

besm.jpg

iut\_logo.png

دانشگاه صنعتی اصفهان  
دانشکده مهندسی برق و کامپیوتر

## بهبود کارایی الگوریتم یادگیری فدرال برای داده‌های غیرمستقل و غیریکنواخت با در نظر گرفتن میزان شباهت بین شبکه‌های عصبی در دستگاه‌های نهایی

پایان‌نامه کارشناسی ارشد مهندسی کامپیوتر - هوش مصنوعی و رباتیکز

علی بزرگزاد

استاد راهنما

دکتر امیر خورسندی

کلیه حقوق مالکیت مادی و معنوی مربوط به این پایان نامه متعلق به دانشگاه صنعتی اصفهان و پدیدآورندگان است. این حقوق توسط دانشگاه صنعتی اصفهان و بر اساس خط مشی مالکیت فکری این دانشگاه، ارزش گذاری و سهم بندی خواهد شد. هر گونه بهره برداری از محتوا، نتایج یا اقدام برای تجاری سازی دستاوردهای این پایان نامه تنها با مجوز کتبی دانشگاه صنعتی اصفهان امکان پذیر است.

## فهرست مطالب

<u>صفحه</u>	<u>عنوان</u>
چهار	فهرست مطالب
۱	چکیده

### فصل اول: مقدمه

۲	۱-۱ شناخت موضوع
۳	۱-۱-۱ یادگیری متمرکز
۳	۲-۱-۱ یادگیری غیرمتمرکز
۳	۳-۱-۱ یادگیری توزیع شده
۳	۲-۱ یادگیری فدرال
۵	۳-۱ تاریخچه یادگیری فدرال
۵	۴-۱ کاربرد یادگیری فدرال
۶	۱-۴-۱ یادگیری فدرال در شهر هوشمند
۷	۲-۴-۱ یادگیری فدرال در بیمارستان
۷	۳-۴-۱ یادگیری فدرال در فروشگاه برنامه‌های کاربردی تلفن همراه
۸	۵-۱ انگیزه و هدف پژوهش
۹	۶-۱ مروری بر روند ارائه مطالب پایان نامه

### فصل دوم: مفاهیم پایه و پیشینه پژوهش در یادگیری فدرال

۱۱	۱-۲ مقدمه
۱۲	۲-۲ چارچوب ریاضی یادگیری فدرال
۱۲	۱-۲-۲ مفاهیم پایه در یادگیری ماشین و یادگیری عمیق
۱۳	۲-۲-۲ فرمول‌های پایه در یادگیری عمیق
۱۴	۳-۲-۲ ارتباط مفاهیم یادگیری عمیق با یادگیری فدرال
۱۴	۴-۲-۲ بیان ریاضی یادگیری فدرال
۱۶	۳-۲ چالش‌های موجود در یادگیری فدرال و نگاه کلی مقالات به آن‌ها
۱۶	۱-۳-۲ چالش تبادل داده
۱۷	۲-۳-۲ چالش ناهمگنی‌های سیستمی
۱۷	۳-۳-۲ چالش ناهمگنی‌های آماری

۱۸	۴-۳-۲ چالش حریم شخصی
۱۹	۴-۲ رویکردهای کلی و پایه‌ای در حل چالش‌ها
۱۹	۱-۴-۲ الگوریتم میانگین‌گیری فدرال (FedAvg)
۲۲	۲-۴-۲ بهینه‌سازی FedProx
۲۳	۵-۲ جمع‌بندی

### فصل سوم: بررسی اختصاصی پیشینه روش‌های حل مشکل ناهمگنی آماری

۲۵	۱-۳ مقدمه
۲۶	۲-۳ نگرش برپایه داده
۲۶	۱-۲-۳ اشتراک‌گذاری داده
۲۸	۲-۲-۳ بهبود داده
۲۹	۳-۲-۳ انتخاب داده
۲۹	۳-۳ نگرش برپایه مدل
۳۰	۱-۳-۳ تجمع و به‌روزرسانی مدل
۳۱	۲-۳-۳ بهینه‌سازی تطبیقی
۳۱	۳-۳-۳ بهینه‌سازی منظم
۳۲	۴-۳ نگرش برپایه چهارچوب
۳۳	۱-۴-۳ خوشه‌بندی مشابهت
۳۴	۲-۴-۳ تقطیر دانش
۳۴	۳-۴-۳ لایه‌های شخصی‌سازی
۳۵	۵-۳ نگرش برپایه الگوریتم
۳۵	۱-۵-۳ فرایادگیری
۳۶	۲-۵-۳ یادگیری چندوظیفه‌ای
۳۷	۳-۵-۳ یادگیری مادام‌العمر
۳۸	۶-۳ جمع‌بندی

### فصل چهارم: روش پیشنهادی برای جابه‌جایی مدل‌های شبکه عصبی بین کاربران

۳۹	۱-۴ مقدمه
۴۰	۲-۴ روش جابه‌جایی فدرال (FedSwap)
۴۳	۳-۴ نحوه جابه‌جایی مدل‌ها در یادگیری فدرال
۴۳	۱-۳-۴ روش جابه‌جایی فدرال به صورت تصادفی
۴۳	۲-۳-۴ روش پیشنهادی جابه‌جایی فدرال بر پایه شباهت (SimFedSwap)
۴۵	۴-۴ تعریف معیار مشابهت
۴۵	۵-۴ پایداری در معیارهای مشابهت
۴۶	۱-۵-۴ پایداری نسبت به تبدیل متعامد
۴۷	۲-۵-۴ پایداری نسبت به مقیاس‌بندی یکسان

۴۷	۶-۴ مقایسه ساختارهای مشابهت
۴۸	۱-۶-۴ ضرب داخلی
۴۸	۲-۶-۴ انتخاب هسته
۴۹	۷-۴ معیارهای سنجش مشابهت
۵۰	۱-۷-۴ قرینه مجموع اختلاف مطلق (OSAD)
۵۱	۲-۷-۴ تحلیل همبستگی کانونی (CCA)
۵۲	۳-۷-۴ معیار استقلال هیلبرت-اشمیت (HSIC)
۵۳	۴-۷-۴ هم‌ترازی هسته مرکزی (CKA)
۵۳	۵-۷-۴ هم‌ترازی هسته مرکزی بدون مداخله (dCKA)
۵۶	۸-۴ شاخص مشابهت بین شبکه‌های عصبی
۵۶	۹-۴ بررسی تأثیرات جابه‌جایی مدل‌ها
۵۷	۱-۹-۴ تأثیرات جابه‌جایی مدل‌ها بر ترافیک شبکه
۵۸	۲-۹-۴ تأثیر جابه‌جایی مدل‌ها بر حریم شخصی در ارتباط بین سرور و کاربران
۵۹	۳-۹-۴ تأثیر جابه‌جایی مدل‌ها بر حریم شخصی با توجه به واسطه بودن سرور
۶۰	۱۰-۴ نحوه تعیین کاربران نهایی جهت جابه‌جایی مدل‌ها در روش SimFedSwap
۶۰	۱-۱۰-۴ روش جابه‌جایی حریصانه (GS)
۶۲	۲-۱۰-۴ مرتبه زمانی روش جابه‌جایی حریصانه
۶۳	۳-۱۰-۴ روش جابه‌جایی حداقل شباهت (MSS)
۶۴	۴-۱۰-۴ مرتبه زمانی و نحوه پیاده‌سازی روش جابه‌جایی حداقل شباهت
۶۶	۱۱-۴ جمع‌بندی

## فصل پنجم: پیاده‌سازی و بررسی نتایج

۶۸	۱-۵ مقدمه
۶۸	۲-۵ پیاده‌سازی مدل‌های شبکه عصبی
۶۹	۱-۲-۵ مدل MLP
۶۹	۲-۲-۵ مدل CNN
۷۱	۳-۵ مجموعه داده MNIST
۷۳	۴-۵ مجموعه داده CIFAR-10
۷۶	۵-۵ مجموعه داده CINIC-10
۷۹	۶-۵ مجموعه داده FEMNIST
۸۱	۱-۶-۵ رویکردهای پایه در مجموعه داده FEMNIST
۸۱	۲-۶-۵ مقایسه نتایج در رویکرد کلاس‌بندی (FEMNISTclass)
۸۳	۳-۶-۵ مقایسه نتایج در رویکرد نویسندگان (FEMNISTwriter)
۸۴	۷-۵ مقایسه جابه‌جایی حریصانه با جابه‌جایی حداقل شباهت در روش SimFedSwap
۸۵	۸-۵ تحلیل کاهش تعداد کاربران در هر دور و افزایش تعداد کل دورها در روش SimFedSwap

۸۷	۹-۵ بررسی نحوه پیاده‌سازی کد، سخت‌افزار مورد استفاده و زمان اجرای کدها
۸۸	۱۰-۵ جمع‌بندی

## فصل ششم: نتیجه‌گیری و پیشنهادها

۹۰	۱-۶ نتیجه‌گیری
۹۲	۲-۶ پیشنهادها

## پیوست اول: بررسی نمودارهای خطا

۹۳	آ-۱ مقایسه روش SimFedSwap با روش‌های پایه
۹۳	آ-۱-۱ مجموعه داده MNIST
۹۴	آ-۱-۲ مجموعه داده CIFAR-10
۹۵	آ-۱-۳ مجموعه داده CINIC-10
۹۵	آ-۱-۴ مجموعه داده FEMNIST
۹۷	آ-۲ مقایسه جابه‌جایی حریصانه با جابه‌جایی حداقل شباهت در روش SimFedSwap
۹۸	آ-۳ تحلیل کاهش تعداد کاربران در هر دور و افزایش تعداد کل دورها در روش SimFedSwap
۱۰۰	مراجع

