Federalno učenje u uređajima na rubu temeljeno na grupama

Krklec, Andrija

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1573

FEDERALNO UČENJE U UREĐAJIMA NA RUBU TEMELJENO NA GRUPAMA

Andrija Krklec

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1573

FEDERALNO UČENJE U UREĐAJIMA NA RUBU TEMELJENO NA GRUPAMA

Andrija Krklec

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

ZAVRŠNI ZADATAK br. 1573

Pristupnik:	**Andrija Krklec (0036540539)**
Studij:	Elektrotehnika i informacijska tehnologija i Računarstvo
Modul:	Računarstvo
Mentor:	izv. prof. dr. sc. Igor Čavrak

Zadatak: **Federalno učenje u uređajima na rubu temeljeno na grupama**

Opis zadatka:

Proučiti osnovna svojstva federalnog učenja i njegovu primjenu u arhitekturama sustava rub-oblak, s naglaskom na performance učenja i kvalitetu rezultirajućeg dijeljenog modela. Istražiti utjecaj heterogenost performanci rubnih čvorova, mrežne komunikacije i nedostupnosti rubnih čvorova tijekom procesa učenja. Također proučiti utjecaj različitosti distribucija podataka rubnih čvorova na performance sustava, ispitati mogućnosti automatskog odvajanja i združivanja dijeljenih modela s obzirom na sličnosti i razlike u distribucijama podataka između članova grupa rubnih čvorova. Predložiti postupak združivanja i odvajanja modela temeljen na nenadziranom učenju, ispitati svojstva postupka korištenjem javno dostupnih skupova podataka.

Rok za predaju rada: 14. lipnja 2024.

SADRŽAJ

1. Uvod

Analiza i obrada podataka iznimno je važna u modernim aplikacijama kao što su prepoznavanje objekata u slikama, transkripcija audio zapisa i sl. Ti podaci se većinom nalaze na uređajima krajnjih korisnika, računalima, laptopima i mobitelima, no to mogu biti i razni IoT (engl. *Internet of Things*) uređaji poput temperaturnih senzora. Klasičan pristup analiziranju tih podataka zahtjeva da se oni "sirovi" pošalju na neki centralni uređaj koji će ih obraditi, primjenom strojnog učenja i odgovarajućeg modela. Problem je što se kod tog pristupa ne čuva privatnost vlasnika podataka i neskalabilan je u smislu da će vrlo brzo zagušiti komunikacijski kanal te se takvi načini prikupljanja nastoje izbjeći.

Novi pristup federalnog učenja nalaže da se podaci ne šalju u izvornom obliku. Uređaji korisnika lokalno treniraju zajednički model i na centralni server šalju samo novoizračunate težine modela nakon odgovarajućeg broja epoha. Na taj način nije ugrožena privatnost podataka i puno bolje skalira s povećanjem broja krajnjih uređaja i količine podataka.

Ovaj pristup također nije savršen. Javljaju se problemi poput kada trenirati model na krajnjem uređaju, to sigurno ne bi bilo poželjno kad korisnik aktivno koristi uređaj ili ako nije spojen na izvor napajanja. Još jedan problem je odabir čvorova, krajnjih uređaja koji će sudjelovati u procesu. Ako su podaci korisnika jako različiti, treniranje centralnog modela trajat će znatno dulje i možda neće biti moguće postignuti željene rezultate.

U radu će se ponuditi moguća rješenja na navedene poteškoće, primjenom algoritama grupiranja na klijentske čvorove. Time se pokušava postići upravo da su slični čvorovi u istim grupama, koje će zasebno provoditi federalno učenje i rezultate slati centralnom serveru. Također će se ispitivati razne situacije koje mogu nastati u praksi, npr. ispad krajnjeg čvora — osoba je ugasila mobilni uređaj, ili smanjena brzina prijenosa parametara na centralni čvor — osoba je povezana na sporu mrežu.

2. Federalno učenje

2.1. Ideja i motivacija

Potreba za raspodijeljenim sustavom strojnog učenja javlja se prirodno. Jedan od primjera je autocorrect kod tipkanja, odnosno automatsko ispravljanje gramatičkih grešaka. Koncept je prikupiti velike količine podataka, riječi koje osobe upisuju putem tipkovnice na svom mobitelu i šalju ostalima. Zatim treniramo model nad tim prikupljenim podacima koji će moći s određenom preciznošću ispraviti riječ koja mu se preda na ulaz. Kad bi podatke o nizovima znakova koje korisnici uređaja tipkaju slali u izvornom obliku na centralni uređaj, pojavila bi se velika opasnost curenja privatnih informacija korisnika, poput brojeva kreditnih kartica ili lozinki.

Još jedan primjer gdje bi takav sustav bio od koristi su medicinska istraživanja — detektor tumora temeljen na umjetnoj inteligenciji. Za takav projekt potrebna je velika baza medicinskih podataka stvarnih pacijenata, koje je nemoguće prikupiti u sirovom obliku jer su privatni i zaštićeni raznim regulacijama, poput GDPR[1]. Argument da se podaci ipak mogu koristiti ako se maknu ime i datum rođenja također nije dovoljan za osiguravanje privatnosti pacijenata.

Federalno učenje je paradigma kojom se nastoje spriječiti povrede privatnosti korisnika. Pokazuje se da se njime mogu postići performanse usporedive s onima kod centraliziranog strojnog učenja nad sirovim podacima i puno bolje nego u slučaju kad treniramo model nad jednim izoliranim skupom podataka, u prethodnom slučaju kad bi bolnica trenirala model nad podacima samo svojih pacijenata [12].

Još jedna od prednosti federalnog učenja je skalabilnost procesa. Mrežom je potrebno prenositi samo centralni model sa servera na klijentske čvorove i novoizračunate težine s čvorova natrag na centralni server. Takav način prijenosa zahtijeva puno manje mrežnog prometa i rasterećuje centralni čvor, no klijenti preuzimaju ulogu treniranja lokalnih modela, potencijalno resursno zahtjevan zadatak.

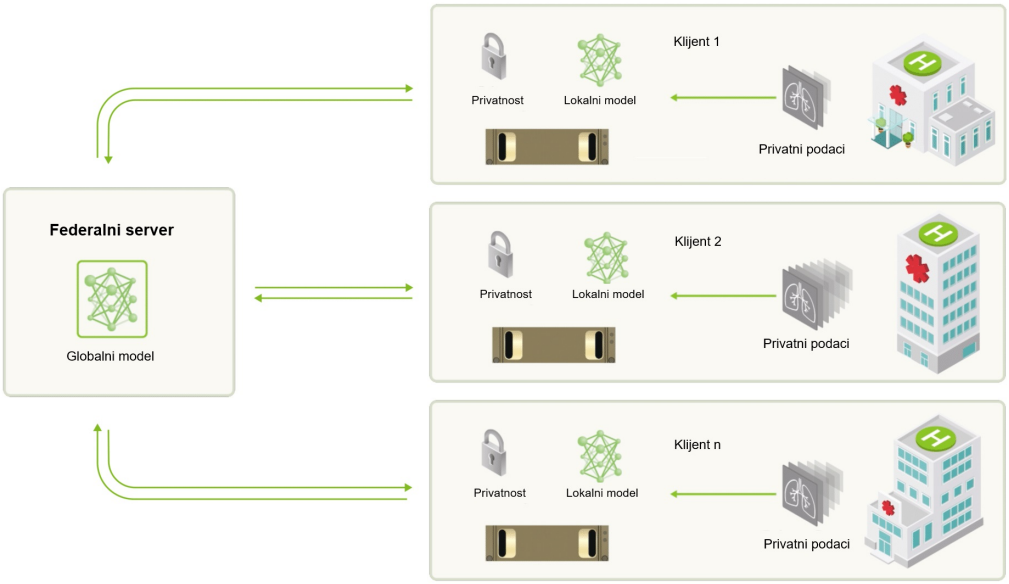
[1] General Data Protection Regulation

2.2. Opis procesa

Federalno učenje provodi se u pet koraka [6]:

- inicijalizacija parametara
- prijenos globalnog modela na klijente
- lokalno treniranje modela
- prijenos izmijenjenih parametara na server
- agregacija modela

Prvi korak je inicijalizacija modela na serveru, isto kao i kod klasičnog centraliziranog učenja — inicijalizacija parametara, slučajno ili spremljene od posljednje kontrolne točke. Zatim se parametri globalnog modela šalju klijentskim čvorovima, koji započinju treniranje s istim parametrima. Ne treniraju do potpune konvergencije, već jednu epohu ili samo nekoliko koraka i nakon toga te promijenjene parametre šalju nazad centralnom serveru. Server sada ima više, međusobno malo različitih verzija globalnog modela koje je potrebno agregirati u jedan, centralni model. Najjednostavniji način je proces „federated averaging“ (FedAvg), koji uzima težinski prosjek svih modela, pri čemu je težina broj primjeraka koje je klijentski čvor koristio za lokalno testiranje. Ti se koraci ponavljaju sve dok se ne dobije potpuno trenirani model koji ima dobre performanse nad svim podacima klijentskih čvorova.



Slika 2.1: Arhitektura federalnog sustava[7]

2.3. Formalna definicija

Definira se N vlasnika podataka $\{F_1, \ldots, F_N\}$ koji će sudjelovati u procesu federalnog učenja trenirajući lokalne modele nad svojim podacima $\{D_1, \ldots, D_N\}$. Konvencionalna metoda bi zahtijevala da se treniranje vrši na centralnom čvoru nad podacima $D = D_1 \cup D_2 \cup \ldots \cup D_N$ i modelom M_{SUM}. U slučaju federalnog procesa učenja vlasnici bi zajednički trenirali model M_{FED} u kojem vlasnik podataka F_i ne otkriva drugim sudionicima svoje podatke D_i. Ovom definicijom se garantira privatnost koja je jedna od ključnih faktora. Definira se preciznost modela V_{FED}. Preciznost modela M_{FED} (V_{FED}) trebala bi biti približna preciznosti modela M_{SUM}. Neka postoji nenegativni realni broj δ za koji vrijedi $\mid V_{SUM} - V_{FED} \mid < \delta$. Onda algoritam federalnog učenja ima preciznost δ [14].

2.4. Svojstva

Privatnost je temeljno svojstvo sustava federalnog učenja. Postoje različiti modeli kojima se pružaju određene mjere privatnosti.

Sigurno udaljeno računanje (engl. *Secure Multi-party Computation*) podrazumijeva da klijenti koji sudjeluju u komunikaciji nemaju ikakva znanja jedni o drugima (engl. *zero knowledge*) osim vlastitog ulaza i izlaza. To svojstvo je poželjno, no nije nužno. Moguće je ublažiti zahtjeve tako da klijenti ipak znaju nešto, priskrbili su neke informacije o drugim klijentima u komunikaciji. Time se postiže učinkovitije računanje, uz nešto slabije sigurnosne postavke, ako scenarij to dozvoljava [3].

Sigurnost podataka može se osigurati dodavanjem bijelog šuma ili korištenjem metoda generalizacije (engl. *differential privacy*). Na taj način nije moguće razaznati doprinose klijenata trećim stranama, osiguravajući anonimnost. Problem je što se dodavanjem šuma podacima gubi na preciznosti modela federalnog učenja. Metoda se najčešće primjenjuje na strani klijenata prije koraka slanja podataka na centralni server [14].

Drugi način je koristeći homomorfnu enkripciju. Umjesto podataka, šalju se parametri modela (težine prilikom procesa učenja) šifrirani privatnim ključevima klijenata. Server zatim vrši agregaciju šifriranih parametara i uspostavlja zajednički model koristeći prethodno razmijenjeni javni ključ. Podaci se, za razliku od prethodne metode, više ne prenose mrežom i treće strane ih ne mogu pogoditi. Vjerojatnost curenja informacija iznimno je mala [2].

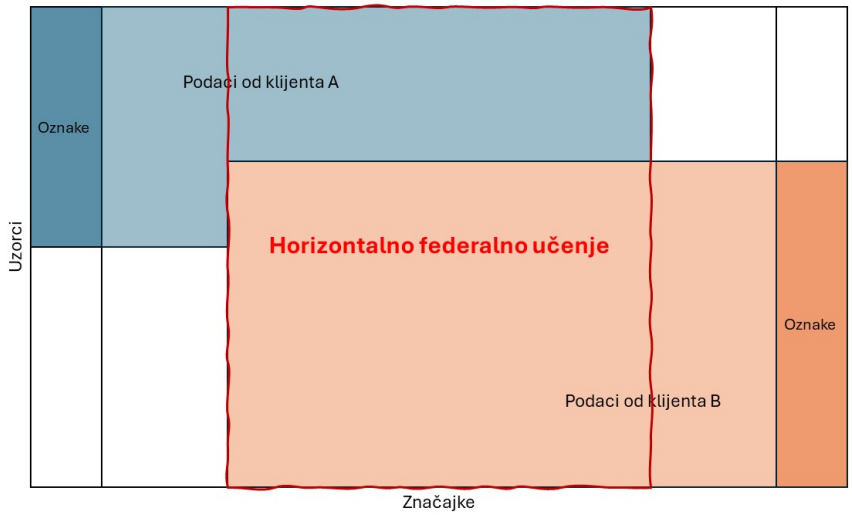
Postoji mogućnost da je klijent maliciozan. Jedna od metoda napada je ubacivanje

stražnjih vrata (engl. *backdoor*) u globalni model, kako bi mogli saznati informacije ostalih klijenata [1]. Mogući su i napadi u kojima maliciozni klijent može zaključiti identitete i informacije o nekim značajkama klijenata [11].

