíti meg az egyszínű, homogén tartományok létrejötte, amelyek mindhárom ábrán fellelhetőek. Megfigyelhető továbbá, hogy a 3 résztvevő reprezentációi a FedMOD esetében különülnek el a legjobban egymástól, illetve az egy adott klienshez tartozó leképezésben több homogén régiót is felfedezhetünk, ami a predikció javulására is utalhat a kollaborációnak köszönhetően. A résztvevők önálló reprezentációkat képesek fenntartani, úgy, hogy közben egymás számára hasznos módon át tudják ezeket képezni. A nem kooperatív tanítás esetén amellett, hogy kevesebb homogén, egyszínű régió fedezhető fel, a három résztvevő reprezentációinak egy része nem is rendelhető egyértelműen egy adott klienshez. A FedAvg alapú kollaboráció végén pedig látható, hogy az átlagolásnak köszönhetően mindhárom kliens reprezentációja ugyanazon altérben található, az egyes modellek nem tudtak önálló reprezentációkat kialakítani, ami a feladataik eltérésének mértékétől függően hátrányos lehet a teljesítmény tekintetében. Ennek ellenére a predikciós osztályok szerinti homogén csoportosulások itt is megfigyelhetőek, ami a helyes működésre utal.

Az előbbihez hasonló módon vizualizáltam a reprezentációk közötti átképzés folyamatát is. Az 5.5 ábra szemlélteti a *Kliens 1* (pöttyök) és *Kliens 2* (keresztek) publikus reprezentációit egy adott tanulási ciklusban közvetlenül a *Kliens 2* vektorainak a *Kliens 1* alterébe történő átképzése előtt (bal oldali ábra) és után (jobb oldali ábra). Látható, hogy a transzformáció során a két modell reprezentációi ugyanazon altérbe képződtek, sőt, az algoritmus taszkonként is képes volt megfelelő tartományba transzformálni: adott (prediktált) címkével rendelkező vektorok az átképzés után is ugyanazon osztály által meghatározott homogén klaszterbe képződtek, mint amilyen klaszterhez előzőleg is tartoztak. Továbbá ezen ábra azt is mutatja, hogy a modellek olyan önálló reprezentációkat tudtak kialakítani, amelyek a megfelelő lineáris transzformációk sorozatával egymásba kis hibával átképezhetőek. Ezen példa esetében látható, hogy a *Kliens 2* reprezentációinak ortogonális, lineáris transzformációja valóban átképez a *Kliens 1* reprezetációinak alterébe, melyhez a PacMAP-es dimenzió redukció előtti reprezentációs vektorok forgatása elégséges. Az eljárás sikeresen megtalálta ezen elemi transzformációk optimális paramétereit az átképzési hiba minimalizálásához.

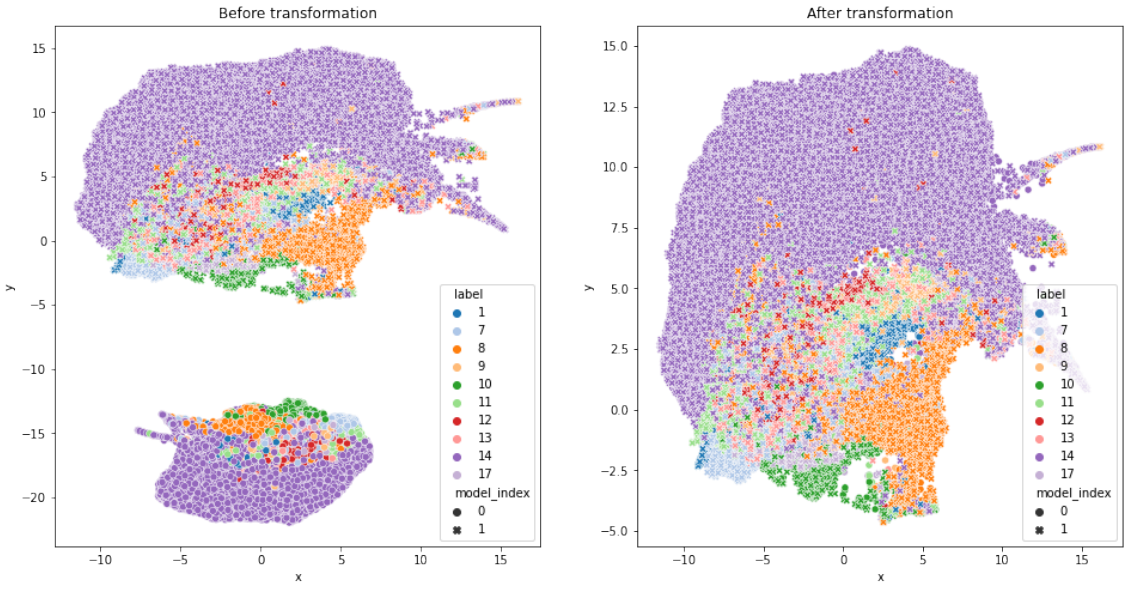


(a) Nem kooperatív tanítás **(b)** FedAvg



(c) FedMOD

5.4. ábra. A publikus adathalmaznak kliensek által meghatározott reprezentációjának 2 dimenzióra vetített változata a három eljárás (nem kooperatív tanítás, FedAvg, FedMOD) futtatását követően