## چکیده

در عصر حاضر، با پیشرفت سریع فناوری، افزایش تعداد دستگاه‌های متصل به اینترنت و نقش فزاینده هوش مصنوعی و یادگیری ماشین، اهمیت برقراری ارتباطات مؤثر و حفاظت از حریم شخصی کاربران بیش از پیش احساس می‌شود. این ضرورت به توسعه روش‌های توزیع‌شده‌ای مانند یادگیری فدرال انجامیده است که از جمله راهکارهای پیشرفته در این حوزه به‌شمار می‌رود. در یادگیری فدرال، داده‌ها به‌جای ارسال به یک سرور مرکزی، در همان دستگاه‌های نهایی باقی می‌مانند و مدل‌ها به‌صورت محلی آموزش داده می‌شوند. سپس این مدل‌ها با هم ترکیب می‌شوند تا یک مدل جامع ایجاد شود. این روش نه تنها نیاز به انتقال داده‌ها را کاهش می‌دهد، بلکه به حفظ بهتر حریم شخصی کاربران نیز کمک می‌کند. با این حال، یادگیری فدرال با چالش‌های زیادی روبه‌رو است که یکی از آن‌ها ناهمگنی آماری داده‌ها می‌باشد، به این معنی که داده‌های موجود در دستگاه‌های مختلف می‌توانند بسیار متنوع و متفاوت از یکدیگر باشند. این ناهمگنی باعث می‌شود که مدل‌های محلی نتوانند تمامی ویژگی‌های داده‌ها را به‌خوبی یاد بگیرند و در نتیجه، مدل جامع نیز به خوبی همگرا نشود. بنابراین، دستیابی به یک مدل جامع با عملکرد مناسب ممکن است دشوار شود. در این راستا، ارائه روش‌هایی برای مقابله با ناهمگنی آماری از اهمیت بالایی برخوردار هستند. روش‌های پیشنهادی باید علاوه بر تمرکز بر حل این مشکل، از جنبه‌های محاسباتی، ارتباطی و حفظ حریم شخصی نیز پایداری خود را حفظ کنند. یکی از راهکارهای پیشنهادی برای مقابله با این چالش، جابه‌جایی مدل‌های شبکه عصبی بین کاربران نهایی در طول فرآیند یادگیری است. این کار باعث می‌شود مدل‌های محلی با داده‌های متنوع‌تری مواجه شوند و در نتیجه، مدل جامع به همگرایی بهتری برسد. در روش‌های معمول، جابه‌جایی مدل‌ها به‌صورت تصادفی انجام می‌شود. اما در این پژوهش پیشنهاد شده است که به‌جای روش تصادفی، این جابه‌جایی به‌صورت هوشمند و بر اساس معیارهای شباهت صورت گیرد. به این ترتیب، مدل‌هایی که کمترین شباهت را با هم دارند، جابه‌جا می‌شوند. این رویکرد باعث می‌شود مدل‌ها با داده‌هایی روبه‌رو شوند که کمتر با آن‌ها آشنا هستند و این امر می‌تواند به بهبود همگرایی مدل جامع منجر شود. از جنبه دیگر، این پژوهش به بررسی تأثیر جابه‌جایی مدل‌ها بر حفظ حریم شخصی کاربران پرداخته است. روش‌های معمول جابه‌جایی مدل‌ها، به‌طور مستقیم بین کاربران نهایی انجام می‌شوند. اگرچه این روش می‌تواند سربار شبکه را کاهش دهد، اما ممکن است به تضعیف حریم شخصی کاربران منجر شود. در این پژوهش پیشنهاد شده است که سرور مرکزی به عنوان واسطه‌ای در فرآیند جابه‌جایی عمل کند. با این روش، حفظ حریم شخصی کاربران بهتر تضمین می‌شود و پیاده‌سازی تکنیک‌های مختلف این حوزه نیز ساده‌تر خواهد شد. در نهایت، این پژوهش نشان می‌دهد که جابه‌جایی هوشمندانه مدل‌های شبکه عصبی بر اساس معیارهای شباهت، می‌تواند فرآیند همگرایی مدل جامع را تسریع کند. این روش به‌ویژه در شرایطی که تعداد کاربران زیاد است، تأثیر بیشتری دارد و توانسته است نتایج را حدود ۱٪ بهبود بخشد.

**کلمات کلیدی:** ۱- یادگیری فدرال، ۲- یادگیری توزیع‌شده، ۳- یادگیری عمیق، ۴- شباهت در شبکه عصبی، ۵- ناهمگنی آماری



# فصل اول

## مقدمه

### ۱-۱ شناخت موضوع

در سال‌های اخیر، به دلیل پیشرفت‌های سریع فناوری و دسترسی آسان به اینترنت، بسیاری از دستگاه‌ها به اینترنت متصل شده‌اند. این پدیده که به اینترنت اشیا<sup>۱</sup> معروف است، شامل انواع دستگاه‌ها از جمله دستگاه‌های پوشیدنی<sup>۲</sup>، خودروهای خودران، خانه‌های هوشمند<sup>۳</sup> و به ویژه تلفن‌های هوشمند<sup>۴</sup> می‌شود. این دستگاه‌ها به‌طور چشمگیری زندگی روزمره انسان‌ها را دگرگون کرده‌اند. استفاده از این سیستم‌ها همگی باعث تولید حجم قابل توجهی داده در طول روز می‌شوند که شرکت‌های بزرگ فناوری از این داده‌ها بهره برده و با استفاده از آن‌ها اقدام به ارائه انواع سرویس به کاربران خود می‌نمایند. همچنین این روند سبب ظهور فناوری هوش مصنوعی و موارد مرتبط با آن شده است.

با پیشرفت علم هوش مصنوعی و استفاده گسترده از روش‌های یادگیری ماشین، امکان بهره‌برداری بهینه از حجم عظیم داده‌های تولید شده فراهم گردیده است. این داده‌ها می‌توانند برای اجرای الگوریتم‌های مختلف به منظور دستیابی به اهداف متنوع به کار گرفته شوند. روش‌های متعددی برای مدیریت و اجرای این الگوریتم‌های

---

<sup>۱</sup>Internet of Things

<sup>۲</sup>Wearable Devices

<sup>۳</sup>Smart Homes

<sup>۴</sup>Smart Phones

یادگیری وجود دارد که در ادامه به توضیح هر یک پرداخته خواهد شد.

### ۱-۱-۱ یادگیری متمرکز<sup>۱</sup>

در روش یادگیری متمرکز که در بسیاری از سیستم‌های امروزی به کار می‌رود، تمامی گره‌ها<sup>۲</sup> اطلاعات خود را به‌طور کامل به یک سرور دهنده ابری<sup>۳</sup> ارسال می‌کنند و سرور دهنده با دسترسی به تمامی داده‌ها، الگوریتم‌های مورد نیاز را اجرا می‌کند. این روش به دلیل تمرکز داده‌ها در یک مکان مرکزی، امکان اجرای دقیق و هماهنگ الگوریتم‌ها را فراهم می‌کند، اما وابستگی به سرور دهنده مرکزی می‌تواند منجر به مشکلات امنیتی و چالش‌های مقیاس‌پذیری شود [۱]. این روش در شکل ۱-۱ (آ) به تصویر کشیده شده است.

### ۲-۱-۱ یادگیری غیرمتمرکز<sup>۴</sup>

در روش یادگیری غیرمتمرکز، هر گره به‌طور مستقل الگوریتم‌های مورد نیاز خود را اجرا کرده و پس از چند مرحله، اطلاعات به‌روزرسانی‌شده را با گره‌های همسایه به اشتراک می‌گذارد. این فرآیند تا زمان دستیابی به همگرایی کامل بین گره‌ها ادامه پیدا می‌کند. این روش با بهره‌گیری از استقلال گره‌ها، باعث افزایش مقیاس‌پذیری و مقاومت در برابر خرابی‌های سیستم می‌شود، اما در عین حال، ممکن است به تبادل مکرر اطلاعات و تأخیر در دستیابی به همگرایی منجر شود [۲]. این روش در شکل ۱-۱ (ب) به نمایش درآمده است.

### ۳-۱-۱ یادگیری توزیع شده<sup>۵</sup>

در روش یادگیری توزیع‌شده، یک هسته مرکزی مدیریت کل سیستم و داده‌ها را بر عهده دارد، اما برای کاهش فشار پردازشی، این بار را بین گره‌های موجود توزیع می‌کند. این تقسیم بار پردازشی منجر به سرعت و کارایی بیشتر در فرآیند آموزش می‌شود و امکان استفاده همزمان از منابع متنوع برای تحلیل داده‌های بزرگ را فراهم می‌آورد. با این حال، وابستگی به هسته مرکزی ممکن است نقطه‌ضعفی باشد که می‌تواند به بروز مشکلاتی در صورت خرابی یا اختلال در عملکرد آن منجر شود. این روش در شکل ۱-۱ (ج) نشان داده شده است.

### ۲-۱ یادگیری فدرال<sup>۶</sup>

سیستم‌های متمرکز تا پیش از این بیشتر نیازها را برطرف می‌کردند، اما در دنیای امروزی و با افزایش تعداد دستگاه‌های متصل، چالش‌های جدیدی مطرح شده است. هزینه‌های بالای ناشی از انتقال حجم زیاد داده‌ها از

<sup>1</sup>Centralized Learning

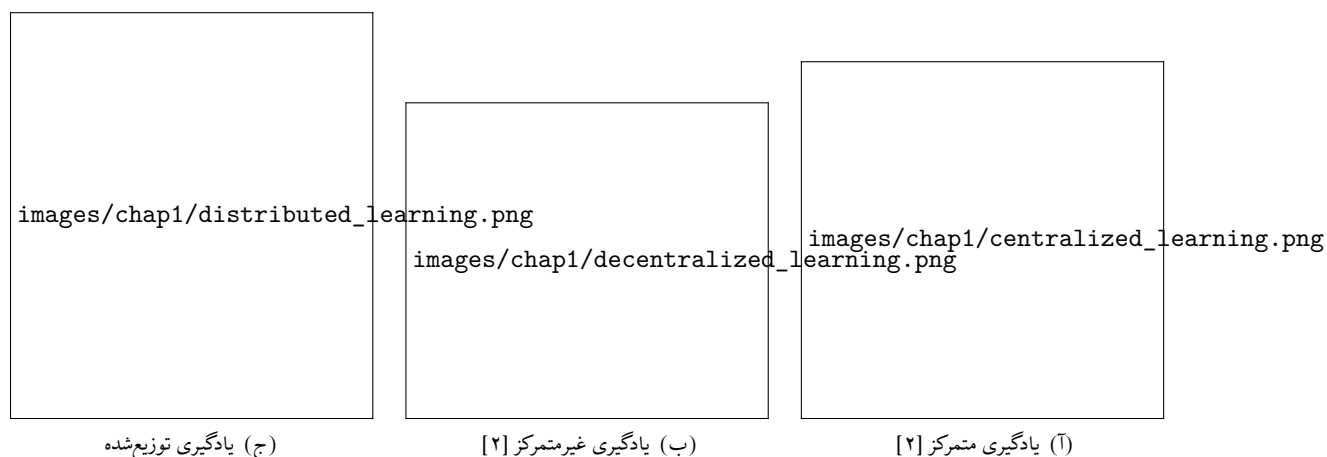
<sup>2</sup>Nodes

<sup>3</sup>Cloud Server

<sup>4</sup>Decentralized Learning

<sup>5</sup>Distributed Learning

<sup>6</sup>Federated Learning



شکل ۱-۱: انواع روش‌های یادگیری.