2.5. Kategorizacija

Procesi federalnog učenja mogu se kategorizirati na temelju distribucije podataka klijenata.

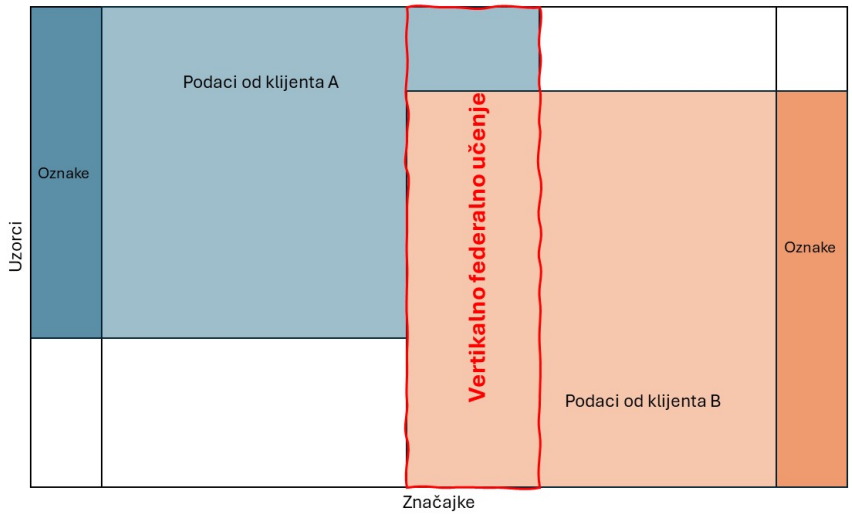
Horizontalno federalno učenje podrazumijeva da klijenti imaju isti prostor značajki, no kontekstualno nepovezane skupove podataka. Primjer su dvije banke, od kojih jedna posluje u Kini, a druga u SAD-u. Zbog sličnih načina poslovanja banaka možemo tvrditi da imaju sličan prostor značajki, no zbog malog preklapanja klijenata, različite skupove podataka. U sigurnosnom modelu pretpostavlja se da su klijenti iskreni (engl. *honest*), a server je iskren, no znatiželjan (engl. *honest-but-curious*). Znači samo server može kompromitirati podatke klijenata, dok oni međusobno to ne mogu.



Slika 2.2: Prikaz horizontalnog sustava

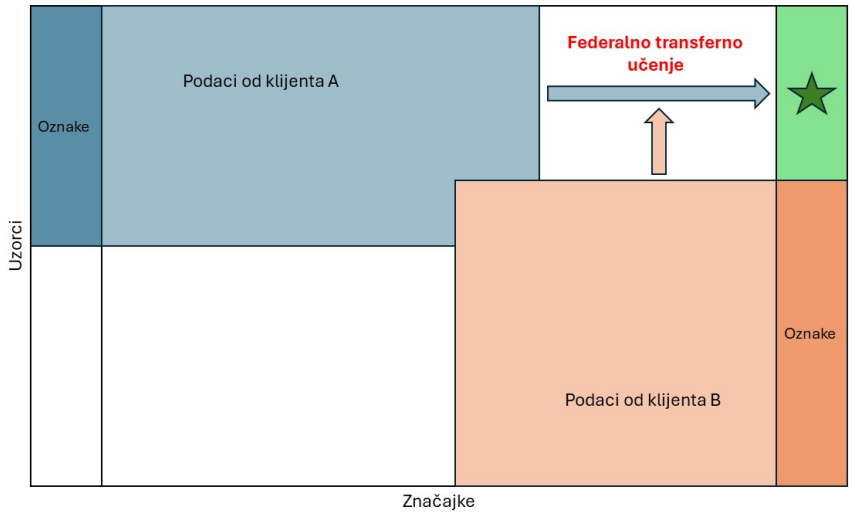
Vertikalno federalno učenje podrazumijeva da klijenti imaju različite prostore značajki, no njihovi skupovi podataka su kontekstualno povezani. Primjer su banka i trgovački centar koji posluju u istom gradu. Klijenti u banci vjerojatno kupuju u trgovačkom centru, tako da postoji presjek između skupova podataka. Zbog različitih modela poslovanja, ne postoji presjek u prostoru značajki. Što se sigurnosti tiče, za vertikalni sustav pretpostavlja se da su klijenti iskreni, no znatiželjni. U slučaju komunikacije

između dva klijenta, pretpostavlja se da maliciozni klijent može kompromitirati jednog klijenta. Tada ima pristup podacima klijenta nad kojim je izvršen napad, no nema pristup drugom klijentu koji sudjeluje u komunikaciji.



Slika 2.3: Prikaz vertikalnog sustava

Federalno transferno učenje podrazumijeva da klijenti imaju podatke različitih prostora značajki i različitih prostora oznaka. Važna je ekstenzija na prethodne sustave jer ponekad nije moguće primijeniti ni horizontalno i ni vertikalno učenje na takve skupove podataka. Pretpostavke sigurnosti transfernog sustava slične su kao i kod vertikalnog federalnog učenja [14].



Slika 2.4: Prikaz transfernog sustava

3. Federalno učenje temeljeno na grupama

3.1. Ideja i motivacija

Za razliku od tradicionalnog procesa učenja u oblaku, kod kojeg se svi podaci klijenata šalju na centralni server i tamo obrađuju, federalno učenje predstavlja primamljiv odabir za distribuirano učenje temeljeno na dijeljenju parametara. Zbog toga je jedan od najboljih odabira, pružajući svojstva privatnosti podatka i anonimnosti klijenata. Problem je da upravo zbog tih svojstava, sustav nema pristup podacima klijenata i ne može prikupljati statističke podatke. Može se dogoditi da su podaci klijenata statistički izrazito neheterogeni, odnosno podaci su nezavisni s različitim distribucijama (engl. *Non-IID*) .

Ponuđeno je nekoliko mogućih rješenja. Algoritam agregacije FedAvg[1] može konvergirati nad statistički heterogenim, Non-IID podacima [10]. FedAvg dopušta da se podaci spremaju lokalno, agregirajući samo gradijente lokalnih modela klijenata, koji se šalju na centralni server. Istraživanja pokazuju da se ipak javlja znatna deteoracija performasi nad Non-IID podacima. Lokalni modeli konvergiraju prema različitim modelima zbog heterogenosti distribucija podataka između klijenata. Agregacijom tih modela dovodi se do sve veće divergencije zajedničkog globalnog modela, od idealnog koji bi bio rezultat algoritma nad IID podacima [15]. Pokazuje se da će preciznost globalnog modela biti lošija i bit će potrebno više vremena do konvergencije.

Novije rješenje, klasterirano federalno učenje (engl. *Clustered Federated Learning, CFL*), uzima u obzir geometrijske značajke površina gubitka (engl. *loss surface*) federalnog učenja i grupira ih temeljeno na smjeru optimizacije. Na taj način zaobilazi prepreke statističke neheterogenosti podataka.

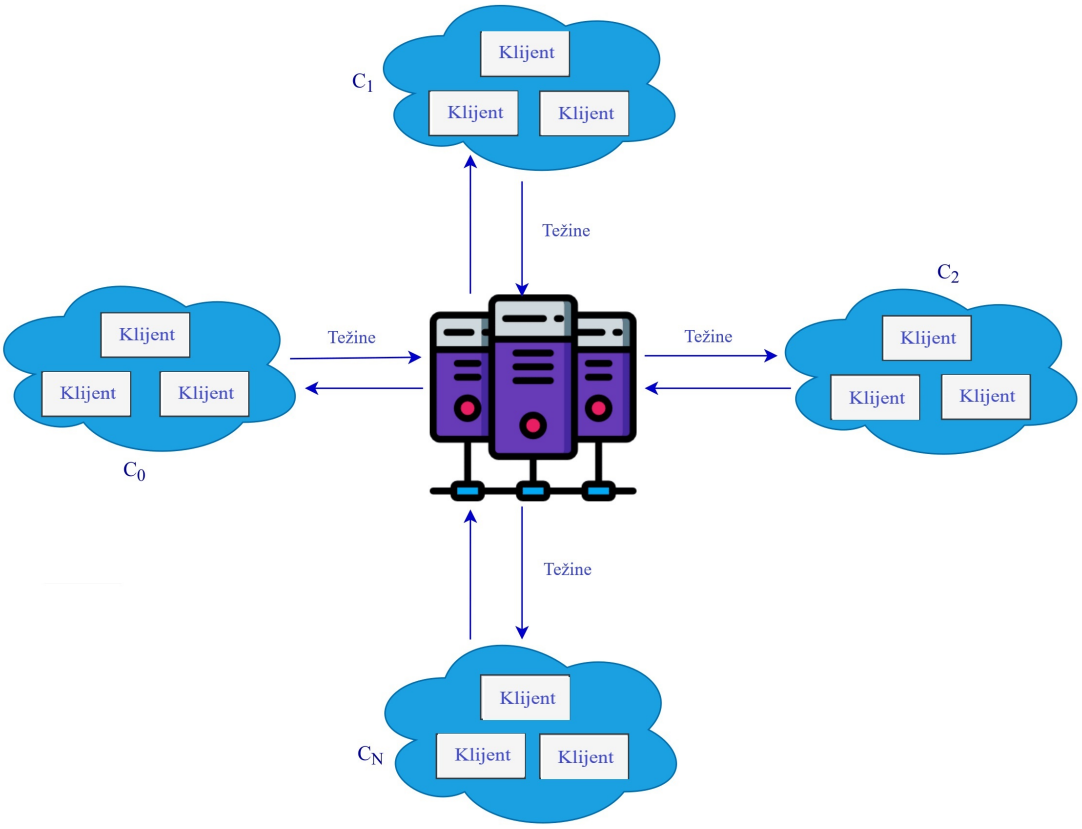
Primjer bi bili klijenti koje zanimaju različite kategorije vijesti, novosti i treniramo

[1] Federated Averaging

model koji će im svima moći preporučiti vijesti iz traženih kategorija. Struktura im je slična kao kod sustava preporučivanja (engl. *Recommendation System*).

Problem CFL temeljenih okvira je da nisu efektivni kao veći sustavi i ne uzimaju u obzir dolaske i odlaske klijenata. Pomaci u distribucijama podataka koji nastaju zbog isključivanja klijenata mogu dodatno pogoršati performanse grupnog sustava [4].

U nastavku će se razmotriti postojeći algoritmi grupiranja te će se usporediti performase pojedinih simulacijom u virtualnom okruženju.



Slika 3.1: Arhitektura federalnog sustava temeljenog na grupama[8]

3.2. Algoritmi grupiranja

3.2.1. Klasterirano federalno učenje, Sattler

Konvencionalan pristup federalnom slučaju ne pokriva slučajeve kada su podaci na klijentima različitih distribucija. Model nije dovoljno ekspresivan da pokrije sve distribu-

cije klijenata, pa se pretpostavlja da su one iste. Distribucije klijenata su kongruentne ako zadovoljavaju prethodno navedeni kriterij, a nekongruentne inače [13].

Kriterij kongruentnih distribucija klijenata u stvarnosti često nije zadovoljen, jedan od razloga je i to što centralni server ne može analizirati podatke klijenata zbog privatnosti i sigurnosnih svojstava federalnog učenja. Još jedan primjer je prethodno spominjani autocorrect, pretpostavlja se da starije osobe drugačije pišu tekstualne poruke od mlađih, zaključujemo da postoje različite distribucije tekstualnih poruka. Nedovoljno ekspresivan model ne bi mogao doći do pozitivnih rezultata nad takvim podacima.

Još jedan slučaj nekongruentnosti distribucija bi bio kad bi malicionzni klijent mijenjao svoje podatke tako da globalni model zaključuje beznačajne rezultate, potencijalno radeći štetu ostalim klijentima.

Rješenje problema predstavlja koncept klasteriranog federalnog učenja, CFL. Pretpostavlja se da postoji skup particija $C = \{c_1, c_2, \ldots, c_k\}$, $\cup_{i=1}^{k} c_k = \{1, \ldots, m\}$ populacije klijenata tako da svaki podskup particije $c \in C$ sadrži podatke s kongruentnim distribucijama.

U nastavku (Algoritam 1)[2] prikazan je algoritam klasteriranog federalnog učenja [13].

Algoritam 1 Klasterirano Federalno Učenje (CFL)

Ulaz: Inicijalni parametri θ, skup klijenata c, $\varepsilon > 0$

$\theta^* \leftarrow$ FederalnoUčenje(θ, c)
$\alpha_{i,j} \leftarrow$ sličnost između i-tog i j-tog klijenta, $i, j \in c$
$c_1, c_2 \leftarrow \arg\min_{c_1 \cup c_2 = c}(\max_{i \in c_1, j \in c_2} \alpha_{i,j})$
$\alpha_{cross}^{max} \leftarrow \max_{i \in c_1, j \in c_2} \alpha_{i,j}$
if $\alpha_{cross}^{max} \geq \varepsilon$ **i zadovoljeni su ostali uvjeti then**
    $\theta_i^*, i \in c_1 \leftarrow$ KlasteriranoFederalnoUčenje(θ^*, c_1)
    $\theta_i^*, i \in c_2 \leftarrow$ KlasteriranoFederalnoUčenje(θ^*, c_2)
else
    $\theta_i^* \leftarrow \theta^*, i \in c$
end if
return $\theta_i^*, i \in c$

Sad je jedino preostalo riješiti problem particioniranja klijenata. Particioniranje klijenata je ispravno ako klijenti s različitim distribucijama podataka nakon procesa particioniranja završe u istim skupovima — klasterima. To je moguće ostvariti bi-

[2] Algoritam je pojednostavljen u odnosu na originalni za potrebe ovog rada

particioniranjem, osnovna particija sa svim klijentima dijeli se na dvije tako da je particioniranje ispravno. Lako se vidi da će skup particija C nastati nakon $k-1$ koraka ispravnog bi-particioniranja.