5.5. ábra. Kliens 1 és Kliens 2 publikus reprezentációinak 2 dimenzióra vetített változata az átképzés előtt közvetlenül (bal oldali ábra) és utána (jobb oldali ábra)

5.7. Tudás transzfer mértéke

Multitaszk tanulásnál felmerül a kérdés, hogy mennyire hatékony a tudás transzfer, milyen jósági metrikával tudjuk ezt mérni, mi ennek a maximuma? A FedMOD esetében az alábbi vizsgálatot eszközöltem: Legyen egy tetszőleges taszkfelosztás esetén $0 \leq s \leq 1$ és $0 \leq F \leq 1$ rendre a szeparált tanulással és a FedMOD eljárással elérhető, egymással korreláló taszkokat tanuló klienseken (*Kliens 1* és *Kliens 2*) átlagolt validációs mAP érték abban az esetben, amikor minden kliens (az előző vizsgálatokhoz hasonlóan) 2500 mintából álló privát adathalmazzal rendelkezik. Legyen továbbá $0 \leq S \leq 1$ a s-nek megfelelő érték abban az esetben, amikor *Kliens 1* és *Kliens 2* privát, címkézett adathalmazait egybeöntjük[2] és ezen mindkét modellt szeparáltan tanítjuk. Általános esetben $s \leq F \leq S$, valamint nulla hatékonyságú transzfer esetén $s = F$, 100%-os hatékonyságnál pedig $F = S$, hiszen akkor maximális a transzfer, ha ugyanazt a hatást éri el, mintha a korreláló taszkokhoz tartozó adathalmazokból együttesen tanulhatnának a modellek (ami gyakorlatban a privacy miatt nem lehetséges). Az 5.6 táblázat tartalmazza s, F és S értékeit, valamint a transzfer hatásfokát ($\frac{F-s}{S-s}$) százalékban mérve mindhárom fajta taszkfelosztásnál. Látható, hogy mindhárom esetben hatékony volt a tudás transzfer, a teljes, illetve részleges átlapolódásnál a számolt hatékonyság meghaladja az 50%-ot, azaz a FedMOD-al olyan hatást sikerült elérni, mely ekvivalens a kollaboráló kliensek privát adathalmazának akár több mint 50%-os bővítésével.

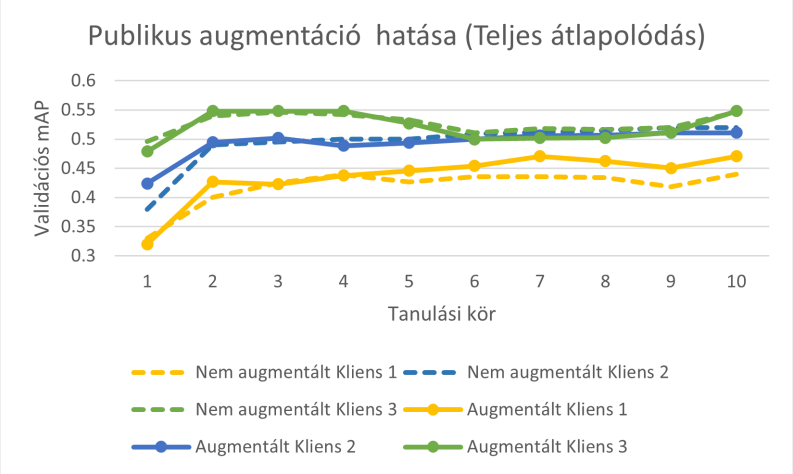
[2] A nem teljesen átlapolódó esetekben a taszkok különbsége miatt ez technikailag azt jelenetette, hogy az adott kliensek privát adathalmazának méretét megdupláztam

5.6. táblázat. Tudás transzfer mértéke

	Teljes átlapolódás	Részleges átlapolódás	Nincs átlapolódás
Szeparált 2500 minta (s)	0.45	0.594	0.467
Szeparált 5000 minta (S)	0.537	0.641	0.58
FedMOD 2500 minta (F)	0.516	0.62	0.518
Transzfer ($\frac{F-s}{S-s}$)	**57%**	**57.9%**	**45%**