یک جهت، و افزایش نگرانی‌ها درباره امنیت اطلاعات حساس و شخصی از جهت دیگر، محققان را به سمت استفاده از الگوریتم‌های غیرمتمرکز و توزیع شده در حوزه یادگیری ماشین سوق داده است. یکی از جدیدترین زیرمجموعه‌های مهم و پرکاربرد روش‌های یادگیری توزیع شده، یادگیری فدرال است که بسیار مورد توجه قرار گرفته است.

در روش یادگیری فدرال، برخلاف رویکردهای متمرکز یادگیری ماشین، تجزیه و تحلیل داده‌ها به دستگاه‌های لبه<sup>۱</sup> یا کاربران<sup>۲</sup> منتقل می‌شوند [۳]. این روش، به عنوان یک جایگزین مطلوب برای مدل‌سازی داده‌ها در محیط‌هایی با تعداد زیادی کاربر معرفی شده است. در این چارچوب، به جای انتقال داده‌های اصلی، پارامترهای مدل‌های محلی در هر مرحله از فرآیند آموزش به سمت سرور منتقل می‌شوند، که این امر زمینه‌ساز بهبود امنیت و کاهش هزینه‌های ارتباطی می‌باشد. در شکل ۱-۲ این معماری به نمایش گذاشته شده است.

سرور در حقیقت نقش رهبری را ایفا می‌کند و با توجه به نوع داده‌ها، یک مدل شبکه عصبی<sup>۳</sup> ایجاد کرده و آن را به سمت کاربران ارسال می‌کند. در ادامه کاربران با توجه به داده‌های خود شبکه را آموزش می‌دهند و بعد از چند بار تکرار به صورت محلی، وزن‌های به روزرسانی شده را به سمت سرور بر می‌گردانند. همان‌طور که در شکل ۱-۲ مشاهده می‌شود، داده‌ها همگی در سمت کاربران قرار گرفته‌اند و به سمت سرور ارسال نمی‌شوند. عدم اجبار در به اشتراک گذاشتن اطلاعات گره‌ها در یادگیری فدرال، کمک شایانی به حفظ حریم شخصی کاربران می‌کند [۴].

<sup>۱</sup>Edge Devices

<sup>۲</sup>Clients

<sup>۳</sup>Neural Network

images/chap1/federated\_learning.png

شکل ۱-۲: یادگیری فدرال [۳].

### ۳-۱ تاریخچه یادگیری فدرال

در اوایل فصل بهار سال ۲۰۱۷، محققان گوگل (Google) برای اولین بار موضوع یادگیری فدرال را در یک مطلب کوتاه در وبلاگ هوش مصنوعی خود معرفی کردند. این مطلب با عنوان «یادگیری فدرال: یادگیری ماشین اشتراکی، بدون نیاز به آموزش متمرکز داده‌ها» منتشر شد [۵]. در این نوشته، به‌طور مختصر از Google Keyboard یا به اختصار Gboard صحبت شد که با بهره‌گیری از یادگیری فدرال، قابلیت پیش‌بینی و پیشنهاد لغت بعدی به کاربر را دارد. با استفاده از یادگیری فدرال، دیگر نیازی به ارسال داده‌های کاربران به سرور نبود و مدل به صورت محلی به‌روزرسانی می‌شد.

این روش با استفاده از اطلاعات فراوان ذخیره شده در دستگاه‌ها، خدمات بهتری را ارائه می‌دهد، بدون این که داده‌های حساس به سرور ارسال شوند و حریم شخصی کاربران به خطر نیفتد. در شکل ۱-۳، نحوه استفاده از یادگیری فدرال در این برنامه به نمایش درآمده است.

### ۴-۱ کاربرد یادگیری فدرال

سامانه‌های متمرکز سنتی که تنها مسئول جمع‌آوری، پایش و کنترل شرایط به‌صورت محلی بودند، اکنون جای خود را به دستگاه‌های هوشمندی داده‌اند که قابلیت پردازش و برنامه‌ریزی داده‌ها را در سطح سیار و سیستمی

images/chap1/gboard.png

شکل ۱-۳: استفاده از یادگیری فدرال برای پیش‌بینی کلمه بعدی در Gboard [۶].

دارند. علاوه بر این، گسترش ارتباطات مبتنی بر اینترنت، امکان انتقال و تبادل داده‌ها بین سیستم‌های مختلف را فراهم کرده است. این تحولات منجر به کاهش نیاز به تصمیم‌گیری متمرکز و توسعه سیستم‌های کنترل و پایش پیشرفته شده است. این ویژگی‌ها، همراه با حجم روزافزون داده‌ها، یادگیری فدرال را به یکی از بهترین روش‌ها برای توسعه سیستم‌های هوشمند تبدیل کرده است [۷]. در ادامه، سه نمونه از کاربردهای یادگیری فدرال شرح داده خواهد شد.

#### ۱-۴-۱ یادگیری فدرال در شهر هوشمند<sup>۱</sup>

در یک شهر هوشمند، اطلاعات جمع‌آوری شده از حسگرهای مختلف مانند داده‌های ترافیک، مصرف انرژی، پسماند شهری و رویدادهای امنیتی، ارزش بالایی دارند و به عنوان منبعی کلیدی برای بهبود عملکرد شهر هوشمند و ارتقای کیفیت زندگی شهروندان محسوب می‌شوند. اما در کنار این مزایا، حفظ حریم شخصی و امنیت اطلاعات شهروندان نیز از اهمیت بالایی برخوردار است. در این‌جا یادگیری فدرال به عنوان یک رویکرد نوین که مبتنی بر حفظ حریم شخصی است، به کار گرفته می‌شود.

اگرچه در یک شهر هوشمند، سازمان‌های مختلف هر کدام اطلاعات خاص خود را دارند، اما با تجمیع این اطلاعات می‌توان مدیریت بهتری انجام داد. یادگیری فدرال با حفظ حریم شخصی کاربران، این امکان را فراهم می‌کند که سازمان‌ها بدون نیاز به اشتراک‌گذاری داده‌های حساس خود با یکدیگر، از تمامی داده‌های موجود بهره‌برداری کنند و مدل‌های هوش مصنوعی و الگوریتم‌های بهبود عملکرد شهر هوشمند را توسعه دهند. به عنوان مثال، با استفاده از یادگیری فدرال می‌توان بهبود مدیریت ترافیک، بهینه‌سازی مصرف انرژی، کاهش آلودگی هوا و افزایش امنیت شهری را تحقق بخشید، در حالی که حریم شخصی شهروندان به بهترین شکل حفظ می‌شود.

<sup>۱</sup> Smart City

#### ۱-۴-۲ یادگیری فدرال در بیمارستان

در یک بیمارستان، اطلاعات پزشکی دارای طبقه‌بندی حساس و مهم هستند و باید به صورت محرمانه نگهداری شوند. با این حال، بهره‌برداری از این داده‌ها برای ارتقاء خدمات بهداشتی و درمانی بسیار ارزشمند است. در این شرایط، یادگیری فدرال می‌تواند نقش مهمی ایفا کند. با استفاده از روش‌های یادگیری فدرال، بیمارستان‌ها می‌توانند از داده‌های پزشکی بیماران خود برای توسعه مدل‌هایی استفاده کنند که به بهبود خدمات، ارتقاء روش‌های تشخیص و درمان بیماری‌ها و افزایش بهره‌وری پزشکان کمک می‌کنند، بدون این که نیاز باشد این داده‌ها به طور مستقیم به یک مرکز جمع‌آوری اطلاعات ارسال شوند.

به عبارت دیگر، یادگیری فدرال این امکان را فراهم می‌کند که مدل‌های هوش مصنوعی روی داده‌های محلی بیماران در هر بیمارستان آموزش ببینند و بیماری‌ها را شناسایی و تشخیص دهند. این فرایند به بهبود درمان‌ها کمک می‌کند، بدون این که اطلاعات حساس بیماران به بیرون درز کند و امنیت آن‌ها حفظ می‌شود. در شکل ۱-۴ شمای کلی استفاده از یادگیری فدرال در بیمارستان‌ها به نمایش درآمده است.

#### ۱-۴-۳ یادگیری فدرال در فروشگاه برنامه‌های کاربردی<sup>۱</sup> تلفن همراه

یک فروشگاه مجازی برنامه‌های کاربردی تلفن همراه را در نظر بگیرید که به کاربران امکان دریافت و نصب برنامه‌های مختلف را می‌دهد. این فروشگاه می‌خواهد با استفاده از داده‌های کاربران خود، الگوریتمی را توسعه

images/chap1/hospital.png

شکل ۱-۴: کاربرد یادگیری فدرال در بیمارستان [۸].

<sup>۱</sup> App Store

دهد که بتواند به‌طور دقیق‌تری برنامه‌های مورد علاقه کاربران را به آن‌ها پیشنهاد دهد. اگر این فروشگاه از روش‌های متمرکز استفاده کند، باید داده‌های حساس و شخصی کاربران را جمع‌آوری و تحلیل کند که این موضوع می‌تواند نگرانی‌های جدی در مورد حریم شخصی کاربران ایجاد کند و غیرعملی باشد.