Ispravno bi-particioniranje moguće je ostavariti promatrajući sličnost kosinusa (engl. *cosine similarity*) između gradijenata lokalnih modela klastera. Pretpostavlja se pojednostavljeni slučaj kod kojeg u učenju sudjeluju samo dva klijenta. Globalnu funkciju gubitka $F(\theta)$ moguće je zapisati kao $F(\theta) = a_1 R_1(\theta) + a_2 R_2(\theta)$, pri čemu je $R(\theta)$ funkcija gubitka pojedinog klijenta, θ parametri zajedničkog modela, $a_i = \sum_{i=1}^{m} \frac{|D_i|}{|D|}$, m broj klijenata, D_i podaci klijenta i, D podaci svih klijenata. Cilj federalnog učenja je ostvaren kada je zadovoljeno

$$0 = \nabla F(\theta^*) = a_1 \nabla R_1(\theta^*) + a_2 \nabla R_2(\theta^*) \tag{3.1}$$

gdje θ^* predstavlja finalne parametre modela nakon provedenog učenja.

Ako vrijedi $0 = \nabla R_1(\theta^*) = \nabla R_2(\theta^*)$, funkcije gubitka pojedinih klijenata su minimalne iz čega proizlazi da su distribucije klijenata kongruentne.

Inače, mora vrijediti

$$\nabla R_1(\theta^*) = -\frac{a_2}{a_1} \nabla R_2(\theta^*) \neq 0. \tag{3.2}$$

Sličnost kosinusa $\alpha_{i,j}$ između $\nabla R_i(\theta^*)$ i $\nabla R_j(\theta^*)$ dana je kao

$$\begin{aligned} \alpha_{i,j} &= \frac{\langle \nabla R_{I(i)}(\theta^*), \nabla R_{I(j)}(\theta^*) \rangle}{\|\nabla R_{I(i)}(\theta^*)\| \|\nabla R_{I(j)}(\theta^*)\|} \\ &= \begin{cases} 1 & \text{ako } I(i) = I(j) \\ -1 & \text{ako } I(i) \neq I(j) \end{cases} \end{aligned}$$

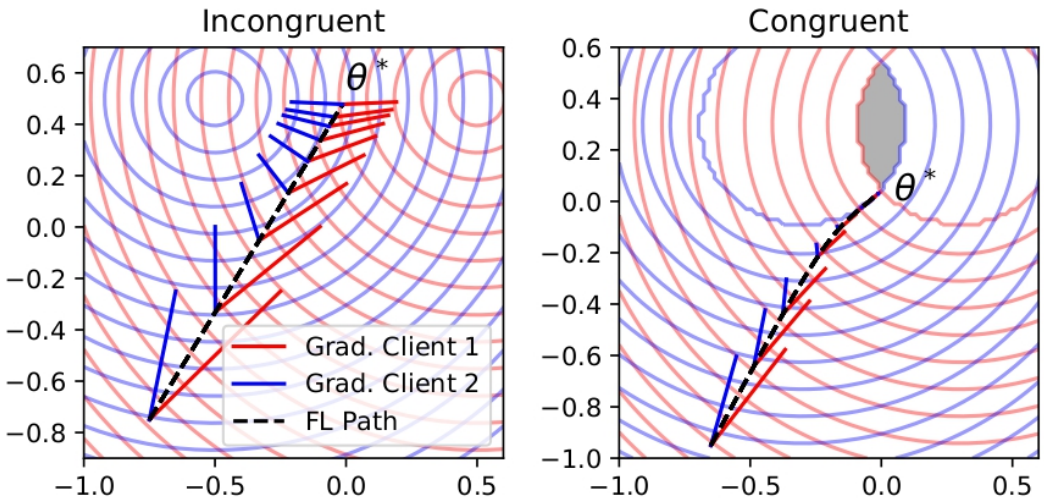
pri čemu funkcija $I : \{1, \ldots, m\} \to \{1, \ldots, k\}$ klijentu i pridružuje njegovu distribuciju podataka $\varphi_{I(i)}$.

Ispravno bi-particioniranje ostvareno je kao

$$c_1 = \{i | \alpha_{i,0} = 1\}, \quad c_2 = \{i | \alpha_{i,0} = -1\}.$$

Razliku između optimizacijskih puteva federalnog učenja nad kongruentnim i nekongruentnim skupovima podataka moguće je vidjeti na slici 3.2.

Kod primjera s nekongruentnim podacima vidi se da su norme klijentskih gradijenata pozitivne i da su gradijenti okrenuti u suprotnim smjerovima. U slučaju kongruentnih podataka postoji sivo područje gdje su obje funkcije gubitka minimalne. Ako federalno učenje konvergira prema tom području, norme gradijenata spuštaju se u nulu.



Slika 3.2: Prikaz optimizacijskih puteva između modela kongruentnih i nekongruentnih podataka[13]

Ako su podaci klijenata nekongruentni, stacionarno rješenje[3] federalnog učenja ne može biti stacionarno za individualne klijente po definiciji. Kao posljedica tog zaključka, norma klijentskih gradijenata mora biti striktno veća od nule. Ako su podaci kongruentni, norma gradijenata bit će jednaka nuli.

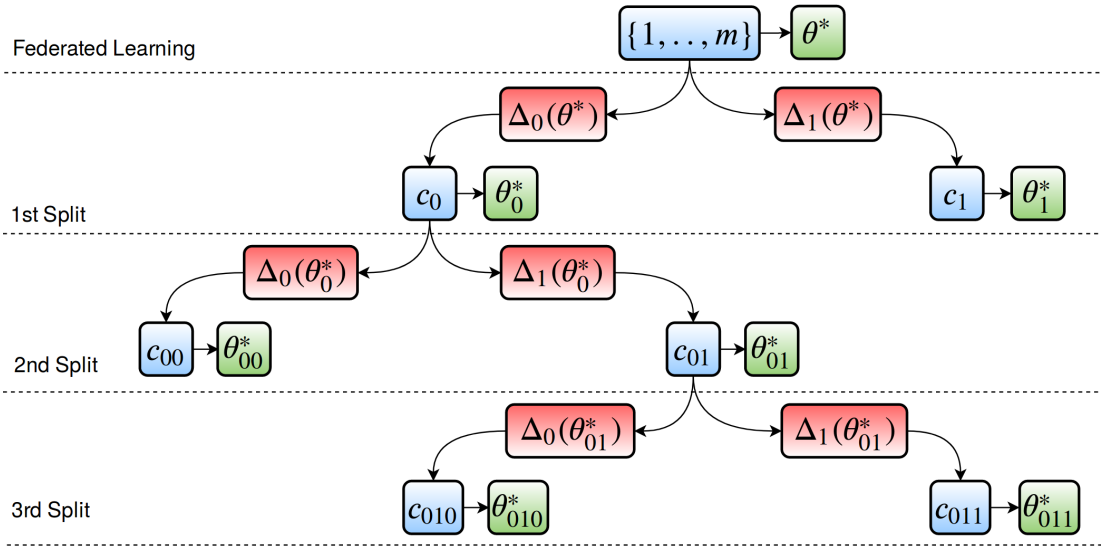
Particioniranje bi se trebalo raditi samo kad je zajednički model blizu inferencije, no individualni klijenti su daleko od stacionarne točke cilja učenja. U praksi će se prvo provesti konvencionalna metoda federalnog učenja, zatim će se ponoviti proces uz primjenu bi-particioniranja i u svakom koraku uzimaju se bolji rezultati. Ako se model degradira primjenom particioniranja, uvijek je moguće vratiti se na model dobiven konvencionalnim pristupom. Primjer procesa vidljiv je na slici 3.3.

3.2.2. Iterativno klasterirano federalno učenje, IFCA

I kod ovog algoritma odnosno okvira pokušava se riješiti problem heterogenosti podataka između različitih klijenata. Najveći izazov su nepoznati identiteti različitih klastera — ne zna se odmah na početku kako će se klijenti grupirati, i optimizacija modela klastera. Glavna ideja IFCA algoritma je strategija koja alternira između procjene identiteta klastera i minimizacije funkcije gubitka (engl. *loss function*) i kao takav se može svrstati u algoritme alternirajuće minimizacije u distribuiranom okruženju [5].

Jedno od svojstava IFCA algoritma je da ne zahtijeva računanje od strane centralnog servera, što ga znatno rasterećuje i smanjuje računalne i memorijske zahtjeve.

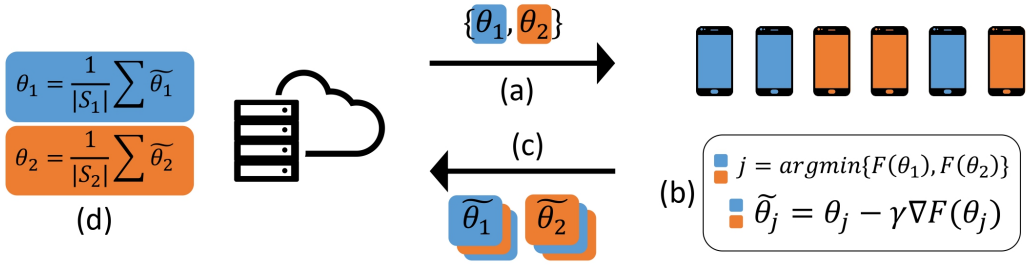
[3]Model dobiven provedbom konvencionalnog procesa federalnog učenja, bez primjene grupiranja



Slika 3.3: Prikaz bi-particioniranja temeljen na distanci gradijenata [13]

Autori su dokazali da algoritam pruža eksponencijalne brzine konvergencije u odnosu na broj komunikacijskih rundi između servera i klijenata, uz korištenje linearne ili vrlo konveksne funkcije gubitka.

Postoje dvije različite varijante algoritma, jedna agregira gradijente, a druga modele klijenata. Na slici 3.4 prikazani su koraci rada algoritma.



Slika 3.4: Prikaz rada IFCA algoritma. (a) Server šalje modele klijentima. (b) Klijenti identificiraju svoj klaster i počnu trenirati modele. (c) Klijenti šalju svoje lokalne modele natrag serveru. (d) Server agregira modele unutar istih klastera. [5]

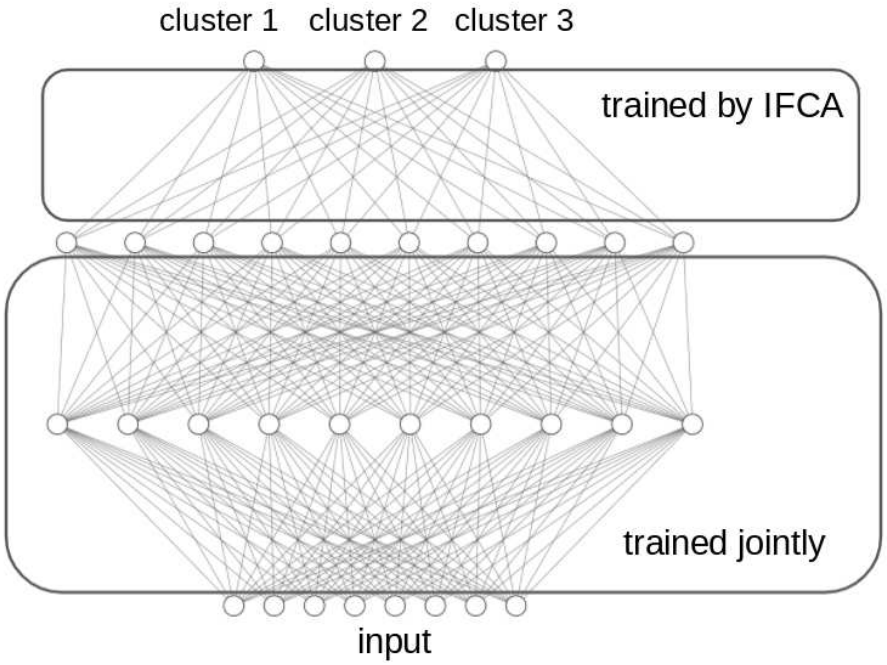
Algoritam započinje s k inicijalnih parametara modela $\theta_j, j \in \{1, \ldots, k\}$. U t-toj iteraciji, centralni server slučajno određuje podskup klijenata M_t. Svaki klijent posjeduje empirijsku funkciju gubitka F_i. Pomoću funkcije F_i i dobivenih parametara, i-ti klijent procjenjuje identitet svog klastera tako da pronađe parametar modela s najmanjim gubitkom $\widehat{j} = \text{argmin}_{j \in \{1,\ldots,k\}} F_i(\theta_j)$.