5.8. Publikus adathalmaz augmentálásának hatása

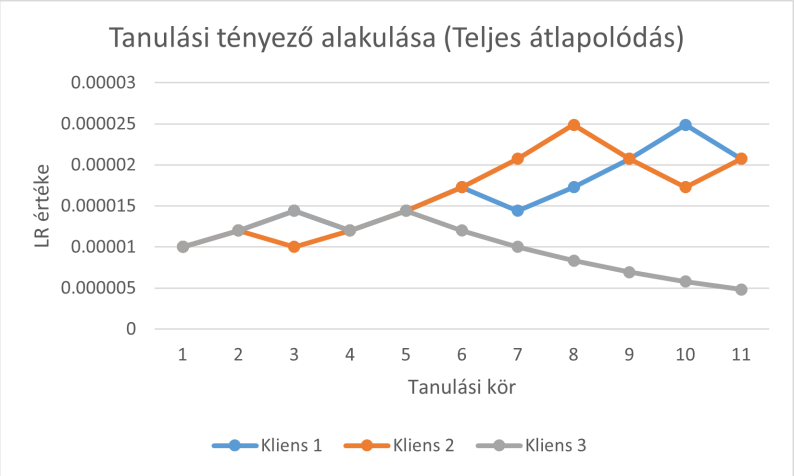
Az 5.6 ábrán látható a FedMOD teljesítménye tanulási ciklusonként a publikus adathalmaz véletlenszerű augmentálásával (folytonos vonalak) és anélkül (szagatott vonalak). Látható, hogy a két esetben a validációs mAP értékek közel azonosak, a *Kliens 1* résztvevőnél látható egyértelmű javulás. Hipotézisunk alapján annak oka, hogy elenyésző a publikus adathalmaz augmentálásának pozitív hatása a következő: a publikus augmentáció célja, hogy a résztvevők a kollaboráció során is olyan reprezentációkat tudjanak tanulni, melyek invariánsak a publikus adathalmaz transzformációira. Ilyen jellegű invarianciát viszont minden résztvevő tanul a privát adathalmazának augmentálásával is. Sejtésünk alapján a módszer olyan esetekben bizonyul hatásosabbnak, amikor a publikus adathalmazon olyan jellegű transzformációkat is végzünk, amelyeket egy vagy több résztvevő nem alkalmazott a privát adathalmazán. Ezen hipotézisnek vizsgálata a jövőbeli kutatásom egyik célja.



5.6. ábra. FedMOD teljesítménye a publikus adatok augmentálásával (folytonos vonalak) és nélküle (szaggatott vonalak)

5.9. Kollaboráció erősségének szabályozása

A 4.3.4 fejezetben bemutatott, kliensenkénti kollaboráció erősségét szabályozó komponens helyes működését a következő kísérlettel vizsgáltam: a teljesen átlapolódó tazkfelosztás esetében kezdeti 10^{-5} értékű reprezentáció regularizációs tanulási tényezővel és $m = 1.2$ multiplikátorral megfigyeltem a kliensenkénti tanulási tényező alakulását, ami az előbbiekben leírtak alapján ekvivalens a kliensekhez rendelt η paraméterek adaptációjával. Egy kezdeti előtanítást követő 10 tanulási ciklusos kollaboráció során a három résztvevő disztillációhoz használt tanulási tényezőjének alakulását az 5.7 ábra szemlélteti. A várakozásoknak megfelelően az látszik, hogy az azonos taszkokat tanuló *Kliens 1* és *Kliens 2* tanulási tényezője növekvő trend mentén alakul, ezáltal felgyorsítva a modelloptimalizációk konvergenciáját. Ezzel ellentétben megfigyelhető, hogy a *Kliens 3* számára a kollaboráció az 5. tanulási ciklus után már nem előnyös, ezt azonosította és tanulási tényezőjét folyamatosan csökkenti. Ezáltal képes redukálni a többi résztvevő taszkjainak eltérése miatt fennálló esetleges negatív tudás transzfert a kollaboráció második felében.

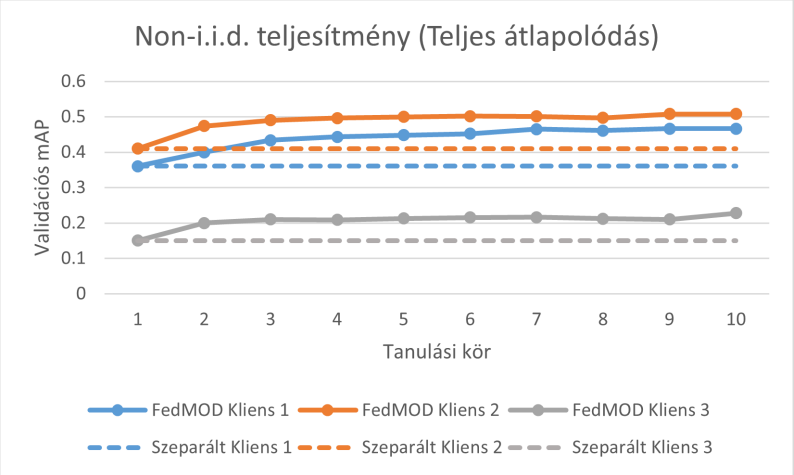


5.7. ábra. Reprezentáció regularizációnál használt tanulási tényező alakulása teljesen átlapolódó taszkfelosztás esetében