در حالی که با استفاده از یادگیری فدرال، این فروشگاه می‌تواند الگوریتم خود را بر روی داده‌های محلی هر کاربر اجرا کند. به این ترتیب، هیچ داده حساسی به یک مرکز جمع‌آوری داده‌ها ارسال نمی‌شود و حریم شخصی کاربران حفظ می‌گردد. به عنوان مثال، اگر یک کاربر به برنامه‌های موسیقی علاقه‌مند باشد، الگوریتم محلی در تلفن هوشمند او می‌تواند این الگو را شناسایی کند و پیشنهادهای مربوط به برنامه‌های موسیقی را ارائه دهد، بدون این که نیاز به ارسال داده‌های شخصی و حساس او به سرور فروشگاه باشد.

## ۱-۵ انگیزه و هدف پژوهش

در یادگیری فدرال، توزیع داده‌ها و پردازش آن‌ها در سیستم‌های مختلف و مستقل، چالش‌های زیادی را ایجاد می‌کند. یکی از این چالش‌ها، کاهش تبادل داده‌ها به‌منظور حفظ حریم شخصی کاربران است. علاوه بر این، تفاوت‌های سیستمی و تنوع داده‌ها بین کاربران مختلف نیز فرآیند آموزش مدل‌ها را پیچیده می‌کند. بنابراین، برای بهبود کارایی مدل‌های فدرال و مقابله با این مشکلات، نیاز به روش‌های خلاقانه و مؤثر احساس می‌شود. همان‌گونه که بیان شد، یکی از چالش‌های اصلی، مواجهه با داده‌های بسیار گوناگون و متفاوتی است که در بین کاربران مختلف پراکنده شده‌اند. این پراکندگی منجر به کندی فرایند همگرایی مدل‌های یادگیری می‌شود. به‌طور معمول، داده‌های موجود در یک دستگاه با داده‌های دستگاه‌های دیگر تفاوت‌های قابل توجهی دارند و این تفاوت‌ها باعث می‌شود تا مدل سراسری نتواند به‌سرعت به یک دقت مطلوب دست یابد. این مشکل باعث شده است تا پژوهشگران به دنبال راهکارهایی باشند که بتوانند این فرایند را تسریع کرده و مدل سراسری را به دقت بالاتری برسانند [۶].

یکی از روش‌های پیشنهادی برای حل این مشکل، جابه‌جایی وزن‌های شبکه‌های عصبی بین کاربران نهایی در طول فرایند یادگیری است [۹]. این جابه‌جایی مدل‌ها بین کاربران، می‌تواند به تغییر وزن‌ها منجر شده و به مدل‌ها کمک کند تا با داده‌های متفاوتی آشنا شوند. به عبارت دیگر، این روش از طریق فراهم کردن امکان آشنایی با داده‌های جدید می‌تواند همگرایی مدل سراسری را بهبود بخشد و مدل را به سمت مقادیر بهتری هدایت کند.

با این حال، در روش‌های قبلی، این جابه‌جایی‌ها به‌طور تصادفی انجام می‌شدند. این تصادفی بودن ممکن است باعث شود که مدل‌ها با داده‌هایی مواجه شوند که قبلاً با آن‌ها آشنا بوده‌اند یا تغییرات ایجاد شده در وزن‌ها

نتوانند بهبود قابل توجهی در دقت مدل ایجاد کنند. بنابراین، این سوال مطرح می شود که آیا می توان به جای استفاده از جابه جایی تصادفی، این فرایند را به صورت هوشمندانه تری انجام داد، به این صورت که با استفاده از معیارهای مشخص، مدل هایی که بیشترین تفاوت را با یکدیگر دارند شناسایی و با هم مبادله شوند.

پژوهش حاضر بر آن است که پاسخ این پرسش را مورد بررسی قرار داده و روشی مناسب در این رابطه ارائه نماید. در روش پیشنهادی سعی خواهد شد، دو مدلی که کمترین میزان شباهت را با یکدیگر دارند، جابه جا شوند. این رویکرد بر این اساس استوار است که مدل هایی که شباهت زیادی به یکدیگر دارند، نشان دهنده این هستند که کاربران نهایی آن ها داده های نسبتاً مشابهی دارند. در مقابل، مدل هایی که شباهت کمتری با یکدیگر دارند، در واقع با داده های متفاوت تری آموزش دیده اند. جابه جایی مدل هایی که داده های متفاوتی را مشاهده کرده اند، می تواند به این معنا باشد که این مدل ها می توانند با داده های جدیدی آشنا شوند که پیش تر با آن ها روبه رو نشده بودند. این آشنایی با داده های جدید می تواند به شبکه عصبی سراسری کمک کند تا بهتر با داده های مختلفی که توسط کاربران گوناگون تولید می شوند، تطابق پیدا کند. در نتیجه، شبکه عصبی سراسری می تواند سریع تر و با دقت بیشتری به همگرایی نهایی برسد.

در نتیجه می توان این طور بیان کرد که این پژوهش روش هوشمندانه ای برای جابه جایی مدل ها بر پایه شباهت پیشنهاد می دهد که می تواند به عنوان راهکاری مؤثر در تسریع همگرایی یادگیری فدرال مورد استفاده قرار گیرد. این روش نه تنها باعث بهبود کیفیت مدل سراسری می شود، بلکه کارایی آن را در برخورد با داده های مختلف کاربران نیز افزایش می دهد.

## ۱-۶ مروری بر روند ارائه مطالب پایان نامه

در فصل دوم، ابتدا مفاهیم پایه ای در یادگیری ماشین و یادگیری عمیق توضیح داده شده و ارتباط آن ها با یادگیری فدرال بررسی می شود. همچنین چارچوب ریاضی مربوط به یادگیری فدرال ارائه می گردد. پس از آن، چالش های موجود در این حوزه تحلیل شده و دیدگاه های مختلف مقالات در این زمینه مرور می شوند. در پایان، راهکارهای اساسی ارائه شده برای حل این چالش ها مورد بحث قرار می گیرند.

فصل سوم به بررسی جامع پیشینه روش های حل مشکل توزیع متفاوت و گوناگون داده ها میان کاربران مختلف در شبکه اختصاص دارد. این بررسی از منظر داده، مدل، چارچوب و الگوریتم انجام می شود.

در فصل چهارم، روش جابه جایی مدل ها معرفی شده و نحوه جابه جایی تصادفی و نیز روش جابه جایی بر پایه شباهت بررسی می گردند. سپس تأثیر این جابه جایی ها بر ترافیک شبکه و حریم شخصی تحلیل می شود. در ادامه معیارهای شباهت و پایداری آن ها نیز مورد بررسی قرار گرفته و تعدادی شاخص برای سنجش شباهت معرفی



می‌گردد. در نهایت، روش‌های مختلف برای تعیین کاربران نهایی جهت جابه‌جایی مدل‌ها توضیح داده می‌شود. در فصل پنجم، ابتدا به توضیح مدل‌های شبکه عصبی که در این پژوهش پیاده‌سازی شده‌اند، پرداخته می‌شود. سپس، انواع مجموعه داده‌های مورد استفاده معرفی و روش جابه‌جایی بر پایه شباهت با سایر روش‌های مرجع در این مجموعه داده‌ها مقایسه می‌شوند. علاوه بر این، تاثیر روش‌های مختلف جابه‌جایی، تغییر تعداد کاربران و تاثیر تعداد دورها نیز به‌طور دقیق تحلیل خواهند شد. در پایان، فصل ششم به نتیجه‌گیری کلی و ارائه پیشنهادهایی برای ادامه پژوهش اختصاص خواهد داشت.

## فصل دوم

### مفاهیم پایه و پیشینه پژوهش در یادگیری فدرال

#### ۱-۲ مقدمه

توزیع داده‌ها بین کاربران در یادگیری فدرال می‌تواند با چالش‌ها و مشکلات مختلفی مواجه شود. یکی از مسائل مهم، تفاوت‌ها و ناسازگاری‌هایی است که ممکن است در طول فرآیند آموزش بین کاربران یا دستگاه‌های مختلف به وجود بیاید. این اختلافات می‌توانند ناشی از تفاوت در کیفیت داده‌ها، سرعت پردازش یا حتی محیط‌های مختلفی باشند که هر دستگاه در آن قرار دارد. همچنین، تأخیر در ارسال و دریافت داده‌ها بین کاربران و سرور مرکزی نیز می‌تواند چالش دیگری باشد که عملکرد کلی سیستم را تحت تأثیر قرار دهد [۶].

اگر این چالش‌ها پیش از آغاز فرآیند مدل‌سازی به درستی شناسایی نشده و راه‌حل‌های مناسبی برای آن‌ها اتخاذ نشود، مدل نهایی احتمالاً با مشکلاتی همچون کاهش دقت و عملکرد روبه‌رو خواهد شد. این مسئله یکی از بزرگترین موانع در مسیر یادگیری فدرال است و نیازمند دقت و استفاده از روش‌های خلاقانه برای حل آن خواهد بود.

در این فصل، ابتدا به بیان چارچوب ریاضی یادگیری فدرال پرداخته می‌شود که البته برای درک آن نیاز به آشنایی پایه با مفاهیم ریاضی در یادگیری ماشین و یادگیری عمیق است. سپس چالش‌های موجود در یادگیری فدرال بررسی شده و دیدگاه‌های مختلف مقالات علمی در مورد هر یک از این چالش‌ها به صورت کلی مرور

می‌شوند. در نهایت، به رویکردهای پایه‌ای جهت حل این چالش‌ها اشاره خواهد شد.

## ۲-۲ چارچوب ریاضی یادگیری فدرال

برای تشریح چارچوب ریاضی یادگیری فدرال، ابتدا باید مفاهیم اساسی یادگیری ماشین و یادگیری عمیق را بررسی کرد. در ادامه با مرتبط کردن این اصول به یادگیری فدرال، می‌توان به‌طور دقیق ریاضیات اولیه در یادگیری فدرال را توضیح داد و نشان داد که چگونه این مفاهیم در این حوزه خاص به کار گرفته می‌شوند.

### ۱-۲-۲ مفاهیم پایه در یادگیری ماشین و یادگیری عمیق

یادگیری ماشین بخشی از هوش مصنوعی است که به سیستم‌ها امکان می‌دهد تا از داده‌ها یاد بگیرند و پیش‌بینی‌هایی انجام دهند، بدون این که از قبل به‌صورت دقیق برنامه‌ریزی شده باشند [۱۰]. در یادگیری ماشین، الگوریتم‌ها با تحلیل داده‌های ورودی و ویژگی‌های استخراج شده توسط انسان، مدل‌هایی ایجاد می‌کنند که قادر به شناسایی الگوها و روابط پیچیده در داده‌های دیده نشده هستند. این فرآیند به کامپیوترها امکان می‌دهد تا با تجربه و مشاهده، بهبود پیدا کنند و بتوانند وظایفی مانند تشخیص تصویر، پردازش زبان طبیعی و پیش‌بینی بازار را انجام دهند.