Ako se koristi metoda agregacije po gradijentima modela, klijenti računaju stohastički gradijent lokalne funkcije gubitka F_i u $\theta_{\widehat{j}}$. Zatim šalje svoju procjenu identiteta

klastera i gradijent natrag centralnom serveru. Centralni server zaprimi sve gradijente klijenata i provodi metodu gradijentnog spusta nad parametrima pripadnih klastera.

Ako se koristi metoda agregacije lokalnih modela (slična algoritmu FedAvg), svaki klijent provodi nekoliko koraka metode gradijetnog spusta nad lokalnim parametrima. Zatim šalje novoizračunati model zajedno s procjenom identiteta klastera natrag centralnom serveru. Centralni server zatim agregira modele klijenata koji pripadaju istom klasteru.

U praktičnoj primjeni, postoji mogućnost da iako su distribucije podataka klijenata u različitim klasterima različite, one ipak imaju neka zajednička svojstva. Za te slučajeve koristi se kombinacija multi-task learninga i IFCA. Specifično, kada se trenira model neuronske mreže, težine prvih par slojeva dijele se između svih klastera i nakon toga se nad zadnjim slojevima primijeni IFCA algoritam, adresirajući pritom različite distribucije različitih klastera. Jedna od prednosti ovog koncepta je smanjenje komunikacije preko mreže, centralni server sad klijentima mora poslati samo jednu kopiju težina dijeljenih slojeva mreže i k različitih podskupa težina za ostale slojeve, umjesto slanja k modela svim klijentima. Shema takvog sustava prikaza je na slici 3.5.



Slika 3.5: Shema sustava s dijeljenim težinama između klastera [5]

3.2.3. FlexCFL

FlexCFL[4] također je okvir koji pruža podršku za provođenje klasteriranog federalnog učenja, inspiriran CFL-om. Postoji niz prednosti koje pruža u odnosu na prethodne algoritme. FlexCFL podržava mogućnost dolaska novih klijenata tijekom procesa federalnog učenja i dodjeljivanje novog klijenta u klaster korištenjem algoritma hladnog paljenja (engl. *cold start*). Također može efikasnije izračunati udaljenost između klijenata, računajući sličnosti kosinusa između gradijenata modela samo u smjerovima dobivenim SVD[5] dekompozicijom [4]. Autori tu udaljenost nazivaju euklidskom udaljenošću dekompozirane sličnosti kosinusa (engl. *Euclidean distance of Decomposed Cosine similarity, EDC*).

FlexCFL prvo pošalje globalni model odabranom podskupu svih klijenata. Naime, uzima se podskup jer je nerealno očekivati da će svi klijenti biti dostupni u tom trenutku, što može uzrokovati kašnjenja. Zatim podijeli podskup u m klastera korištenjem statičkog K-means++ algoritma temeljenog na EDC. Prednost takvog pristupa je da iako podaci mogu biti kongruentni, nećemo izgubiti na performansama ili preciznosti ako klijente podijelimo u m klastera i upotrijebimo FlexCFL algoritam umjesto konvencionalnog FedAvg. Računanje matrice sličnosti zahtijeva samo jednu komunikacijsku rundu, koja je ujedno i prva. Naziva se grupnim hladnim paljenjem (engl. *group cold start*).

Nakon grupnog hladnog paljenja, računaju se optimizacijski smjerovi pojedinačnih grupa $\Delta\mathbf{w}^{(j)}, j \in \{1, \ldots, m\}$ koji se šalju centralnom serveru. Centralni server agregira sve modele grupa za potrebe hladnog paljenja novih klijenata. Proces se ponavlja (osim grupnog hladnog paljenja) do inferencije.

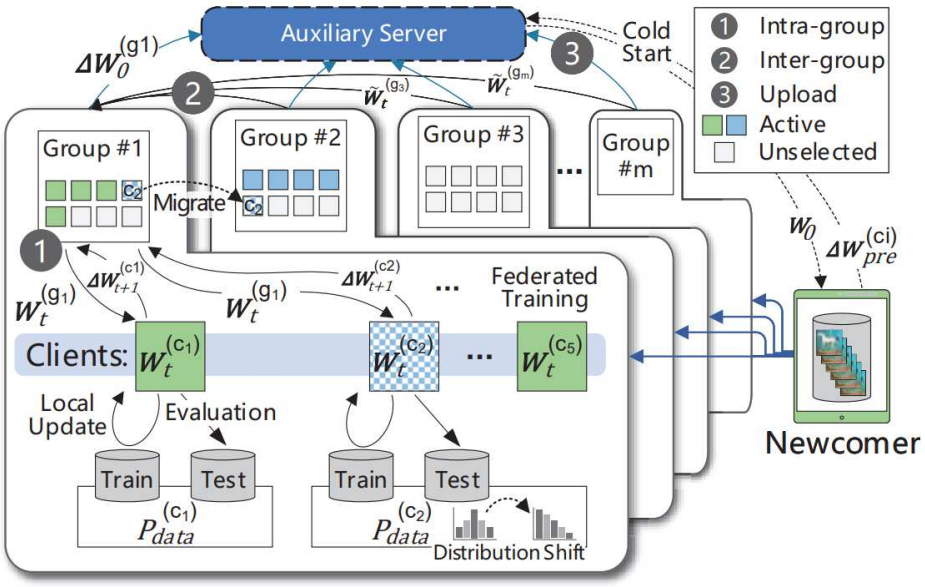
FlexCFL također pokriva slučaj kad se distribucija podataka mijenja (engl. *distribution shift*), npr. kod senzora. U tom slučaju potrebno je klijente premjestiti u nove klastere. Zbog toga se prije početka treniranja računa Wasserstein udaljenost za sve klijente unutar grupe. Ako udaljenost prelazi preko preddefinirane vrijednosti, za tog klijenta pokreće se hladno paljenje.

Na slici 3.6 prikazana je arhitektura okvira FlexCFL.

3.2.4. FeSEM

Okvir za federalno učenje FeSEM[6] sličan je okviru FlexCFL.

[4] Flexible Clustered Federated Learning
[5] Singular Value Decomposition
[6] Federated Stochastic Expectation Maximization



Slika 3.6: Arhitektura okvira FlexCFL [4]

Pretpostavlja se da klijent i ima zadane podatke D_i, s time da X_i predstavljaju značajke, a Y_i oznake odnosno labele. Svaki od skupova podataka D_i će se koristiti za treniranje lokalnih modela $M_i : X_i \rightarrow Y_i$, M predstavlja duboki neuronski model parametriziran po težinama W. Za i-tog klijenta procedura treniranja definirana je kao $\min_{W_i} L_s(M_i, D_i, W_i)$, gdje $L_s(\cdot)$ predstavlja funkciju gubitka klijenta.

Generalno su podaci jednog klijenta nedovoljni za treniranje modela neuronske mreže. Zbog toga federalni sustav provodi distribuirano treniranje nad više klijenata kako bi se minimizirala funkcija gubitka. Optimizacija u konvencionalnom FL frameworku može se zapisati kao $\min_{\{W_i\}_{i=1}^{m}} \alpha_i L_s(M_i, D_i, W_i)$, gdje m predstavlja broj klijenata, a $\alpha_i = \frac{|D_i|}{\Sigma_j D_j}$ predstavlja značajnost podataka i-tog klijenta.

Server zatim agregira težine klijenata u globalni model M_{global}, odnosno uzima aritmetičku sredinu težina klijenata moduliranu s α.

$$\tilde{W}^g = \sum_{i=1}^{m} \alpha_i W_i \tag{3.3}$$

$\tilde{W}^g$ predstavlja sredinu, najbližu točku svim težinama klijenata W_i, u ovom slučaju koristeći L2 udaljenost, no moguće ju je poopćiti na bilo koju drugu funkciju udaljenosti $Dist(\cdot, \cdot)$.

Cilj je pronaći model za koji će udaljenost između lokalnih i globalnog modela biti

minimalna, odnosno minimizaciju funkcije $\min_{\tilde{W}} \frac{1}{m} \sum_{i=1}^{m} Dist(\tilde{W}, W_i)$. Funkcija će biti minimalna ako klijente podijelimo u m klastera tako da su podaci u klasteru kongruentni. Funkcija se zatim rastavi u m funkcija tako da postoji $\tilde{W}^{(k)}, k \in \{1, \ldots, m\}$ za svaki klaster. Problem se svodi na minimizaciju

$$\min_{\{r_i^{(k)}\},\{\tilde{W}^{(k)}\}} \frac{1}{m} \sum_{k=1}^{K} \sum_{i=1}^{m} r_i^{(k)} Dist(\tilde{W}^{(k)}, W_i) \tag{3.4}$$

gdje $r_i^{(k)}$ predstavlja funkciju pripadnosti klijenta i klasteru k, K broj klastera. Na zaključku iz 3.4 temelji se algoritam FeSEM [9].

Proces FeSEM-a dijeli se na tri faze. Prije početka podijelimo klijente u klastere koristeći K-means algoritam temeljen na L2 udaljenosti. Zatim, u prvoj — **E-fazi**, za klijenta i osvježavaju se $r_i^{(k)}$ s fiksiranim W_i. $r_i^{(k)}$ bit će jednak 1 ako je za klaster k L2 udaljenost između W_i i centra $\tilde{W}^{(k)}$ minimalna, inače je jednak 0.

U sljedećoj **M-fazi** osvježavaju se centralne točke $\tilde{W}^{(k)}$ klastera koristeći formulu

$$\tilde{W}^{(k)} = \frac{1}{\sum_{j=1}^{m} r_j^{(k)}} \sum_{i=1}^{m} r_i^{(k)} W_i \tag{3.5}$$

Na kraju, osvježavaju se lokalni modeli, $\tilde{W}^{(k)}$ šalju se klijentima pripadnog klastera. Klijenti zatim na temelju dobivenih parametara pokreću lokalno treniranje nad svojim privatnim podacima.

Tri faze se ponavljaju dok se ne zadovolji uvjet konvergencije. U nastavku je naveden algoritamski prikaz procesa.

Algoritam 2 FeSEM

Ulaz: Inicijalni parametri K, $\{W_i\}_{i=1}^m$, $\{\tilde{W}^{(k)}\}_{k=1}^K$

while nije zadovoljen uvjet stajanja **do**
    E-faza:
    Izračunaj udaljenost $d_{ik} = Dist(W_i, \tilde{W}^{(k)})\ \forall i, k$
    Osvježi $r_i^{(k)}$ koristeći d_{ik}
    M-faza:
    Osvježi $\tilde{W}^{(k)}$ koristeći $r_i^{(k)}$ (formula 3.5)
    for each klaster $k = 1, \ldots, K$ **do**
        for each $i \in C_k$ **do**
            Pošalji $\tilde{W}^{(k)}$ klijentu i
            $W_i \leftarrow$ **Lokalno_treniranje**$(i, \tilde{W}^{(k)})$
        end for
    end for
end while

Algoritam 3 Lokalno_treniranje

i – indeks klijenta

$\tilde{W}^{(k)}$ – parametri modela dobiveni od servera

W_i – osvježeni lokalni model

Inicijalizacija: $W_i \leftarrow \tilde{W}^{(k)}$
for N lokalnih epoha **do**
    Osvježi W_i procesom treniranja nad podacima D_i
end for
Vrati W_i serveru

4. Implementacija i testiranje

Nakon opisa rada algoritama federalnog klasteriranja, potrebno im je usporediti performanse. U tu svrhu napravljena su testiranja nad različitim modelima neuronskih mreža, implementirane pomoću programskih paketa. Za evaluaciju rada algoritama koristi se istoimeni FlexCFL okvir [4], temeljen na programskom jeziku Python i programskom paketu TensorFlow. Uspoređuju se performanse konkretno FedAvg, IFCA, FeSEM i FlexCFL algoritama. Treniranje modela radi se nad skupovima podataka MNIST, FEMNIST i Fashion MNIST. Skupovi su modificirani tako da se približi doživljaj statističke heterogenosti, prisutan na stvarnim podacima. Korišteni su modeli višeslojnog perceptrona (MLP) i konvolucijske neuronske mreže (CNN), ovisno o ulaznom skupu podataka. Za MNIST i FEMNIST je korišten CNN model od 6 konvolucijskih slojeva, za Fashion MNIST od 2 konvolucijska sloja. Za MLP model su za FEMNIST i Fashion MNIST korištene neuronske mreže s jednim skrivenim slojem od 512 neurona, a za MNIST sa jednim skrivenim slojem od 128 neurona. Treniranje se provodilo u 300 komunikacijskih rundi između servera i klijenata, a klijenti su provodili 10 epoha lokalnog treniranja. Učenje se provodilo s minigrupama (engl. *mini-batches*) od 10 predočenih uzoraka. U procesu je sudjelovalo 20 klijenata.

U nastavku su prikazani grafovi i tablice za preciznost i pogrešku evaluirane prilikom treniranja i testiranja te stupanj različitosti težina klijentskih modela od globalnog.