5.10. Kollaboráció non-iid környezetben

Gyakorlatban a kollaborációban részvevő vállalatok nagy valószínűséggel más-más forrásokból gyűjtenek privát adathalmazokat, melyek többnyire más eloszlásból mintavételezett elemeket tartalmaznak (non-i.i.d. adathalmaz), viszont a kollaborációnak ebben az esetben is hatékonyan kell működnie. Ilyen valós környezet imitálása érdekében a következő vizsgálatot eszközöltem: Az előbbiekhez hasonló három résztvevős tanulási környezetben *Kliens 1* és *Kliens 2* ugyanazon taszkokat tanulja, de eltérő privát adathalmazokon, melyek a PascalVOC részhalmazai. A harmadik kliens szintén hasonló taszkokat tanul (Bicycle, Car, Dog, Cat, Bus, Chair), viszont az ő privát adathalmaza a COCO egy részhalmaza. A publikus minták mindegyike ezen vizsgálat során a PascalVOC adathalmazból származik. Ezen környezetben futtattam a három kliens teljesen szeparált tanítását konvergenciáig, majd a FedMOD

eljárást alkalmazva 10 tanulási cikluson keresztül kollaboratív tanulás történt. Az 5.8 ábra mutatja az elért validációs mAP értékeket a szeparált tanítás végén (szagatott vonalak) és a FedMOD eljárást alkalmazva (folytonos vonalak). Megfigyelhető, hogy a kollaboráció ebben a non-i.i.d. környezetben is hatékony volt, teljesítmény növekedés történt mindhárom kliensnél a szeparált tanításhoz viszonyítva. Ezen eredmény megerősíti, hogy a FedMOD eljárás képes eltérő eloszlású adatokból is a többi résztvevő számára hasznos információt kinyerni és átadni, mindezt úgy, hogy közben a privát adathalmazról vagy a modell paraméterekről semmilyen érzékeny információt nem oszt meg.



5.8. ábra. FedMOD teljesítménye a szeparált tanításhoz viszonyítva non-i.i.d. környezetben, amikor Kliens 1 és Kliens 2 a PascalVOC adathalmazból, míg Kliens 3 a COCO adathalmazból tanult lokálisan

5.11. Skálázhatóság

A javasolt FedMOD eljárás skálázhatóságát a futásidők tekintetében vizsgáltam meg. Az alábbi eredményeket Nvidia V100-as videókártyákon történő többszöri futtatásból összegeztem. Az 5.7 táblázat a szeparált tanítás futásidejét tartalmazza három klienses környezetben különböző számú, ResNET18 fölé illesztett konvolúciós réteg, batch méretet és epochszámot használva. Az egyes kliensek privát adathalmazai 2500 képből álltak. Látható, hogy a futásidőt leginkább az epochszám befolyásolja, a batch méret és konvolúciós rétegek számának változtatásának hatása az epochszámhoz képest jóval kevesebb.

5.7. táblázat. Nem kooperatív tanítás futásideje különböző hiperparaméterek mellett

Teljes futásidő	Batch méret	Epoch szám	Felső konvolúciós rétegek száma
73 min	16	39	3
136 min	32	59	4
41 min	64	22	2
77 min	16	42	2
118 min	64	57	3

A szeparált tanítást követően a FedMOD alapú kollaboráció egy tanulási ciklusának futásideje az 5.8 táblázatban látható értékek szerint alakult a különböző disztillációs-, finetuning- és lokális epochszám esetében. 2500 képből álló publikus adathalmazon egy epoch-nyi reprezentáció regularizáció átlagosan 20 másodpercet, míg a lokális tanítás ugyanekkora adathalmazon 18 másodpercet vett igénybe kliensenként, tehát ezen két lépés erőforrásigénye közel azonos.

5.8. táblázat. FedMOD futásideje különböző hiperparaméterek mellett

Futásidő/ciklus	Disztillációs epoch szám	Finetuning epoch szám	Lokális epoch szám
28.9 min	2	2	7
33.8 min	4	2	6
20.4 min	2	1	3
41.6 min	7	3	4
25.7 min	2	3	4