در بین روش‌های متنوع یادگیری ماشین، یادگیری عمیق یک زیرمجموعه از یادگیری ماشین است که از شبکه‌های عصبی مصنوعی برای مدل‌سازی و یادگیری از داده‌ها استفاده می‌کند [۱۰]. این روش‌ها از لایه‌های متعدد برای استخراج ویژگی‌ها و یادگیری الگوها در داده‌های پیچیده بهره می‌برند. شبکه‌های عصبی عمیق، که شامل چندین لایه پنهان<sup>۱</sup> هستند، قادر به یادگیری ویژگی‌های سطح بالا از داده‌های ورودی می‌باشند. این لایه‌ها به ترتیب اطلاعات را پردازش کرده و به یکدیگر منتقل می‌کنند تا خروجی نهایی تولید شود. جهت درک بهتر این مفاهیم به شکل ۱-۲ توجه نمایید.

یادگیری عمیق از الگوریتم‌های بهینه‌سازی برای تنظیم وزن‌های شبکه عصبی بهره می‌برد. یکی از این الگوریتم‌ها، گرادیان نزولی<sup>۲</sup> است که با تعیین شیب تابع هزینه<sup>۳</sup>، وزن‌ها را به‌طور مکرر به‌روزرسانی می‌کند تا به کمترین مقدار ممکن برای این تابع برسد [۱۲]. الگوریتم انتشار به عقب<sup>۴</sup> یکی از مهم‌ترین روش‌ها در این زمینه است که از گرادیان نزولی برای بهینه‌سازی وزن‌ها استفاده می‌کند [۱۲]. در این فرآیند، ابتدا خطای خروجی شبکه محاسبه می‌شود و سپس این خطا به‌صورت معکوس از لایه خروجی به سمت لایه‌های ورودی منتقل می‌شود تا وزن‌ها تنظیم شوند و شبکه به دقت مطلوب دست یابد.

<sup>۱</sup>Hidden Layer

<sup>۲</sup>Gradient Descent

<sup>۳</sup>Loss Function

<sup>۴</sup>Backpropagation

images/chap2/machine\_learning\_vs\_deep\_learning.png

شکل ۲-۱: تفاوت یادگیری ماشین و یادگیری عمیق [۱۱].

## ۲-۲-۲ فرمول‌های پایه در یادگیری عمیق

- تابع هزینه و انتشار به عقب

تابع هزینه یا تابع خطا معیاری است که اختلاف بین خروجی پیش‌بینی شده و مقدار واقعی را اندازه‌گیری می‌کند. یکی از توابع هزینه رایج، میانگین مربعات خطا<sup>۱</sup> (MSE) است:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^m (y_i - \hat{y}_i)^2 \quad (۱-۲)$$

که در آن  $y_i$  مقدار واقعی،  $\hat{y}_i$  مقدار پیش‌بینی شده و  $m$  تعداد نمونه‌ها است. الگوریتم انتشار به عقب از این تابع هزینه استفاده می‌کند تا وزن‌ها را به‌روزرسانی کند. این فرآیند شامل محاسبه گرادیان‌ها و به‌روزرسانی وزن‌ها در جهت کاهش خطا است.

- بهینه‌سازی با گرادیان نزولی

بهینه‌سازی با گرادیان نزولی یکی از رایج‌ترین روش‌ها برای به‌روزرسانی وزن‌های شبکه عصبی است. رابطه به‌روزرسانی وزن‌ها به‌صورت زیر است:

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j} \quad (۲-۲)$$

<sup>۱</sup> Mean Squared Error

که در آن  $\theta_j$  وزن،  $\alpha$  نرخ یادگیری و  $\frac{\partial J(\theta)}{\partial \theta_j}$  مشتق جزئی تابع هزینه نسبت به وزن  $\theta_j$  است. این فرآیند به اندازه‌ای تکرار می‌شود که تابع هزینه به حداقل مقدار خود برسد.

## ۳-۲-۲ ارتباط مفاهیم یادگیری عمیق با یادگیری فدرال

یادگیری فدرال از همان مفاهیم پایه‌ای یادگیری عمیق و شبکه‌های عصبی بهره می‌برد، اما از ساختاری توزیع‌شده استفاده می‌کند که در آن داده‌ها بین چندین دستگاه تقسیم شده‌اند. در یادگیری فدرال، مدل‌های یادگیری عمیق به‌صورت محلی بر روی دستگاه‌های کاربران آموزش داده می‌شوند و تنها به‌روزرسانی‌های مدل به سرور مرکزی ارسال می‌شود. این روش، علاوه بر حفظ حریم شخصی، امکان استفاده از داده‌های گسترده و متنوع را فراهم می‌کند. الگوریتم‌های بهینه‌سازی مانند گرادیان نزولی و انتشار به عقب به‌صورت محلی اجرا شده و نتایج این به‌روزرسانی‌ها به‌صورت تجمیعی برای بهبود مدل کلی به کار می‌روند. این ویژگی، یادگیری فدرال را به یک رویکرد قدرتمند برای مدل‌سازی در محیط‌های توزیع‌شده تبدیل می‌کند.

## ۴-۲-۲ بیان ریاضی یادگیری فدرال

هدف اصلی یادگیری فدرال، یافتن مجموعه‌ای از پارامترهای مدل است که عملکرد کلی آن را بر روی داده‌های توزیع‌شده بین تعداد زیادی دستگاه بهینه کند. هر دستگاه دارای داده‌های محلی است و یک تابع هزینه محلی بر اساس این داده‌ها برای آن دستگاه تعریف می‌شود. مسئله بهینه‌سازی کلی در یادگیری فدرال به دنبال کمینه کردن مجموع وزنی این توابع هزینه محلی است تا یک مدل جامع و یکپارچه حاصل شود. در ادامه چارچوب ریاضی این مسئله با استفاده از مطالب برگرفته شده و روابط ریاضی مندرج در [۱۳] بیان خواهد شد.

در این چارچوب، فرض بر این است که یک مجموعه ثابت از  $K$  کاربر وجود دارد که هر کدام دارای یک مجموعه داده محلی ثابت هستند. در ابتدای هر دوره، یک زیرمجموعه تصادفی شامل  $C$  کاربر انتخاب می‌شود و سرور وضعیت فعلی پارامترهای مدل سراسری را به هر یک از این کاربرها ارسال می‌کند. سپس هر کاربر انتخاب شده بر اساس وضعیت سراسری و مجموعه داده محلی خود محاسبات محلی را انجام می‌دهد و یک به‌روزرسانی به سرور ارسال می‌کند. در ادامه سرور این به‌روزرسانی‌ها را بر روی وضعیت سراسری خود اعمال می‌کند و این فرآیند تکرار می‌شود [۱۳].

در حالی که تمرکز بر اهداف شبکه عصبی غیرمحدب<sup>۱</sup> است، چارچوب مورد بررسی برای هر هدف جمع-

<sup>۱</sup>non-Convex

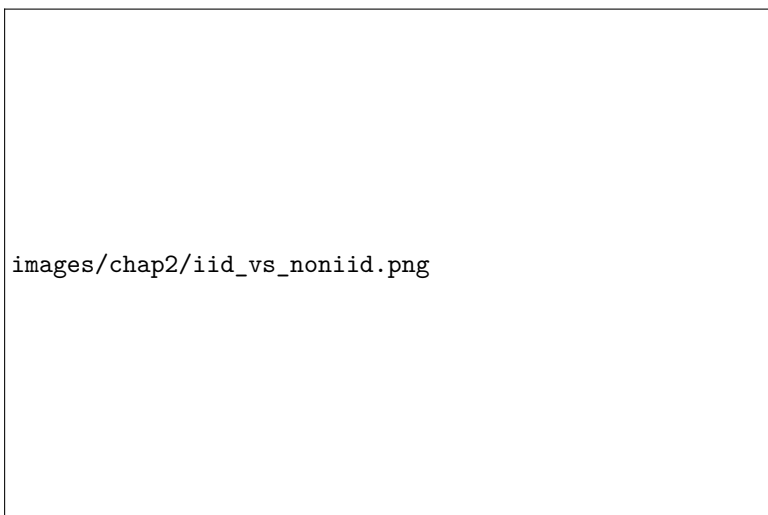
متناهی<sup>۱</sup> به صورت زیر قابل اعمال خواهد بود.

$$\min_{w \in \mathbb{R}^d} f(w) \quad \text{where} \quad f(w) \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^n f_i(w) \quad (۳-۲)$$

برای یک مسئله یادگیری ماشین، معمولاً  $f_i(w) = \ell(x_i, y_i; w)$  در نظر گرفته می شود، به این معنی که این تابع نشان دهنده خطای پیش بینی بر روی نمونه  $(x_i, y_i)$  با استفاده از پارامترهای مدل  $w$  است. فرض می شود که داده ها بین  $K$  کاربر تقسیم شده اند، که در آن مجموعه ای از نقاط داده مربوط به کاربر  $k$  است و  $n_k = |\mathcal{P}_k|$  تعداد این نقاط داده را نشان می دهد. بنابراین، می توان  $f(w)$  در رابطه (۳-۲) را به صورت زیر بازنویسی نمود:

$$f(w) = \sum_{k=1}^K \frac{n_k}{n} F_k(w) \quad \text{where} \quad F_k(w) = \frac{1}{n_k} \sum_{i \in \mathcal{P}_k} f_i(w) \quad (۴-۲)$$

اگر مجموعه  $\mathcal{P}_k$  به صورت تصادفی و با توزیع یکنواخت<sup>۲</sup> از نمونه های آموزشی بین کاربرها تشکیل شده باشد، در این صورت  $\mathbb{E}_{\mathcal{P}_k} [F_k(w)]$  برابر  $f(w)$  خواهد بود، به این معنا که امید ریاضی بر روی مجموعه مثال های اختصاص داده شده به یک کاربر خاص، محاسبه می شود. این همان فرض استقلال و توزیع یکنواخت داده ها<sup>۳</sup> (IID) است که عموماً توسط الگوریتم های بهینه سازی توزیع شده استفاده می شود. در این جا حالتی که فرض مذکور برقرار نیست (یعنی  $F_k$  می تواند تقریباً به هر میزانی از  $f$  فاصله داشته باشد)، به عنوان حالت غیرمستقل و غیریکنواخت (non-IID) شناخته می شود [۱۳]. جهت درک بهتر به شکل ۲-۲ توجه نمایید.