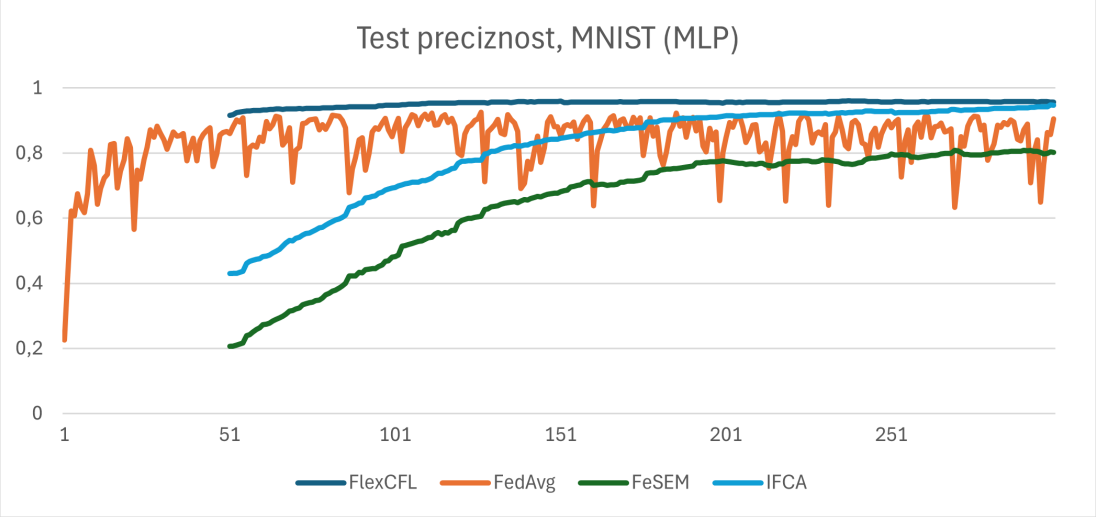
4.1. Rezultati testiranja

Legenda:

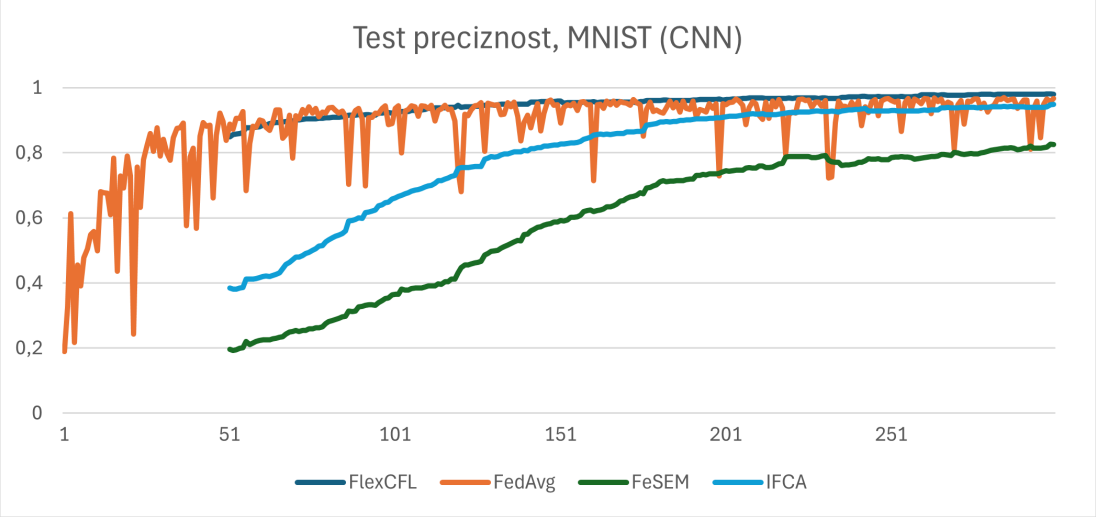
- x-os: redni broj komunikacijske runde
- Test preciznost
    - y-os: preciznost modela na testnom skupu, računa se kao broj točno predviđenih oznaka podijeljen s ukupnim brojem uzoraka
- Pogreška

- y-os: vrijednost funkcije pogreške u x-toj komunikacijskoj rundi

- Stupanj različitosti težina
  - y-os: stupanj različitosti između lokalnih[1] i globalnih težina, računa se kao prosjek apsolutnih odstupanja lokalnih od globalnih težina

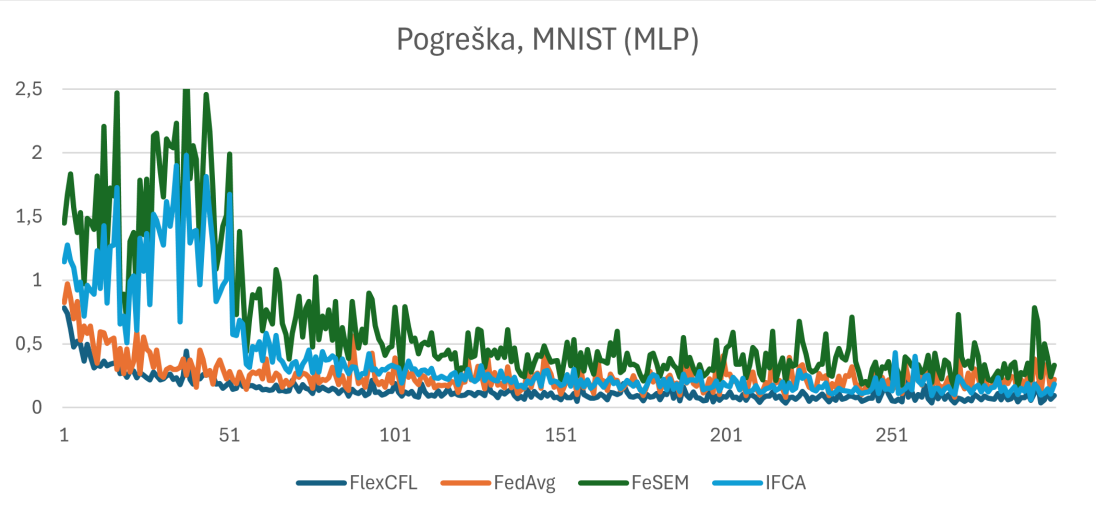


Slika 4.1: Graf test-preciznosti algoritama nad MNIST datasetom, uz korišten MLP model

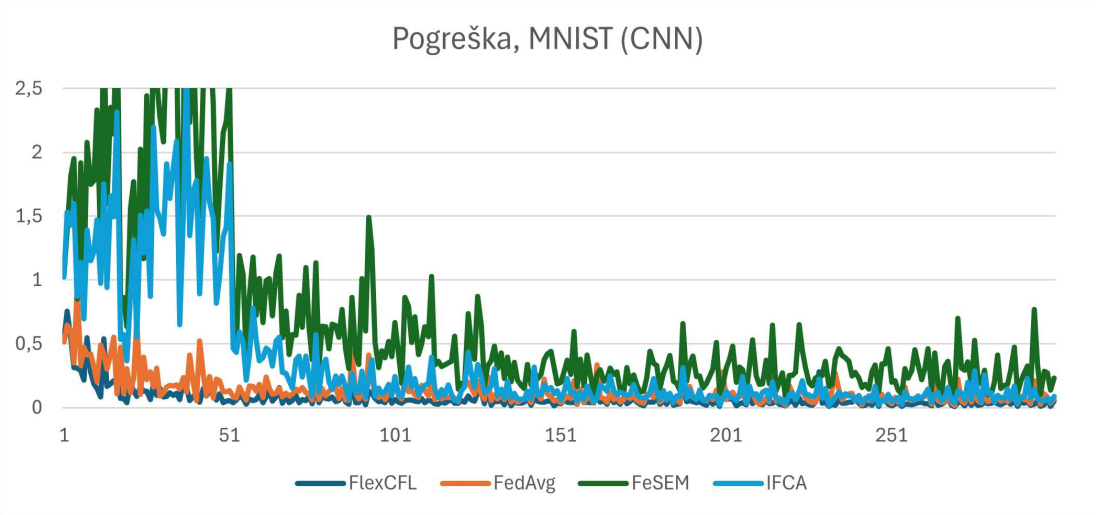


Slika 4.2: Graf test-preciznosti algoritama nad MNIST datasetom, uz korišten CNN model

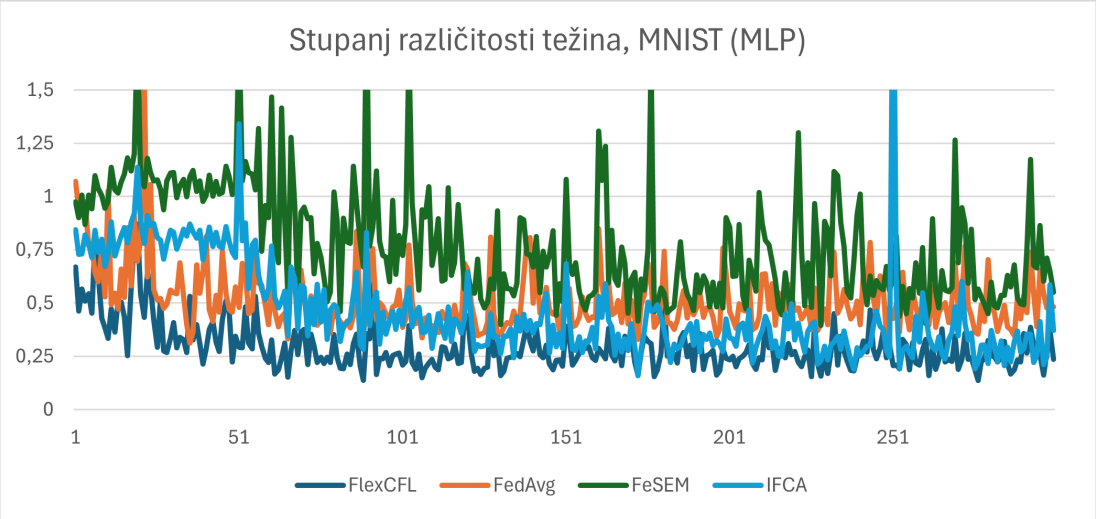
[1]Težine modela klijenata (istog klastera) prije agregiranja u zajednički globalni model



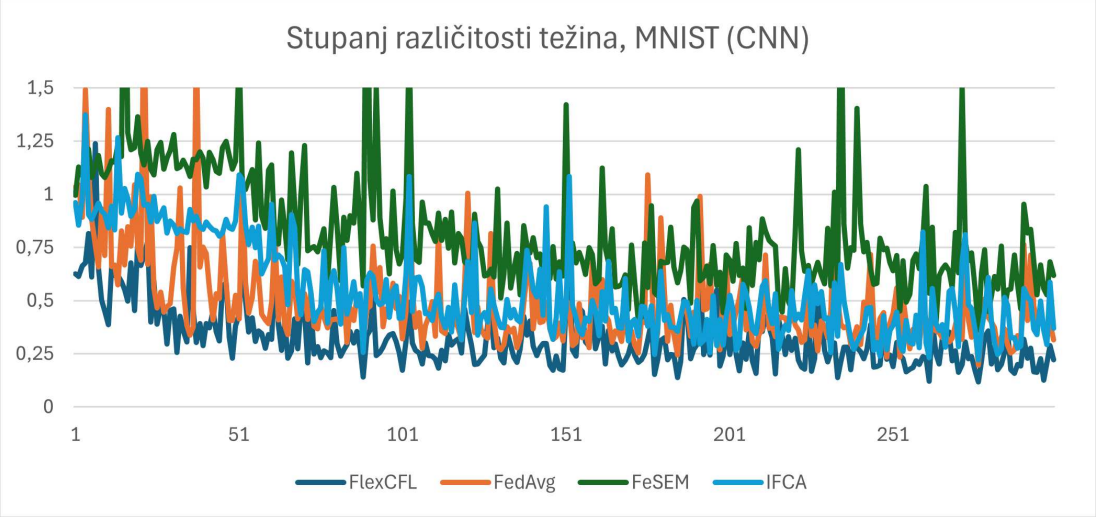
Slika 4.3: Graf pogreške algoritama nad MNIST datasetom, uz korišten MLP model



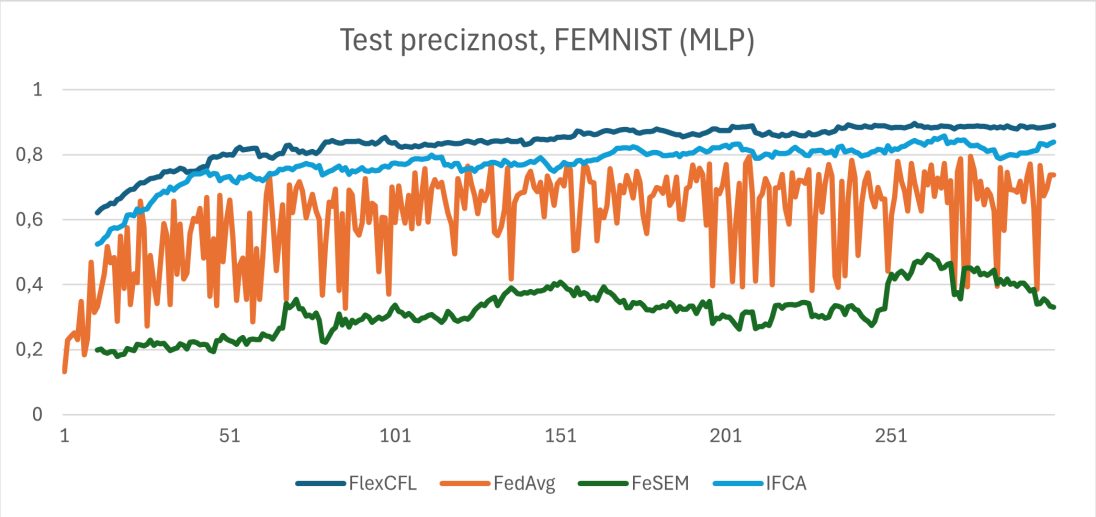
Slika 4.4: Graf pogreške algoritama nad MNIST datasetom, uz korišten CNN model