Megjegyzem, hogy a futtatások során a résztvevők egyszálú környezetben, nem párhuzamos módon tanultak, a FedMOD eljárást használva megvalósítható gyakorlati szerver-kliens környezetben a kliens oldalon végzett műveletek (regularizáció, finetuning, lokális tanulás) párhuzamosítása, ami jelentősen csökkenti a futásidőt. Továbbá fontos megjegyezni, hogy a kliensek közötti átképző mátrixokat úgy tároljuk, hogy annak mérete nem függ a publikus adathalmaz méretétől, csupán a jellemzővektorok dimenzionalitásától, ezáltal minimalizálva a kollaborációból adódó plusz memóriaigényt és a publikus adathalmaz méretét tekintve jól skálázódik az algoritmus. Ezzel szemben a kliesnek számával négyzetesen nő a szerver oldali erőforrásigény, ilyen kompromisszum viszont elengedhetetlen a kliensenként személyre szabott, hatékony tudásmegosztáshoz. Továbbá a FedMOD eljárást leginkább kevesebb, megbízható kliens (nagyvállalat, intézmény) közötti kollaboráció létrehozására fejlesztettük, ezen módszer esetében nem lényeges szempont, hogy a kliensek számában is jól skálázódjon. Ennek ellenére szerver oldalon megfelelő erőforrás rendelkezésre állása esetén Big Data megoldásokat felhasználva (pl. Spark [47]) lehetséges a műveletek párhuzamosítása és a futásidő további redukálása nagyságrendileg több kliens esetén.

6. fejezet

Konklúzió

Ezen dolgozatban egy új keretrendszert javasoltam a federatív multitaszk objektum detekció megoldására, mely a legszigorúbb adatvédelem mellett is képes hatékony kollaborációt megvalósítani az egymással korreláló taszkokat tanuló résztvevők (tipikusan nagyvállalatok, intézmények) között, ezáltal növelve lokális modelljeik teljesítményét. A javasolt FedMOD eljárás legfőképp olyan gyakorlati problémáknál alkalmazható, ahol szigorú privacy védi az amúgy is kis mennyiségű, jó minőségű, felcímkézett adathalmazokat és az ezeken tanuló modelleket, ezáltal korlátozva a lokálisan elérhető maximális teljesítményt. Konrét példa az ilyen jellegű alkalmazásokra a több intézményben elosztottan zajló, mély tanulás alapú orvosi döntéstámogatás, gyárépületekben haszált önvezető targoncák és hasonló gépek, vagy gyártósor mellett termékek minőségelleőrzését végző automata rendszerek kollaboratív tanítása, stb. A FedMOD eljárás a kollaboráció során kizárólag publikusan elérhető, címkézetlen adatmintákat használ, ezáltal is kompenzálva a kis mennyiségben elérhető, felcímkézett adathalmazok problematikáját. A javasolt keretrendszer képes:

- A kliensek közti kereszthasznosságok becslésére, ezáltal kihasználva, hogy hasonló taszkokat tanuló kliensek többet profitálhatnak az egymással történő kollaborációból
- Kereszthasznosságokkal súlyozott reprezentáció regularizációra, mely a szigorú adatvédelmet megőrizve valósítja meg a tudás transzfert a résztvevők között
- Kliens reprezentációk közötti átképzésre, amely biztosítja, hogy az egyes kliensek tudásukat egymás számára a lehető leghasznosabb módon közvetítsék.

A keretrendszer továbbá tartalmaz komponenseket, melyek a reprezentáció tanulás utáni finomhangolásért, a publikus adathalmaz szerver oldali augmentációjáért és a kliensenként szabályozható kollaborációs együttható (η) adaptációjáért felelős.

A PascalVOC és COCO adathalmazokon elvégzett kiértékelés alapján a FedMOD-al jelentős javulást érhetünk el a szeparált tanuláshoz viszonyítva, akár non-i.i.d környezetben is. Ezen dolgozatban javasoltam továbbá az ismert FedAvg eljárásnak egy multitaszk környezetben objektum detekció megoldására alkalmazható változatát. A numerikus eredmények azt mutatták, hogy a FedMOD egyértelműen túlteljesítette a FedAvg alapú eljárást mindhárom vizsgált taszkfelosztás esetében. Továbbá fontos megjegyezni, hogy a FedMOD-al ellentétben a FedAvg alapú módszert csak teljesen megegyező felépítésű reprezentációs csonkkal rendelkező kliense-

ken alkalmazhatjuk, valamint a modell paraméterek megosztása által a privacy is sérülhet.

A kiértékelés során a tudás transzfer hatékonyságát is számszerűsítettem, mely alapján a FedMOD-al olyan hatást sikerült elérni, mely ekvivalens a kollaboráló kliensek privát adathalmazának (taszk felosztástól függően) akár több mint 50%-os bővítésével. Összegzésképpen az eredmények azt mutatják, hogy a javasolt FedMOD keretrendszer hatékonyan alkalmazható a federált multitaszk objektum detekció megoldására olyan gyakorlati problémák esetében is, ahol kritikus a szigorú adatvédelem a résztvevők adathalmazainak és modelljeinek tekintetében egyaránt.