شکل ۲-۲: تفاوت ساختار IID با non-IID [۱۴].

<sup>۱</sup>Finite-Sum

<sup>۲</sup>Uniform Distribution

<sup>۳</sup>Independent and Identically Distributed

## ۲-۳ چالش‌های موجود در یادگیری فدرال و نگاه کلی مقالات به آن‌ها

با وجود این که یادگیری فدرال نسبت به روش‌های سنتی یادگیری ماشین دارای مزایای قابل توجهی می‌باشد، اما به دلیل ساختار شبکه‌ای خود با چالش‌های متعددی روبه‌رو است. در ادامه، مهم‌ترین چالش‌های یادگیری فدرال مورد بررسی قرار گرفته و به دیدگاه‌های کلی مقالات علمی در مورد این چالش‌ها اشاره می‌شود.

### ۲-۳-۱ چالش تبادل داده

تبادل داده بین سرور و کاربران به دلیل محدودیت‌های پهنای باند و منابع موجود در ارتباطات شبکه‌ای، معمولاً هزینه‌بر است. یکی از دلایل اصلی این هزینه‌بر بودن، حجم بالای داده‌هایی است که باید میان دستگاه‌های کاربران و سرور منتقل شود. این مشکل با افزایش تعداد انتقال‌ها در یک دوره زمانی خاص، تشدید می‌گردد. همچنین، با پیچیده‌تر شدن مدل‌ها و افزایش تعداد پارامترها، اندازه این مدل‌ها نیز به‌طور قابل توجهی افزایش می‌یابد [۱۵].

همچنین، مشارکت تعداد زیادی از دستگاه‌های کاربران نهایی در فرآیند آموزش مدل‌ها، می‌تواند هزینه‌های ارتباطی را به‌طور قابل توجهی افزایش دهد. از سوی دیگر، به علت مشکلات ارتباطی، در بسیاری از مواقع تمامی دستگاه‌ها در هر چرخه از فرآیند آموزش حضور ندارند. این موضوع باعث ایجاد پیچیدگی‌های مرتبط با انتقال داده‌ها می‌شود، زیرا زمانی که تنها بخشی از دستگاه‌ها در فرآیند آموزش شرکت می‌کنند، هماهنگی، انتخاب کاربران و تجمیع نتایج از این دستگاه‌ها نیازمند مدیریت پیچیده‌تری است و این نیز منجر به افزایش هزینه‌های کلی می‌شود.

در این رابطه استفاده از فشرده‌سازی داده‌ها می‌تواند هزینه‌های ارتباطی را به میزان قابل توجهی کاهش دهد [۱۶]. لذا برای مدیریت هزینه‌های بالای ارتباطات در فرآیند یادگیری فدرال، روش‌هایی ارائه شده‌اند که بر فشرده‌سازی داده‌های ارسالی از دستگاه‌های نهایی به سرور مرکزی تمرکز دارند.

در [۱۷] روشی به نام PCFL<sup>۱</sup> ارائه شده است که از نظر ارتباطی بسیار کارآمد بوده و شامل سه عنصر اصلی می‌باشد. این عناصر شامل فشرده‌سازی دوطرفه، فشرده‌سازی مکانی وزن‌ها و یک پروتکل پیشرفته برای حفظ حریم خصوصی داده‌ها هستند. فشرده‌سازی دوطرفه، داده‌ها را در دو مرحله، هم قبل از ارسال از دستگاه‌های نهایی به سرور و هم هنگام ارسال نتایج به‌روزرسانی شده از سرور به دستگاه‌ها، فشرده می‌کند تا حجم داده‌های انتقالی کاهش یابد. فشرده‌سازی مکانی وزن‌ها نیز با فشرده کردن وزن‌های مدل، حجم انتقال را کاهش داده و کارایی ارتباطات را بهبود می‌بخشد. پروتکل حفظ حریم خصوصی داده‌ها نیز امنیت اطلاعات کاربران را در

<sup>۱</sup> Privacy Communication efficient Federated Learning

طول فرآیند یادگیری فدرال تضمین می‌کند. این عناصر با همکاری هم، موجب کاهش هزینه‌های ارتباطی و بهبود کارایی در روش PCFL می‌شوند.

### ۲-۳-۲ چالش ناهمگنی‌های سیستمی<sup>۱</sup>

در دنیای یادگیری فدرال، دستگاه‌ها از نظر حافظه، توان محاسباتی و ارتباطات بسیار با یکدیگر متفاوت هستند. این تفاوت‌ها ممکن است از اختلافاتی مانند تفاوت در پردازنده، نوع حافظه، نوع اتصال شبکه و نیاز به انرژی ناشی شود. این ناهمگنی‌های سیستمی می‌تواند منجر به مشکلاتی در زمان‌بندی و همگام‌سازی دستگاه‌ها شود. به‌عنوان مثال، دستگاه‌هایی با توان محاسباتی بالا ممکن است سریع‌تر از سایرین آموزش را به پایان برسانند، در حالی که دستگاه‌های ضعیف‌تر نیاز به زمان بیشتری دارند. این عدم هماهنگی می‌تواند منجر به تأخیر در فرآیند یادگیری و کاهش کارایی کلی سیستم شود.

این تفاوت‌های سیستمی، یکی از چالش‌های یادگیری فدرال محسوب می‌شوند و برای رفع این مشکلات، روش‌های یادگیری فدرال باید توانایی پیش‌بینی دقیق تعداد دستگاه‌هایی که در هر فرآیند شرکت می‌کنند را داشته باشند. همچنین، باید بتوانند در برابر دستگاه‌هایی که در حین عملیات دچار مشکل شده و از دسترس خارج می‌شوند، مقاومتی مناسب داشته باشند [۶].

برای مقابله با ناهمگنی سیستمی، روشی تحت عنوان تعادل در به‌روزرسانی مدل مطرح شده است [۱۸]. در این روش، وزن‌دهی به نمونه‌ها بر اساس میزان نیاز به آموزش در هر دستگاه صورت می‌گیرد. این کار باعث می‌شود که دستگاه‌های با حجم داده کمتر، وزن بیشتری در به‌روزرسانی مدل داشته باشند. در رویکرد دیگری به نام یادگیری فعال، دستگاه‌هایی که مدل‌های خود را به سرور ارسال می‌کنند، فعالیت خود را به نحوی تنظیم می‌کنند که مدل از داده‌های مهم‌تر و کمتر دیده شده بیشتر یاد می‌گیرد [۱۶]. این روش می‌تواند به تعادل در آموزش مدل کمک کند و از ناهمگنی سیستمی جلوگیری کند.

### ۳-۳-۲ چالش ناهمگنی‌های آماری<sup>۲</sup>

روش‌های مختلفی برای تولید و جمع‌آوری داده‌ها بین دستگاه‌ها وجود دارد. این داده‌ها معمولاً به‌صورت مستقل از هم تولید نمی‌شوند و بین آن‌ها ارتباطات و پیوندهایی وجود دارد. برای مثال فرض کنید یک شرکت فناوری از یادگیری فدرال برای بهبود مدل پیش‌بینی متن در تلفن‌های هوشمند کاربران استفاده می‌کند. این شرکت داده‌های خود را از منابع مختلف جمع‌آوری کرده و کاربران در این سیستم می‌توانند به زبان‌ها و سبک‌های نوشتاری متنوعی از جمله فارسی، انگلیسی و اسپانیایی مطالب خود را ثبت کنند. همچنین، برخی کاربران بیشتر

<sup>1</sup>Systems Heterogeneity

<sup>2</sup>Statistical Heterogeneity



از شکلک‌ها<sup>۱</sup> استفاده می‌کنند، در حالی که دیگران بیشتر از اصطلاحات تخصصی شغلی یا علمی بهره می‌برند. چنین الگویی از تولید داده با فرضیات استقلال و توزیع یکنواخت داده‌ها (IID) در مسائل بهینه‌سازی در تضاد است، که منجر به پیچیدگی‌هایی در فرآیند مدل‌سازی، تحلیل نظری و ارزیابی عملکرد می‌شود. بنابراین، با وجود هدف نهایی که یادگیری یک مدل جامع و یکپارچه است، روش‌های جایگزین مانند یادگیری چندوظیفه‌ای<sup>۲</sup> و فرایادگیری<sup>۳</sup> به عنوان راه‌حل‌های ممکن مطرح شده‌اند [۶].

یک روش برای حل مشکل ناهمگنی آماری در یادگیری فدرال استفاده از رویکرد ترکیبی یا ترکیب روش‌های یادگیری محلی است [۱۸]. در این رویکرد، به جای استفاده از یک الگوریتم یادگیری مشترک برای تمام دستگاه‌ها، از چندین الگوریتم یادگیری محلی با تنوع مدل‌ها و تنظیمات مختلف استفاده می‌شود. سپس، اطلاعات مدل‌های محلی روی سرور یا گره مرکزی جمع‌آوری می‌شود و با استفاده از ترکیب این اطلاعات، یک مدل یادگیری مشترک به‌روزرسانی خواهد شد.

## ۲-۳-۴ چالش حریم شخصی

همان‌گونه که در ۱-۲ اشاره شد، حفظ حریم شخصی یک مزیت مهم در یادگیری فدرال به شمار می‌رود اما این ویژگی در صورت عدم کنترل مناسب می‌تواند به یک چالش تبدیل شود. در این رابطه لازم به یادآوری است که در یادگیری فدرال، تنها مدل‌های به‌روز شده یا گرادیان‌های آن‌ها به سرور مرکزی ارسال می‌شوند و داده‌های خام کاربران هرگز از دستگاه‌های محلی خارج نمی‌شوند. به عبارتی، تنها وزن‌های شبکه‌های عصبی که حاوی اطلاعات مستقیم شناسایی فردی نیستند، مبادله می‌شوند. با این وجود، همچنان چالش‌هایی در حفظ حریم شخصی وجود دارد.