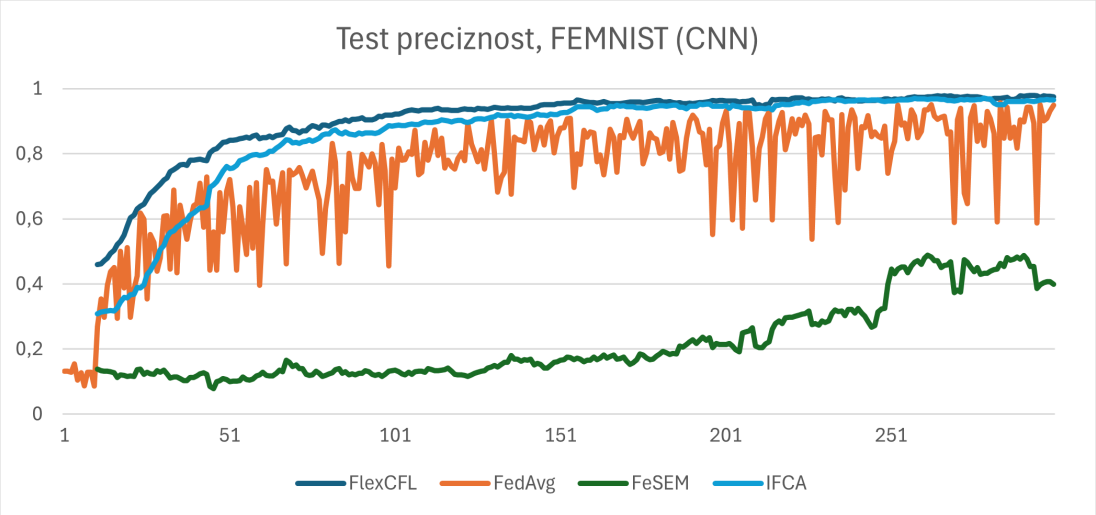
Slika 4.5: Graf stupnja različitosti lokalnih i globalnih težina nad MNIST datasetom, uz korišten MLP model



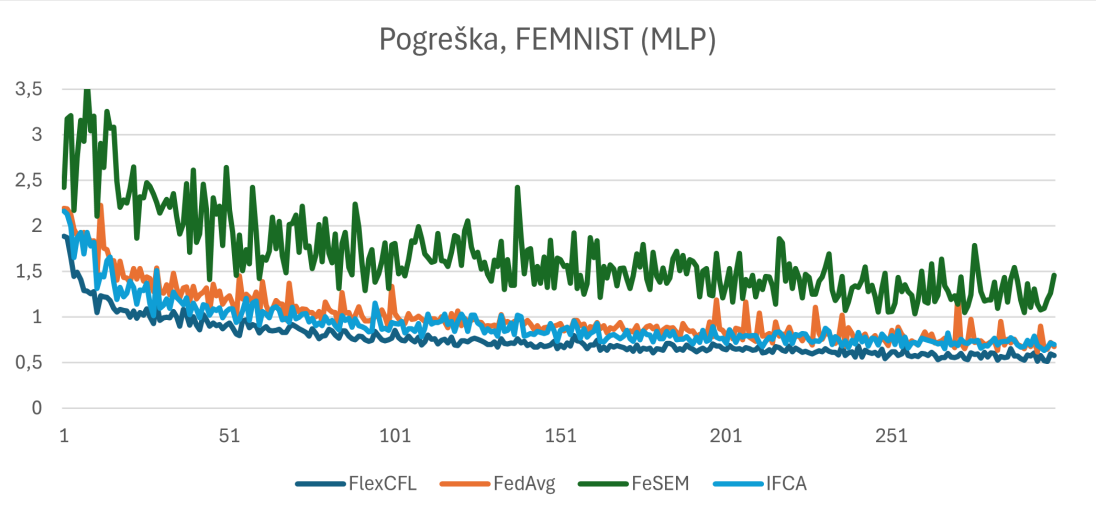
Slika 4.6: Graf stupnja različitosti lokalnih i globalnih težina nad MNIST datasetom, uz korišten CNN model



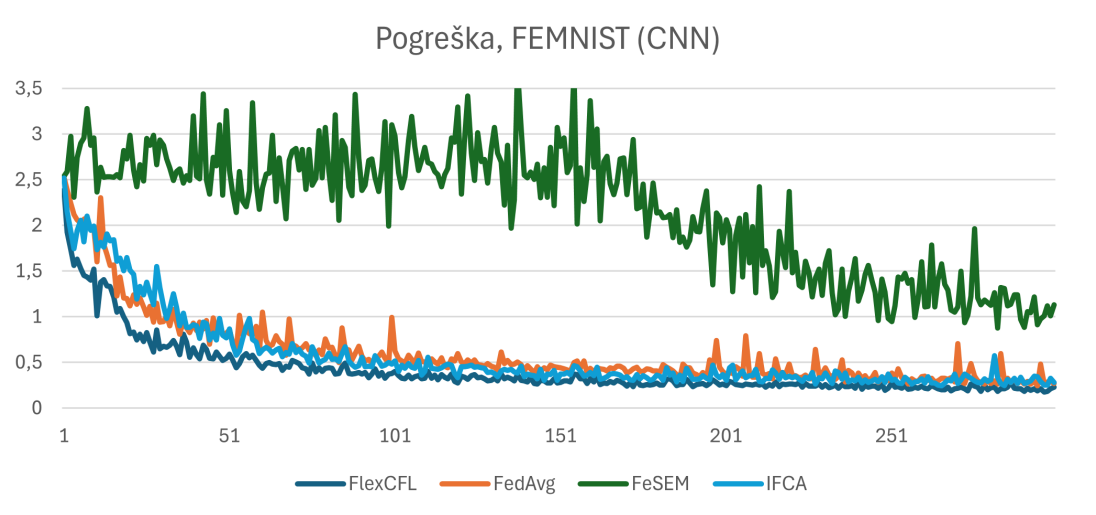
Slika 4.7: Graf test-preciznosti algoritama nad FEMNIST datasetom, uz korišten MLP model



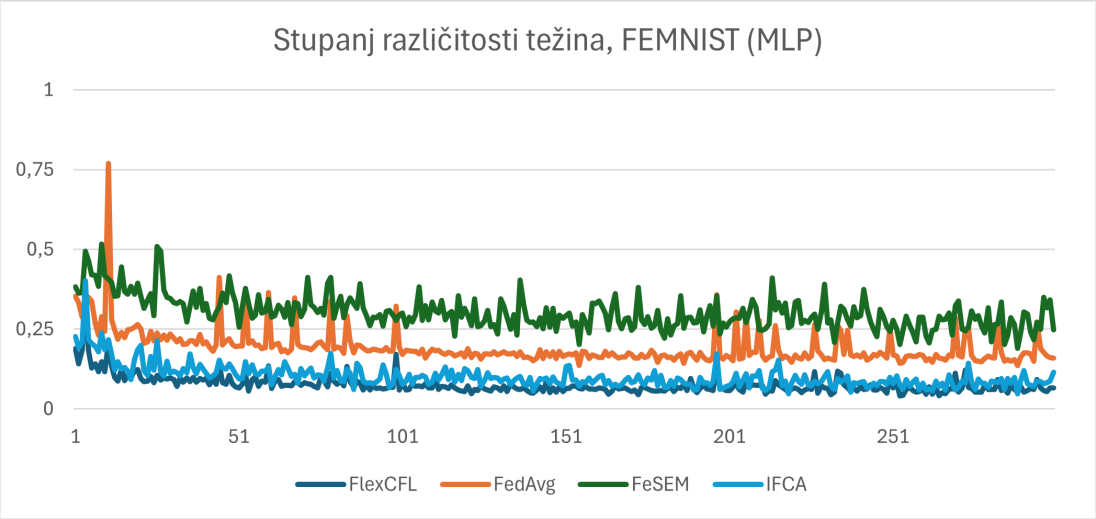
Slika 4.8: Graf test-preciznosti algoritama nad FEMNIST datasetom, uz korišten CNN model



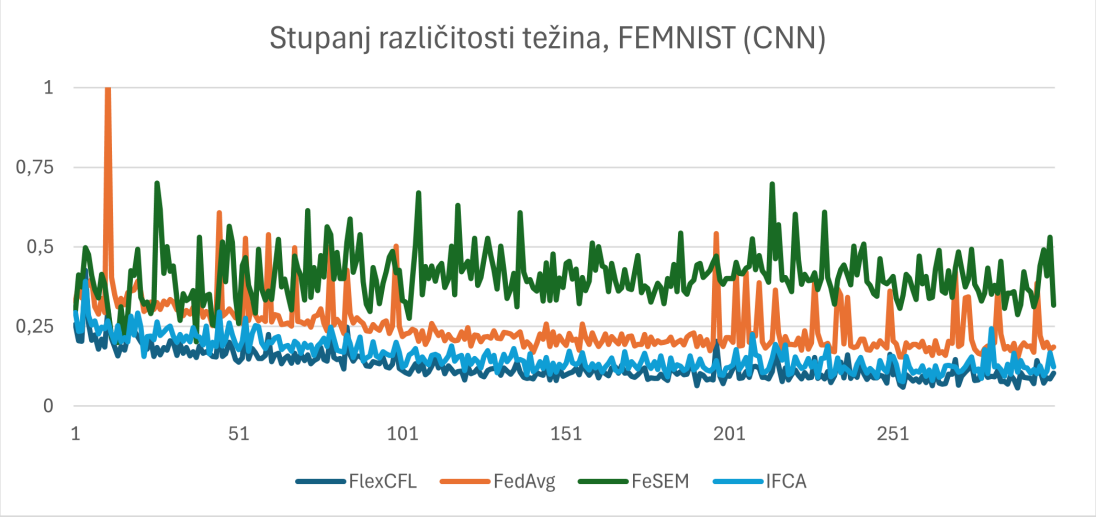
Slika 4.9: Graf pogreške algoritama nad FEMNIST datasetom, uz korišten MLP model



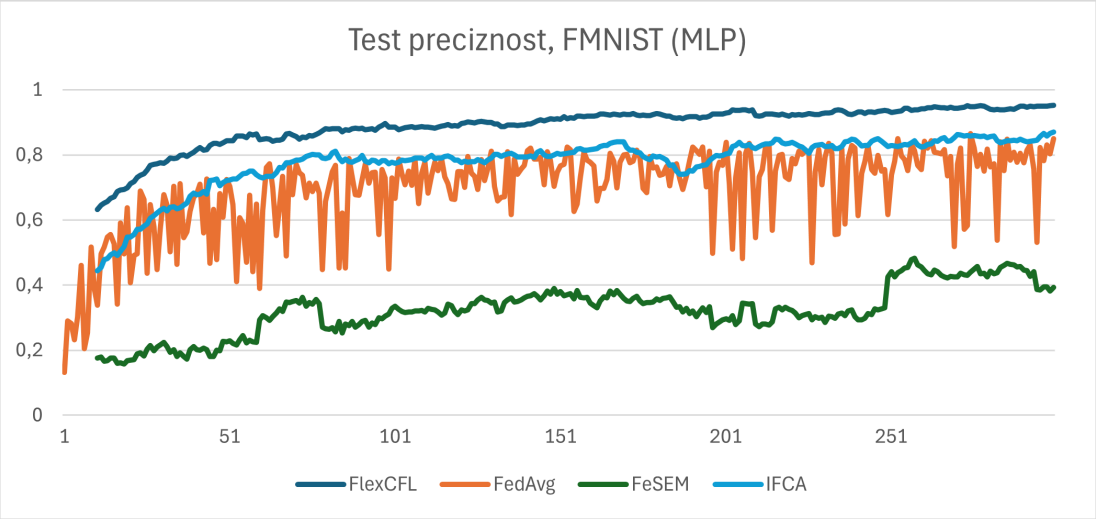
Slika 4.10: Graf pogreške algoritama nad FEMNIST datasetom, uz korišten CNN model