Köszönetnyilvánítás

Ezt a munkát az Emberi Erőforrások Minisztériuma támogatta az Új Nemzeti Kiválósági Program keretében (ÚNKP-22-2-I-BME-228).

Továbbá szeretnék köszönetet mondani konzulensemnek, Hadházi Dánielnek, aki mindig készségesen segített a munkámban, és egyedi ötleteivel nagyban hozzájárult tudásom gyarapodásához és ezen kutatómunka létrejöttéhez.

Irodalomjegyzék

[1] *Adaptive Industrial Robots Using Machine Vision*, ASME International Mechanical Engineering Congress and Exposition konferenciasorozat, Volume 2: Advanced Manufacturing. köt., 2018. 11.
URL https://doi.org/10.1115/IMECE2018-86720. V002T02A093.

[2] Takuya Akiba – Shotaro Sano – Toshihiko Yanase – Takeru Ohta – Masanori Koyama: Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (konferenciaanyag). 2019.

[3] Mahbub Ul Alam – Rahim Rahmani: Federated semi-supervised multi-task learning to detect covid-19 and lungs segmentation marking using chest radiography images and raspberry pi devices: An internet of medical things application. *Sensors*, 21. évf. (2021) 15. sz. ISSN 1424-8220.
URL https://www.mdpi.com/1424-8220/21/15/5025.

[4] Abdullatif Albaseer – Bekir Sait Ciftler – Mohamed Abdallah – Ala Al-Fuqaha: Exploiting unlabeled data in smart cities using federated edge learning. In *2020 International Wireless Communications and Mobile Computing (IWCMC)* (konferenciaanyag). 2020, 1666–1671. p.

[5] Attila Kádár, Dániel Hadházi: Fedlinked: A client-wise distilled representation basedsemi-supervised collaborative multi-task learning scheme. https://tdk.bme.hu/VIK/ViewPaper/FedLinked-Kliensek-kozotti-reprezentacio.

[6] Edwin V Bonilla – Kian Chai – Christopher Williams: Multi-task gaussian process prediction. In J. Platt – D. Koller – Y. Singer – S. Roweis (szerk.): *Advances in Neural Information Processing Systems* (konferenciaanyag), 20. köt. 2008, Curran Associates, Inc. URL https://proceedings.neurips.cc/paper/2007/file/66368270ffd51418ec58bd793f2d9b1b-Paper.pdf.

[7] Stephen Boyd – Lieven Vandenberghe: *Convex optimization*. 2004, Cambridge university press.

[8] Alexander Buslaev – Vladimir I. Iglovikov – Eugene Khvedchenya – Alex Parinov – Mikhail Druzhinin – Alexandr A. Kalinin: Albumentations: fast and flexible image augmentations. *ArXiv e-prints*, 2018.
URL https://doi.org/10.3390%2Finfo11020125.

[9] Michael Crawshaw: Multi-task learning with deep neural networks: A survey. *ArXiv*, abs/2009.09796. évf. (2020).

[10] Dimitar I. Dimitrov – Mislav Balunović – Nikola Konstantinov – Martin Vechev: Data leakage in federated averaging, 2022.
URL https://arxiv.org/abs/2206.12395.

[11] Kaiwen Duan – Song Bai – Lingxi Xie – Honggang Qi – Qingming Huang – Qi Tian: Centernet: Keypoint triplets for object detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (konferenciaanyag). 2019, 6568–6577. p.

[12] Mark Everingham – Luc Gool – Christopher K. Williams – John Winn – Andrew Zisserman: The pascal visual object classes (voc) challenge. *Int. J. Comput. Vision*, 88. évf. (2010. jun) 2. sz., 303–338. p. ISSN 0920-5691.
URL https://doi.org/10.1007/s11263-009-0275-4. 36 p.

[13] Federated multi-task learning. https://papers.nips.cc/paper/7029-federated-multi-task-learning.pdf. Accessed: 2020-05-27.

[14] Yabo Fu – Yang Lei – Tonghe Wang – Walter J. Curran – Tian Liu – Xiaofeng Yang: A review of deep learning based methods for medical image multi-organ segmentation. *Physica Medica*, 85. évf. (2021), 107–122. p. ISSN 1120-1797. URL https://www.sciencedirect.com/science/article/pii/S1120179721001848.

[15] Ross Girshick: Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)* (konferenciaanyag). 2015, 1440–1448. p.

[16] Ross Girshick – Jeff Donahue – Trevor Darrell – Jitendra Malik: Rich feature hierarchies for accurate object detection and semantic segmentation, 2013. URL https://arxiv.org/abs/1311.2524.

[17] Kaiming He – Xiangyu Zhang – Shaoqing Ren – Jian Sun: Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

[18] Geoffrey Hinton – Oriol Vinyals – Jeff Dean: Distilling the knowledge in a neural network, 2015. URL https://arxiv.org/abs/1503.02531.