مدل‌های به‌روزرسانی شده ممکن است به صورت غیرمستقیم حاوی الگوهایی از داده‌های حساس باشند که در فرآیند آموزش استفاده شده‌اند. این الگوها می‌توانند توسط مهاجمان تحلیل شده و اطلاعاتی در مورد داده‌های کاربران را افشا کنند. برای مثال، حملات بازسازی داده یا حملات استنتاج می‌توانند با دسترسی به وزن‌های به‌روزرسانی شده، داده‌های اولیه را تا حدی بازسازی کنند. بنابراین، حتی اگر داده‌های خام مستقیماً ارسال نشوند، نیاز به تدابیر امنیتی برای جلوگیری از این نوع حملات وجود دارد. در نتیجه، اگرچه یادگیری فدرال حریم شخصی را بهتر حفظ می‌کند، اما همچنان نیازمند روش‌های پیشرفته امنیتی مانند رمزنگاری، تنظیم نویز در داده‌ها و روش‌های دیگر برای تضمین حفاظت از حریم شخصی کاربران است.

<sup>1</sup>Emoji

<sup>2</sup>Multi-Tasking

<sup>3</sup>Meta Learning

روش حفظ حریم خصوصی تفاضلی<sup>۱</sup> با افزودن نویز به نتایج محاسبات یا به داده‌های ورودی، اطمینان حاصل می‌کند که حضور یا عدم حضور یک نمونه داده خاص در مجموعه داده‌ها، تأثیر قابل توجهی بر خروجی محاسبات نداشته باشد. این روش به ویژه برای حفظ حریم خصوصی در یادگیری فدرال مفید است زیرا از افشای اطلاعات حساس از طریق پارامترهای مدل جلوگیری می‌کند [۱۹].

یک روش دیگر، رویکرد رمزنگاری هم‌شکل<sup>۲</sup> است که امکان محاسبه روی داده‌های رمزنگاری شده را بدون نیاز به رمزگشایی آن‌ها فراهم می‌کند. این رویکرد به ویژه در یادگیری فدرال برای حفظ حریم خصوصی داده‌ها در حین انجام محاسبات مفید است زیرا نیاز به تغییر ماهیت داده نبوده و چون جابه‌جایی در یادگیری فدرال زیاد رخ می‌دهد، این رویکرد بسیار کارا خواهد بود [۲۰].

## ۲-۴ رویکردهای کلی و پایه‌ای در حل چالش‌ها

روش‌های بهینه‌سازی توزیع‌شده معمولاً برای حل مسائل بهینه‌سازی در سیستم‌هایی با شبکه‌های محاسباتی بزرگ و توزیع‌شده استفاده می‌شوند. این روش‌ها بر مبنای تقسیم مسئله بهینه‌سازی به زیرمسائل کوچک‌تر و حل آن‌ها در گره‌های مختلف شبکه استوارند. در این روش‌ها، اغلب فرض می‌شود که داده‌ها به‌صورت همگن و یکپارچه در سراسر شبکه توزیع شده‌اند و گره‌ها می‌توانند به راحتی با یکدیگر ارتباط برقرار کنند.

بر اساس آنچه در بخش‌های قبلی توضیح داده شد، این فرضیات در یادگیری فدرال به ندرت برقرار است، زیرا در یادگیری فدرال داده‌ها به‌صورت محلی و ناهمگن در دستگاه‌های مختلف قرار دارند و ارتباطات بین دستگاه‌ها ممکن است محدود و نامنظم باشد [۱۸]. بنابراین روش‌ها و رویکردهای لازم جهت حل این چالش‌ها متفاوت از مسائل بهینه‌سازی توزیع‌شده هستند. در ادامه این بخش، تلاش می‌شود دو رویکرد پایه‌ای برای مسائل یادگیری فدرال معرفی شود.

### ۲-۴-۱ الگوریتم میانگین‌گیری فدرال<sup>۳</sup> (FedAvg)

یکی از روش‌های اصلی و پرکاربرد در یادگیری فدرال روش میانگین‌گیری فدرال یا FedAvg است که توسط محققان گوگل در سال ۲۰۱۷ معرفی شد [۱۳]. این الگوریتم به منظور بهینه‌سازی مدل‌های یادگیری ماشین در یک محیط توزیع‌شده طراحی شده است. در این روش داده‌ها به‌صورت محلی در دستگاه‌های کاربران باقی می‌مانند و تنها به‌روزرسانی‌های مدل به اشتراک گذاشته می‌شوند. رویکرد اصلی FedAvg بر مبنای ترکیب به‌روزرسانی‌های محلی از دستگاه‌های مختلف به یک مدل سراسری استوار است.

<sup>1</sup>Differential Privacy

<sup>2</sup>Homomorphic Encryption

<sup>3</sup>Federated Averaging

یکی از مزایای اصلی FedAvg این است که به طور موثری با چالش ناهمگنی داده‌ها مقابله می‌کند. میانگین‌گیری وزنی در FedAvg به مدل کمک می‌کند تا به روزرسانی‌های مختلف را به گونه‌ای ترکیب کند که این ناهمگنی‌ها را در نظر بگیرد. به عبارت دیگر، اگر یک دستگاه داده‌های بیشتری داشته باشد، تأثیر بیشتری بر مدل نهایی خواهد داشت. این رویکرد باعث می‌شود که مدل فدرال به تعادل بهتری در یادگیری از داده‌های ناهمگن برسد و کارایی بالاتری داشته باشد. این ویژگی به ویژه در کاربردهایی مانند فروشگاه برنامه‌های کاربردی که کاربران متنوع و داده‌های متفاوتی دارند، بسیار سودمند است و می‌تواند به بهبود عملکرد مدل در شرایط واقعی کمک شایانی کند.

علاوه بر این، FedAvg به کاهش نیاز به ارتباطات مکرر بین دستگاه‌ها و سرور مرکزی کمک می‌کند. در بسیاری از روش‌های بهینه‌سازی توزیع‌شده، نیاز است که دستگاه‌ها به طور مکرر با سرور مرکزی ارتباط برقرار کنند تا به روزرسانی‌های خود را ارسال کنند. اما در FedAvg دستگاه‌ها می‌توانند چندین مرحله از بهینه‌سازی را به صورت محلی انجام دهند و سپس تنها به روزرسانی نهایی را ارسال کنند. این کاهش در نیاز به ارتباطات نه تنها باعث کاهش پهنای باند مورد نیاز می‌شود، بلکه به حفظ حریم شخصی کاربران نیز کمک می‌کند [۱۳].

برای درک چگونگی پیدایش الگوریتم FedAvg، باید به این نکته اشاره کرد که موفقیت‌های اخیر در یادگیری عمیق عمدتاً به استفاده از انواع الگوریتم نزول گرادیان تصادفی<sup>۱</sup> (SGD) برای بهینه‌سازی وابسته بوده‌اند. در حقیقت، بسیاری از دستاوردها، ناشی از تنظیم مدل و بهینه‌سازی تابع خطا با استفاده از روش‌های ساده گرادیان هستند. از این رو، الگوریتم‌های بهینه‌سازی فدرال نیز با الگوریتم SGD به عنوان نقطه شروع، طراحی و توسعه یافته‌اند [۱۳].

الگوریتم SGD به این صورت است که در هر دور ارتباط، گرادیان‌ها بر اساس داده‌های یک کاربر تصادفی انتخاب شده، محاسبه می‌شوند. این رویکرد از نظر محاسباتی کارآمد است، اما نیازمند تعداد بسیار زیادی از دوره‌های آموزش برای تولید مدل‌های مناسب است. برای مثال حتی با استفاده از رویکرد پیشرفته‌ای مانند نرمال‌سازی دسته‌ای<sup>۲</sup>، برای آموزش مجموعه داده‌ای تنها با ۶۰,۰۰۰ عضو و با دسته‌هایی به اندازه ۶۰ داده، به ۵۰,۰۰۰ دور آموزش جهت رسیدن به مدل مطلوب نیاز می‌باشد [۲۱].

در تنظیمات فدرال، مشارکت تعداد زیادی از کاربران هزینه چندانی در زمان واقعی ندارد زیرا همه کاربران می‌توانند به صورت همزمان به آموزش مدل محلی بپردازند. بنابراین، برای خط مبنا از SGD همزمان با دسته‌های بزرگ استفاده می‌شود. برای اعمال این رویکرد در تنظیمات فدرال، در هر دور، یک زیرمجموعه از کاربران با ضریب کنترلی  $C$  انتخاب می‌شوند و گرادیان خطا روی تمام داده‌های نگهداری شده توسط این کاربران محاسبه

<sup>۱</sup>Stochastic Gradient Descent

<sup>۲</sup>Batch Normalization

می‌گردد. بنابراین پارامتر  $C$  اندازه دسته کلی را کنترل می‌کند، به‌طوری که  $C = 1$  معادل با نزول گرادیان یک دسته کامل است. این الگوریتم خط مبنا FederatedSGD یا FedSGD نامیده می‌شود [۱۳].

یک پیاده‌سازی معمول از FedSGD با  $C = 1$  و نرخ یادگیری ثابت  $\eta$  به این صورت است که هر گره  $k$ ، گرادیان  $g_k = \nabla F_k(w_t)$  که میانگین گرادیان روی داده‌های محلی در مدل فعلی  $w_t$  است را محاسبه می‌کند و سرور مرکزی این گرادیان‌ها را جمع‌آوری کرده و به‌روزرسانی  $w_{t+1} \leftarrow w_t - \eta \sum_{k=1}^K \frac{n_k}{n} g_k$  را انجام می‌دهد، در حالی که  $\sum_{k=1}^K \frac{n_k}{n} g_k = \nabla f(w_t)$  خواهد بود. یک به‌روزرسانی معادل به این صورت است که برای هر گره عبارت  $\forall k, w_{t+1}^k \leftarrow w_t - \eta g_k$  محاسبه و سپس  $w_{t+1} \leftarrow \sum_{k=1}^K \frac{n_k}{n} w_{t+1}^k$  انجام شود.