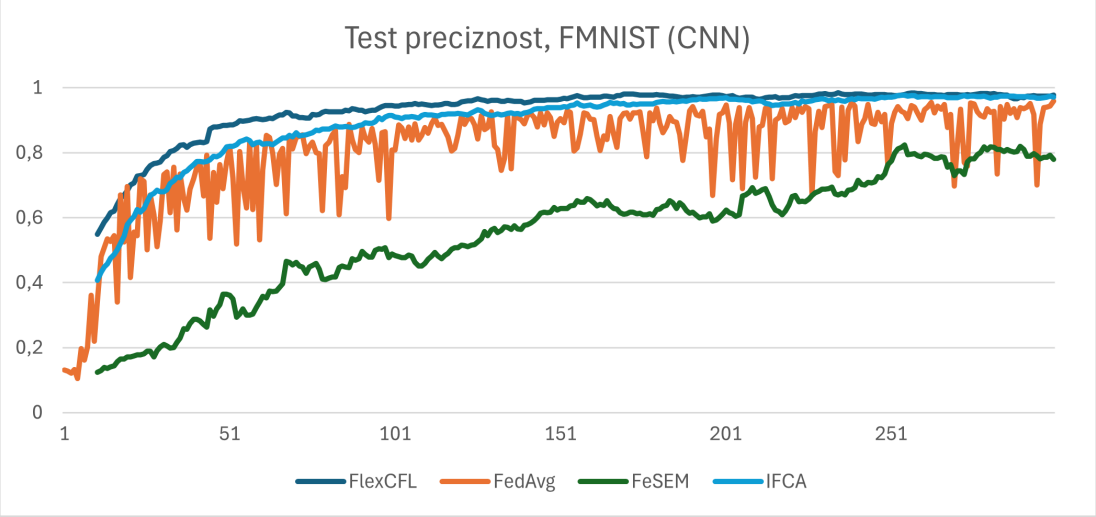
Slika 4.11: Graf stupnja različitosti lokalnih i globalnih težina nad FEMNIST datasetom, uz korišten MLP model



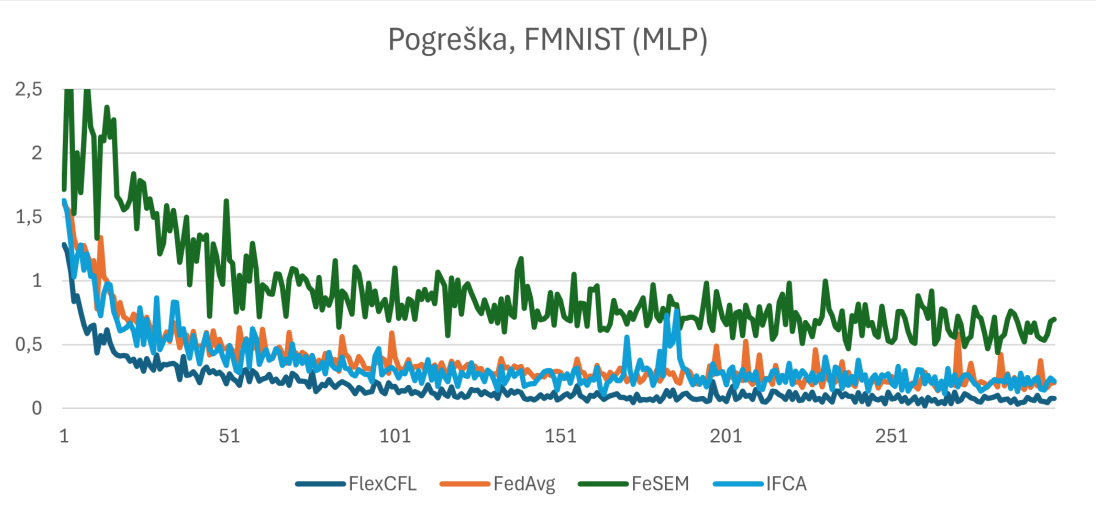
Slika 4.12: Graf stupnja različitosti lokalnih i globalnih težina nad FEMNIST datasetom, uz korišten CNN model



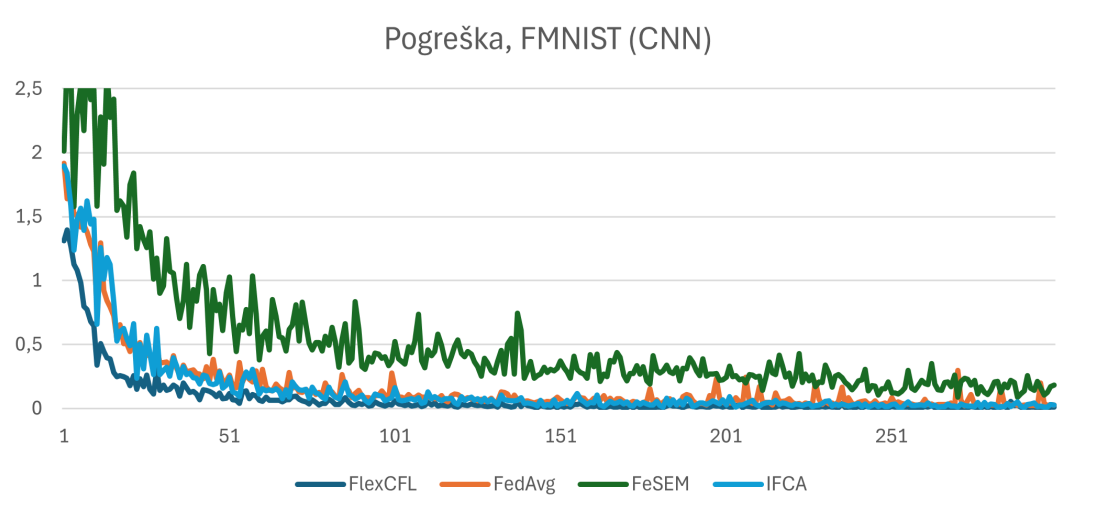
Slika 4.13: Graf test-preciznosti algoritama nad Fashion MNIST datasetom, uz korišten MLP model



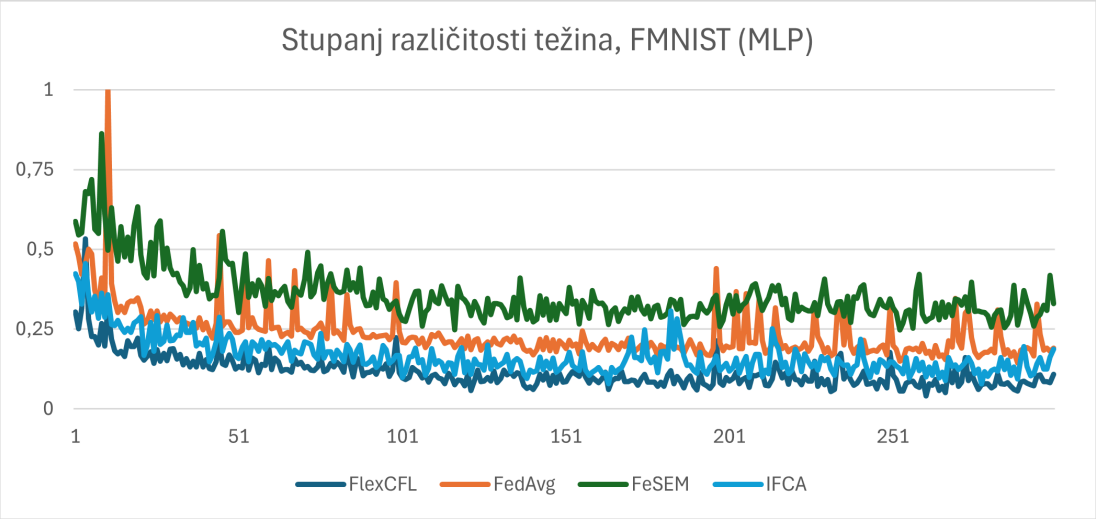
Slika 4.14: Graf test-preciznosti algoritama nad Fashion MNIST datasetom, uz korišten CNN model



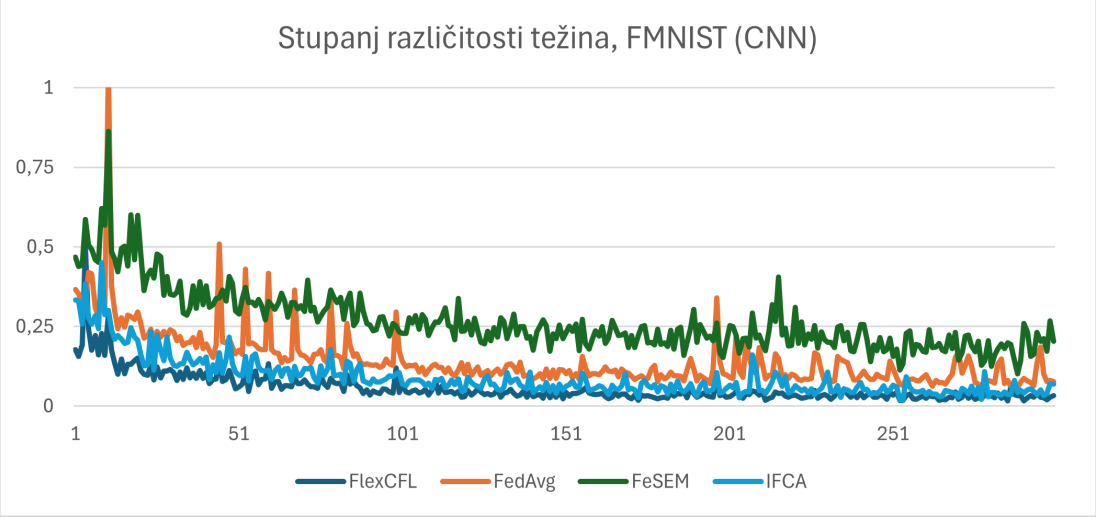
Slika 4.15: Graf pogreške algoritama nad Fashion MNIST datasetom, uz korišten MLP model



Slika 4.16: Graf pogreške algoritama nad Fashion MNIST datasetom, uz korišten CNN model



Slika 4.17: Graf stupnja različitosti lokalnih i globalnih težina nad Fashion MNIST datasetom, uz korišten MLP model



Slika 4.18: Graf stupnja različitosti lokalnih i globalnih težina nad Fashion MNIST datasetom, uz korišten CNN model

4.2. Interpretacija rezultata

Može se zaključiti da FeSEM algoritam ima najmanju preciznost testiranja i najsporije konvergira. To je rezultat neuspjele strategije grupiranja klijenata, npr. za MNIST i FEMNIST (MLP) skupove FeSEM sve klijente grupira u jednu grupu. U tom slučaju ponaša se slično kao FedAvg, skaliran na samo jedan optimizacijski smjer.

Također se iz rezultata može zaključiti da postoji veza između stupnja različitosti (engl. *discrepancy*) lokalnih i globalnih težina i preciznosti modela na testnom skupu. Većinom vrijedi što je manji stupanj različitosti to je veća preciznost modela. Može se vidjeti na tablici 4.1 i tablici 4.2.

FlexCFL i IFCA algoritmi imaju slične performase (FlexCFL ima prednost od +2.1% MNIST, +2.7% FEMNIST, +4.1% FMNIST), no IFCA algoritmu potrebno je više komunikacijskih rundi da konvergira, također je FlexCFL manje zahtjevan u pogledu opterećenja mrežnog prometa u odnosu na IFCA [4].

Model	**FlexCFL**	**FedAvg**	**FeSEM**	**IFCA**
MNIST-MLP	**0.92** / **0.96**	0.86 / 0.93	0.21 / 0.81	0.43 / 0.95
MNIST-CNN	0.85 / **0.98**	**0.89** / 0.97	0.20 / 0.83	0.39 / 0.95
FEMNIST-MLP	**0.80** / **0.90**	0.66 / 0.80	0.23 / 0.49	0.73 / 0.86
FEMNIST-CNN	**0.84** / **0.98**	0.72 / 0.96	0.10 / 0.49	0.75 / 0.97
FMNIST-MLP	**0.84** / **0.95**	0.70 / 0.87	0.23 / 0.48	0.72 / 0.87
FMNIST-CNN	**0.89** / **0.98**	0.82 / 0.96	0.36 / 0.82	0.82 / **0.98**

Tablica 4.1: Test-preciznosti za korištene algoritme, format (preciznost u 50. komunikacijskoj rundi / maksimalna preciznost)

Na temelju rezultata može se zaključiti da korišteni CNN modeli daju bolje rezultate od MLP modela.

Također vjerojatno nije potrebno 300 komunikacijskih rundi za treniranje modela korištenjem opisivanih algoritama (osim FeSEM). Moguće je vidjeti da test-preciznost skoro pa eksponencijalno raste u prvih 50 komunikacijskih rundi, a zatim sporo konvergira prema maksimalnoj preciznosti, prije pojave prenaučenosti. Kod FeSEM algoritma pogreška na skupu podataka za treniranje konstantno pada, a test-preciznost polako konvergira prema maksimalnoj. Očito je da je potrebno povećati broj komunikacijskih rundi da bi se postigla konvergencija.

Algoritme je moguće modificirati tako da se klijenti mogu prebacivati, migrirati u drugu grupu, ako njihova lokalna distribucija više odgovara drugoj grupi. Za FlexCFL

Model	FlexCFL	FedAvg	FeSEM	IFCA
MNIST-MLP	**0.29** / **0.18**	0.48 / 0.41	1.77 / 1.17	1.34 / 0.58
MNIST-CNN	0.43 / **0.21**	**0.41** / 0.37	1.65 / 0.68	1.09 / 0.37
FEMNIST-MLP	**0.07** / **0.05**	0.20 / 0.17	0.26 / 0.21	0.09 / 0.06
FEMNIST-CNN	**0.14** / **0.07**	0.27 / 0.23	0.26 / 0.34	0.18 / 0.18
FMNIST-MLP	**0.13** / **0.11**	0.24 / 0.22	0.30 / 0.37	0.16 / 0.19
FMNIST-CNN	**0.06** / 0.03	0.18 / 0.08	0.29 / 0.23	0.10 / **0.02**

Tablica 4.2: Stupnjevi različitosti težina za korištene algoritme, format (stupanj različitosti u 50. komunikacijskoj rundi / - za maksimalnu test-preciznost)

također je moguće više grupa agregirati u jednu, ako su im distribucije podataka kongruentne, odnosno postoje veze između gradijenata. U nastavku je prikazana usporedba performansi kad se koriste migracija i agregacija u odnosu na *vanilla* algoritme.