[19] Sohei Itahara – Takayuki Nishio – Yusuke Koda – Masahiro Morikura – Koji Yamamoto: Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data. *IEEE Transactions on Mobile Computing*, 2021., 1–1. p.

[20] Deepthi Jallepalli – Navya Chennagiri Ravikumar – Poojitha Vurtur Badarinath – Shravya Uchil – Mahima Agumbe Suresh: Federated learning for object detection in autonomous vehicles. In *2021 IEEE Seventh International Conference on Big Data Computing Service and Applications (BigDataService)* (konferenciaanyag). 2021, 107–114. p.

[21] Peter Kairouz – H. Brendan McMahan – Brendan Avent – Aurélien Bellet – Mehdi Bennis – Arjun Nitin Bhagoji – Keith Bonawitz – Zachary Charles – Graham Cormode – Rachel Cummings – Rafael G. L.

D'Oliveira – Salim El Rouayheb – David Evans – Josh Gardner – Zachary Garrett – Adrià Gascón – Badih Ghazi – Phillip B. Gibbons – Marco Gruteser – Zaid Harchaoui – Chaoyang He – Lie He – Zhouyuan Huo – Ben Hutchinson – Justin Hsu – Martin Jaggi – Tara Javidi – Gauri Joshi – Mikhail Khodak – Jakub Konečný – Aleksandra Korolova – Farinaz Koushanfar – Sanmi Koyejo – Tancrède Lepoint – Yang Liu – Prateek Mittal – Mehryar Mohri – Richard Nock – Ayfer Özgür – Rasmus Pagh – Mariana Raykova – Hang Qi – Daniel Ramage – Ramesh Raskar – Dawn Song – Weikang Song – Sebastian U. Stich – Ziteng Sun – Ananda Theertha Suresh – Florian Tramèr – Praneeth Vepakomma – Jianyu Wang – Li Xiong – Zheng Xu – Qiang Yang – Felix X. Yu – Han Yu – Sen Zhao: Advances and open problems in federated learning, 2019.

[22] Alex Krizhevsky – Ilya Sutskever – Geoffrey E Hinton: Imagenet classification with deep convolutional neural networks. In F. Pereira – C.J. Burges – L. Bottou – K.Q. Weinberger (szerk.): *Advances in Neural Information Processing Systems* (konferenciaanyag), 25. köt. 2012, Curran Associates, Inc. URL `https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf`.

[23] Attila Kádár – Dániel Hadházi: Fedlinked: A client-wise distilled representation based semi-supervised collaborative multitask learning scheme. In *2022 International Joint Conference on Neural Networks (IJCNN)* (konferenciaanyag). 2022, 1–8. p.

[24] Daliang Li – Junpu Wang: Fedmd: Heterogenous federated learning via model distillation, 2019. URL `https://arxiv.org/abs/1910.03581`.

[25] Tian Li – Anit Kumar Sahu – Manzil Zaheer – Maziar Sanjabi – Ameet Talwalkar – Virginia Smith: Federated optimization in heterogeneous networks, 2018. URL `https://arxiv.org/abs/1812.06127`.

[26] You Li – Javier Ibanez-Guzman: Lidar for autonomous driving: The principles, challenges, and trends for automotive lidar and perception systems. *IEEE Signal Processing Magazine*, 37. évf. (2020. jul) 4. sz., 50–61. p.
URL `https://doi.org/10.1109%2Fmsp.2020.2973615`.

[27] Tsung-Yi Lin – Priya Goyal – Ross Girshick – Kaiming He – Piotr Dollár: Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)* (konferenciaanyag). 2017, 2999–3007. p.

[28] Tsung-Yi Lin – Michael Maire – Serge Belongie – Lubomir Bourdev – Ross Girshick – James Hays – Pietro Perona – Deva Ramanan – C. Lawrence Zitnick – Piotr Dollár: Microsoft coco: Common objects in context, 2014.
URL `https://arxiv.org/abs/1405.0312`.

[29] Sulin Liu – Sinno Jialin Pan – Qirong Ho: Distributed multi-task relationship learning. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17 konferenciasorozat. New York, NY, USA, 2017, Association for Computing Machinery, 937–946. p. ISBN 9781450348874. URL `https://doi.org/10.1145/3097983.3098136`. 10 p.

[30] Wei Liu – Dragomir Anguelov – Dumitru Erhan – Christian Szegedy – Scott E. Reed – Cheng-Yang Fu – Alexander C. Berg: Ssd: Single shot multibox detector. In Bastian Leibe – Jiri Matas – Nicu Sebe – Max Welling (szerk.): *ECCV (1)*, Lecture Notes in Computer Science konferenciasorozat, 9905. köt. 2016, Springer, 21–37. p. ISBN 978-3-319-46447-3. URL http://dblp.uni-trier.de/db/conf/eccv/eccv2016-1.html#LiuAESRFB16.