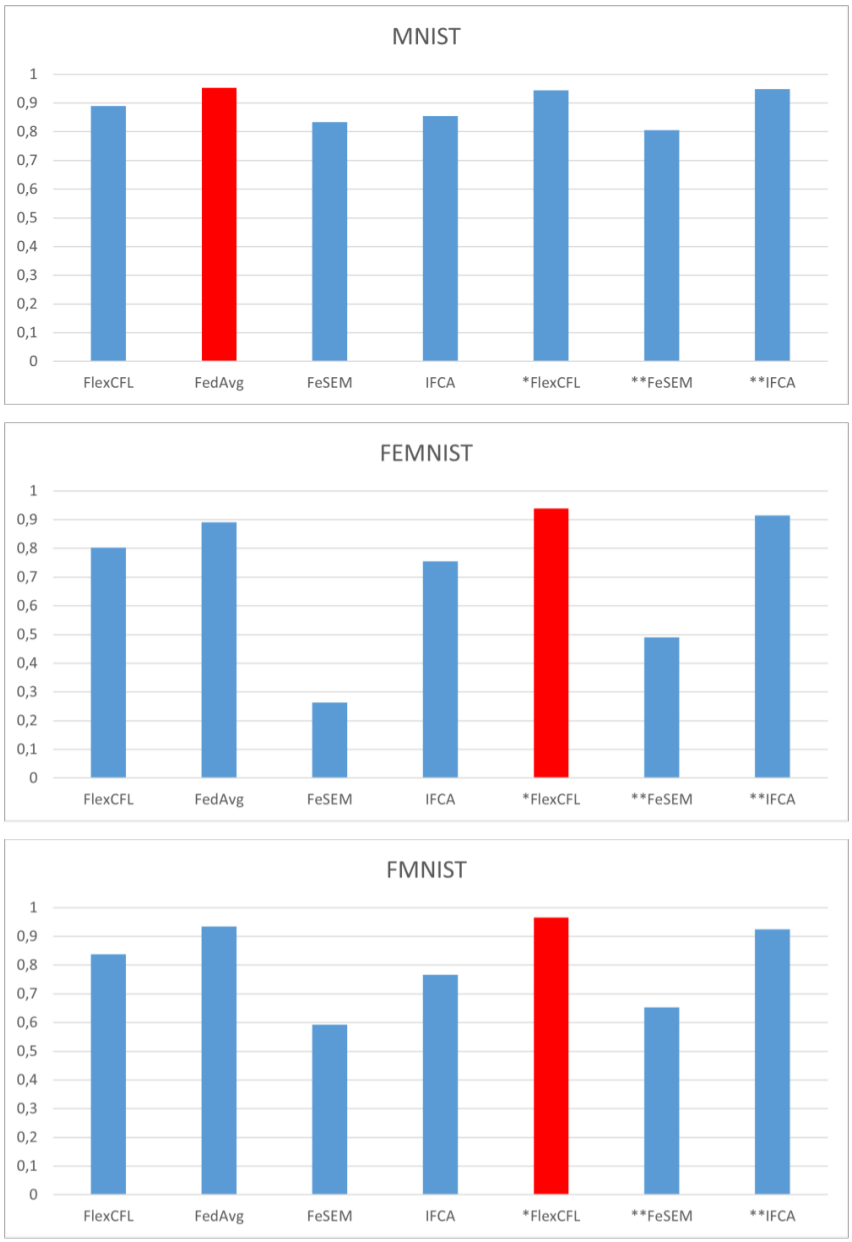
Model	FlexCFL	*FlexCFL	FeSEM	**FeSEM	IFCA	**IFCA
MNIST-MLP	0.85	**0.92**	**0.84**	0.81	0.84	**0.95**
MNIST-CNN	0.93	**0.97**	**0.83**	0.80	0.87	**0.95**
FEMNIST-MLP	0.69	**0.91**	0.36	**0.49**	0.63	**0.86**
FEMNIST-CNN	0.91	**0.97**	0.17	**0.49**	0.88	**0.97**
FMNIST-MLP	0.76	**0.95**	0.38	**0.48**	0.63	**0.87**
FMNIST-CNN	0.91	**0.98**	0.80	**0.82**	0.90	**0.98**

Tablica 4.3: Usporedba performansi algoritama s uključenom mogućnosti agregacije grupa odnosno migracije klijenata (* i ** respektivno)

Važno je napomenuti da su težine modela fiksirane tako da usporedba algoritama bude što više deterministička. Također je uvedena dodatna promjena u lokalnoj distribuciji klastera na način da postoji vjerojatnost od 0.05 da se u komunikacijskoj rundi promijene dvije jedinstvene oznake dvojice različitih klijenata. Time se pokušava što više simulirati statistička heterogenost podataka s ciljem oponašanja tokova podataka iz stvarnog svijeta.

Iz tablice 4.3 može se zaključiti da za gotove sve modele algoritmi koji koriste dinamičke postavke raspodjele klijenata u klastere imaju bolje performase od statičkih (prosječna test preciznost za **MNIST**: *FlexCFL +6.1%; **FeSEM -3.4%; **IFCA +11.0%, **FEMNIST**: *FlexCFL +17.0%; **FeSEM +86.7%; **IFCA +21.2%, **FMNIST**: *FlexCFL +15.3%; **FeSEM +10.0%; **IFCA +20.6%).

Grafovima 4.19 prikazane su prosječne maksimalne test-preciznosti nad MNIST, FEMNIST i FMNIST skupovima.



Slika 4.19: Prikaz maksimalnih test-preciznosti algoritama nad skupovima MNIST, FEMNIST i FMNIST (prosjek MLP i CNN modela)

5. Zaključak

Klasični pristup strojnom učenju bio je prevesti podatke klijenata s njihovih uređaja na centralni server koji bi zatim provodio treniranje modela. Međutim, tim pristupom ugrožava se privatnost korisnika, jer su se podaci prenosili u izvornom obliku. Kao rješenje na taj problem pojavio se novi pristup raspodijeljenom strojnom učenju — federalno učenje. Očuvanje privatnosti osigurano je na način da klijenti lokalno treniraju iste modele dobivene od centralnog servera, zatim gradijente šalju natrag serveru koji ih agregira pomoću nekog algoritma i osvježava globalni model. Također federalno učenje bolje skalira s većim brojem klijenata jer je efikasnije od klasičnog načina.

Iako federalno učenje pruža zaštitu privatnosti uz očuvanje performansi klasičnog pristupa, performanse se znatno smanjuju ako su podaci klijenata nekongruentni, statistički heterogeni. Kao rješenje javlja se princip podjele nekongruentnih skupova podataka u različite grupe. Svaka grupa bi time trenirala svoj model nad podacima homogenih distribucija. Pokazuje se da federalno učenje temeljeno na grupama pod takvim uvjetima pokazuje bolje performanse od klasičnog federalnog učenja. U ovom radu upravo to se željelo pokazati, testirajući i uspoređujući razne algoritme federalnog učenja. Rezultati pokazuju da algoritam FlexCFL uz agregaciju grupa ima u prosjeku najbolje test-preciznosti nad FEMNIST i Fashion MNIST skupovima podataka, dok algoritam FedAvg najbolje ima nad MNIST skupom podataka.

Sljedeći korak u istraživanju predstavljao bi testiranje algoritama nad pravim statistički heterogenim podacima. U ovom radu algoritmi su se testirali nad skupovima MNIST, FEMNIST i Fashion MNIST koji su prilagođeni tako da simuliraju nekongruentnost, no pokazuje se da je to moguće samo na razini pojedinačnih grupa klijenata, a ne na razini cijelog skupa. Kad bi koristili podatke npr. senzora vlage u zraku i predviđali hoće li padati kiša ti podaci ne bi nužno bili kongruentni.

Federalno učenje relativno je nova pojava u svijetu strojnog učenja i pristup se tek počinje implementirati u praktičnom svijetu. Vrlo vjerojatno je da će zbog svojih prednosti očuvanja privatnosti i skalabilnosti prevladati nad klasičnim principom raspodijeljenog učenja.

LITERATURA

[1] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, i Vitaly Shmatikov. How to backdoor federated learning, 2019.

[2] Nathalie Baracaldo i Hayim Shaul. Federated learning meets homomorphic encryption, Dec 2022. URL https://research.ibm.com/blog/federated-learning-homomorphic-encryption. IBM Research Blog.

[3] Wenliang Du, Yunghsiang Han, i Shigang Chen. Privacy-preserving multivariate statistical analysis: Linear regression and classification. 04 2004. doi: 10.1137/1.9781611972740.21.

[4] Moming Duan, Duo Liu, Xinyuan Ji, Yu Wu, Liang Liang, Xianzhang Chen, Yujuan Tan, i Ao Ren. Flexible clustered federated learning for client-level data distribution shift. *IEEE Transactions on Parallel and Distributed Systems*, stranica 1–1, 2021. ISSN 2161-9883. doi: 10.1109/tpds.2021.3134263. URL http://dx.doi.org/10.1109/TPDS.2021.3134263.

[5] Avishek Ghosh, Jichan Chung, Dong Yin, i Kannan Ramchandran. An efficient framework for clustered federated learning, 2021.

[6] Flower Labs GmbH. What is federated learning? https://flower.ai/docs/framework/tutorial-series-what-is-federated-learning.html. zadnji pristup: 4.5.2024.

[7] Abdul Hamid Halabi. What is federated learning? http://blogs.nvidia.com/blog/what-is-federated-learning/. zadnji pristup: 4.5.2024.

[8] Dasaradharami Reddy Kandati i Thippa Reddy Gadekallu. Genetic clustered federated learning for covid-19 detection. *Electronics*, 11(17), 2022. ISSN 2079-

9292. doi: 10.3390/electronics11172714. URL https://www.mdpi.com/2079-9292/11/17/2714.

[9] Guodong Long, Ming Xie, Tao Shen, Tianyi Zhou, Xianzhi Wang, i Jing Jiang. Multi-center federated learning: clients clustering for better personalization. *World Wide Web*, 26(1):481–500, Lipanj 2022. ISSN 1573-1413. doi: 10.1007/s11280-022-01046-x. URL http://dx.doi.org/10.1007/s11280-022-01046-x.

[10] H. Brendan McMahan, Eider Moore, Daniel Ramage, i Blaise Agüera y Arcas. Federated learning of deep networks using model averaging. *CoRR*, abs/1602.05629, 2016. URL http://arxiv.org/abs/1602.05629.

[11] Luca Melis, Congzheng Song, Emiliano De Cristofaro, i Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning, 2018.

[12] Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletari, Holger Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N. Galtier, Bennett A. Landman, Klaus H. Maier-Hein, Sébastien Ourselin, Micah J. Sheller, Ronald M. Summers, Andrew Trask, Daguang Xu, Maximilian Baust, i M. Jorge Cardoso. The future of digital health with federated learning. *CoRR*, abs/2003.08119, 2020. URL https://arxiv.org/abs/2003.08119.

[13] Felix Sattler, Klaus-Robert Müller, i Wojciech Samek. Clustered federated learning: Model-agnostic distributed multi-task optimization under privacy constraints. *CoRR*, abs/1910.01991, 2019. URL http://arxiv.org/abs/1910.01991.

[14] Qiang Yang, Yang Liu, Tianjian Chen, i Yongxin Tong. Federated machine learning: Concept and applications, 2019.

[15] Hangyu Zhu, Jinjin Xu, Shiqing Liu, i Yaochu Jin. Federated learning on non-iid data: A survey, 2021.

Federalno učenje u uređajima na rubu temeljeno na grupama

Sažetak

U ovom radu istražuju se osnovni koncepti federalnog učenja. Definira se pojam federalnog učenja i njegova podjela. Predstavlja se način federalnog učenja temeljen na grupama. Grupe se sastoje od klijenata s homogenim distribucijama podataka, nad kojima se zasebno provodi učenje. Ispituju se performanse različitih algoritama federalnog učenja FedAvg, FlexCFL, FeSEM i IFCA nad skupovima podataka MNIST, FEMNIST, Fashion MNIST. Skupovi su modificirani tako da simuliraju statističku heterogenost koja se pojavljuje u praktičnoj primjeni. Komentiraju se rezultati testiranja i daju ideje za budući nastavak rada.

Ključne riječi: Federalno učenje, Federalno učenje temeljeno na grupama, Strojno učenje, FedAvg, FlexCFL, MNIST

Group-based federated learning on edge devices

Abstract

In this paper, the basic concepts of federated learning are explored. The concept of federated learning and its classification are defined. A method of clustered federated learning is presented. The groups consist of clients with homogeneous data distributions, over which learning is conducted separately. The performance of various federated learning algorithms FedAvg, FlexCFL, FeSEM, and IFCA is examined on the datasets MNIST, FEMNIST, and Fashion MNIST. The datasets have been modified to simulate the statistical heterogeneity that occurs in practical applications. The test results are discussed, and ideas for future work are provided.

Keywords: Federated learning, Clustered federated learning, Machine learning, FedAvg, FlexCFL, MNIST