[31] Yang Liu – Anbu Huang – Yun Luo – He Huang – Youzhi Liu – Yuanyuan Chen – Lican Feng – Tianjian Chen – Han Yu – Qiang Yang: Fedvision: An online visual object detection platform powered by federated learning, 2020. URL https://arxiv.org/abs/2001.06202.

[32] Zewei Long – Liwei Che – Yaqing Wang – Muchao Ye – Junyu Luo – Jinze Wu – Houping Xiao – Fenglong Ma: Fedsemi: An adaptive federated semi-supervised learning framework, 2020.

[33] Brendan McMahan – Eider Moore – Daniel Ramage – Seth Hampson – Blaise Aguera y Arcas: Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh – Jerry Zhu (szerk.): *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research konferenciasorozat, 54. köt. Fort Lauderdale, FL, USA, 2017. Apr 20-22., PMLR, 1273–1282. p. URL http://proceedings.mlr.press/v54/mcmahan17a.html.

[34] Meghan Han, Michael Sarazen: The yolov3 object detection network is fast! https://medium.com/syncedreview/the-yolov3-object-detection-network-is-fast-fcceae0ab650, Megtekintve: 2022. 10. 28.

[35] Ishan Misra – Abhinav Shrivastava – Abhinav Gupta – Martial Hebert: Cross-stitch networks for multi-task learning. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (konferenciaanyag). 2016, 3994–4003. p.

[36] Papers with Code: Object detection on pascal voc 2007. https://paperswithcode.com/sota/object-detection-on-pascal-voc-2007, Megtekintve: 2022. 10. 27.

[37] Vishwa S Parekh – Shuhao Lai – Vladimir Braverman – Jeff Leal – Steven Rowe – Jay J Pillai – Michael A Jacobs: Cross-domain federated learning in medical imaging, 2021. URL https://arxiv.org/abs/2112.10001.

[38] Farheen Ramzan – Muhammad Usman Khan – Asim Rehmat – Sajid Iqbal – Tanzila Saba – Amjad Rehman – Zahid Mehmood: A deep learning approach for automated diagnosis and multi-class classification of alzheimer's disease stages using resting-state fmri and residual neural networks. *Journal of Medical Systems*, 44. évf. (2019. 12).

[39] Joseph Redmon – Santosh Divvala – Ross Girshick – Ali Farhadi: You only look once: Unified, real-time object detection, 2015.
URL https://arxiv.org/abs/1506.02640.

[40] Shaoqing Ren – Kaiming He – Ross Girshick – Jian Sun: Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15 konferenciasorozat. Cambridge, MA, USA, 2015, MIT Press, 91–99. p. 9 p.

[41] Shunli Ren – Siheng Chen – Wenjun Zhang: Collaborative perception for autonomous driving: Current status and future trend. In Zhang Ren – Mengyi Wang – Yongzhao Hua (szerk.): *Proceedings of 2021 5th Chinese Conference on Swarm Intelligence and Cooperative Control* (konferenciaanyag). Singapore, 2023, Springer Nature Singapore, 682–692. p. ISBN 978-981-19-3998-3.

[42] Sambasivarao. K: Non-maximum suppression (nms). `https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c/`, Megtekintve: 2022. 10. 28.

[43] Mohammad Javad Shafiee – Brendan Chywl – Francis Li – Alexander Wong: Fast yolo: A fast you only look once system for real-time embedded object detection in video, 2017. URL `https://arxiv.org/abs/1709.05943`.

[44] Antti Tarvainen – Harri Valpola: Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems* (konferenciaanyag), 30. köt. 2017, Curran Associates, Inc. URL `https://proceedings.neurips.cc/paper/2017/file/68053af2923e00204c3ca7c6a3150cf7-Paper.pdf`.

[45] Shanfeng Wang – Qixiang Wang – Maoguo Gong: Multi-task learning based network embedding. *Frontiers in Neuroscience*, 13. évf. (2020. 01).

[46] Yingfan Wang – Haiyang Huang – Cynthia Rudin – Yaron Shaposhnik: Understanding how dimension reduction tools work: An empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. 2020. URL `https://arxiv.org/abs/2012.04456`.

[47] Matei Zaharia – Reynold S. Xin – Patrick Wendell – Tathagata Das – Michael Armbrust – Ankur Dave – Xiangrui Meng – Josh Rosen – Shivaram Venkataraman – Michael J. Franklin – Ali Ghodsi – Joseph Gonzalez – Scott Shenker – Ion Stoica: Apache spark: A unified engine for big data processing. *Commun. ACM*, 59. évf. (2016. oct) 11. sz., 56–65. p. ISSN 0001-0782.
URL `https://doi.org/10.1145/2934664`. 10 p.

[48] Yu Zhang – Qiang Yang: A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 2021., 1–1. p.