در نتیجه، هر گره به‌صورت محلی یک گام گرادیان نزولی را روی مدل فعلی با استفاده از داده‌های محلی خود انجام داده و سپس سرور میانگین وزنی مدل‌های به‌دست‌آمده را محاسبه می‌کند. با نوشتن الگوریتم به این صورت، امکان تکرار به‌روزرسانی محلی  $w^k \leftarrow w^k - \eta \nabla F_k(w^k)$ ، چندین بار پیش از مرحله میانگین‌گیری فراهم شده و باعث افزایش محاسبات در هر گره خواهد شد. این مقدمه پیدایش الگوریتم FederatedAveraging یا FedAvg است [۱۳]. جهت درک بهتر این روش، می‌توان شکل ۱-۲ را مشاهده نمود و در گام سوم به میانگین وزنی مدل‌ها توجه کرد.

در این روش میزان محاسبات توسط سه پارامتر کلیدی کنترل می‌شود:

- $C$ : زیرمجموعه‌ای از تعداد گره‌هایی که در هر مرحله محاسبات انجام می‌دهند
- $E$ : تعداد مراحل آموزشی که هر گره در هر دور روی مجموعه داده محلی خود انجام می‌دهد
- $B$ : اندازه دسته محلی که برای به‌روزرسانی‌های هر گره استفاده می‌شود

در این جا  $B = \infty$  انتخاب می‌شود تا نشان دهد که کل مجموعه داده محلی به عنوان یک دسته واحد در نظر گرفته می‌شود. بنابراین، به عنوان یک نمونه از این الگوریتم گسترده شده جدید، انتخاب  $B = \infty$  و  $E = 1$  باعث می‌شود که این روش دقیقاً مانند FedSGD عمل کند. همچنین برای یک گره با  $n_k$  نمونه محلی، تعداد به‌روزرسانی‌های محلی در هر دور با  $u_k = E \frac{n_k}{B}$  نمایش داده می‌شود [۱۳]. شبه کد کامل این روش در الگوریتم ۱-۲ ارائه شده است. لازم به ذکر است که تمامی نمادهای مورد استفاده در این الگوریتم در جدول ۱-۲ توضیح داده شده‌اند.

---

```

1 initialize  $w_0$ ;
2 for each round  $t = 1, 2, \dots, T$  do
3    $m \leftarrow \max(C \cdot K, 1)$ ;
4    $U_t \leftarrow$  (random set of  $m$  clients);
5   for each client  $k \in U_t$  in parallel do
6      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$ ;
7   end
8    $w_{t+1} \leftarrow \sum_{k=1}^K \frac{n_k}{n} w_{t+1}^k$ ;
9 end
10 Function  $\text{ClientUpdate}(k, w)$ : // Run on client  $k$ 
11    $\mathcal{B} \leftarrow$  (split  $\mathcal{P}_k$  into batches of size  $B$ );
12   for each local epoch  $i$  from 1 to  $E$  do
13     for batch  $b \in \mathcal{B}$  do
14        $w \leftarrow w - \eta \nabla \ell(w \sim b)$ ;
15     end
16   end
17   return  $w$  to server;
18 end

```

---

جدول ۲-۱: نمادهای به کار رفته در الگوریتم FedAvg

متغیر	توضیحات
$w$	وزن‌های شبکه عصبی
$T$	تعداد گام‌ها
$K$	تعداد کاربران
$C$	ضریب کنترلی برای زیرمجموعه‌ای از کاربران
$m$	زیرمجموعه‌ای از کاربران
$U_t$	مجموعه‌ای نامرتب از $m$ در گام $t$
$n$	تعداد داده‌های آموزشی
$\mathcal{P}_k$	مجموعه داده‌های متعلق به کاربر $k$
$B$	اندازه دسته محلی
$E$	تعداد مراحل آموزش محلی
$\eta$	نرخ یادگیری

## ۲-۴-۲ بهینه‌سازی FedProx

روش FedProx با ایجاد تغییرات جزئی در رویکرد FedAvg، به بهبود پایداری و دقت در شبکه‌های ناهمگن کمک می‌کند. این تغییرات شامل اضافه کردن یک عبارت نزدیک‌سازی<sup>۱</sup> به تابع هدف است که به سرور کمک می‌کند تا ناهمگنی را مدیریت کند [۲۲]. عبارت نزدیک‌سازی، یک اصطلاح در زمینه بهینه‌سازی است که به تابعی افزوده می‌شود تا مشکلات ناشی از تفاوت‌های میان مدل‌های محلی و مدل سراسری در یادگیری فدرال را برطرف کند.

بر اساس روابط مندرج در [۲۲]، رابطه هدف FedProx به صورت زیر تعریف می‌شود:

$$\min_w f(w) = \min_w \sum_{k=1}^K \frac{n_k}{n} \left( F_k(w) + \frac{\mu}{2} \|w_t - w_t^k\|^2 \right) \quad (۵-۲)$$

---

<sup>۱</sup> Proximal Term

در رابطه (۵-۲) بخش  $\frac{\mu}{2} \|w_t - w_t^k\|^2$  همان عبارت نزدیک‌سازی است که به تابع هدف اضافه شده است. همچنین  $\mu$ ، یک پارامتر تنظیم برای این عبارت به حساب می‌آید و در نهایت  $w_t^k$  وزن‌های مدل محلی دستگاه  $k$  در تکرار  $t$  است.

حال با توجه به رابطه (۵-۲)، به‌روزرسانی وزن‌ها به شکل زیر تغییر پیدا خواهد کرد که در آن بخش  $\mu(w_t - w_t^k)$ ، گرادیان عبارت نزدیک‌سازی است.

$$w_{t+1} = w_t - \eta(\nabla F_k(w_t) + \mu(w_t - w_t^k)) \quad (۶-۲)$$

بنابراین، به‌روزرسانی‌های محلی در هر گام با به‌روزرسانی سراسری مرحله قبل مرتبط هستند. در حقیقت عبارت نزدیک‌سازی به عنوان یک مکانیزم منظم‌کننده<sup>۱</sup> عمل می‌کند که تفاوت‌های بین وزن‌های سراسری  $w$  و وزن‌های محلی  $w_t^k$  را کاهش می‌دهد. این عبارت به کاهش تأثیرات منفی ناهمگنی سیستم‌ها و داده‌ها کمک می‌کند و باعث پایداری بیشتر در فرآیند همگرایی می‌شود [۲۲].

## ۵-۲ جمع‌بندی

یادگیری فدرال به عنوان یک چارچوب نوین در یادگیری ماشین، از مفاهیم پایه یادگیری ماشین و یادگیری عمیق بهره می‌برد، اما از ساختاری توزیع‌شده استفاده می‌کند که در آن داده‌ها بین چندین دستگاه پخش می‌شود. در فرایند یادگیری فدرال، الگوریتم‌های بهینه‌سازی مانند گرادیان نزولی و انتشار به عقب به‌طور محلی اجرا می‌شوند و سپس نتایج این به‌روزرسانی‌ها با هم ترکیب می‌شوند تا مدل کلی بهبود یابد.

یادگیری فدرال با چالش‌هایی چون تبادل داده‌های پرهزینه، ناهمگنی‌های سیستمی و آماری و حفظ حریم شخصی مواجه است. به دلیل حجم بالای داده‌ها، تبادل اطلاعات بین سرور و دستگاه‌ها پرهزینه است و روش‌هایی مانند فشرده‌سازی داده‌ها برای کاهش این هزینه‌ها پیشنهاد شده است. همچنین، ناهمگنی در توان محاسباتی و آماری دستگاه‌ها می‌تواند تأثیرات منفی بر فرایند یادگیری داشته باشد که با روش‌هایی مانند تعادل در به‌روزرسانی مدل‌ها و یادگیری فعال می‌توان این مشکلات را به حداقل رساند. حفظ حریم شخصی نیز چالشی جدی است و برای مقابله با آن روش‌های امنیتی پیشرفته‌ای نظیر حریم خصوصی تفاضلی و رمزنگاری هم‌شکل به کار گرفته می‌شوند.

الگوریتم‌های FedAvg و FedProx به عنوان رویکردهای پایه‌ای در یادگیری فدرال، چالش‌های مرتبط با ناهمگنی داده‌ها و تفاوت‌های سیستمی را هدف قرار می‌دهند. FedAvg با جمع‌آوری و میانگین‌گیری وزن‌های مدل‌های محلی که توسط کاربران آموزش دیده‌اند، به یک مدل سراسری همگرا می‌شود، اما در برابر تنوع داده‌ها

<sup>۱</sup> Regularization

و دستگاه‌ها حساس است. FedProx با اضافه کردن یک عبارت جریمه به تابع هدف، سعی می‌کند ناهم‌آهنگی بین مدل‌های محلی و مدل مرکزی را کاهش دهد و در نتیجه بهبود پایداری و همگرایی در شرایط مختلف را فراهم کند. این دو الگوریتم پایه، راهکارهایی مهم برای مدیریت چالش‌های یادگیری فدرال هستند.

## فصل سوم

### بررسی اختصاصی پیشینه روش‌های حل مشکل ناهمگنی آماری

#### ۳-۱ مقدمه

همان‌گونه که در ۲-۳-۳ بیان شد، یکی از چالش‌های اصلی در یادگیری فدرال، مسئله داده‌های غیرمستقل و غیریکنواخت (non-IID) است که منجر به بروز مشکلات آماری و تفاوت‌های داده‌ای می‌شود. این موضوع می‌تواند باعث کاهش کارایی مدل‌های یادگیری در استفاده از داده‌های توزیع‌شده شود. به دلیل اهمیت این مسئله، بسیاری از پژوهشگران تلاش‌های زیادی برای رفع این مشکل انجام داده‌اند. از آن‌جا که مبحث اصلی این پژوهش نیز به‌طور دقیق به همین مسئله اشاره دارد، یک فصل مجزا به این موضوع اختصاص داده شده است. در ادامه این فصل، به‌صورت خلاصه به بررسی راه‌حل‌هایی که تاکنون برای حل این مشکل مطرح شده‌اند، پرداخته خواهد شد. همچنین باید توجه داشت که هر یک از این راه‌حل‌ها نقاط قوت و ضعف خاص خود را دارند و بسته به شرایط و نوع داده‌ها، می‌توانند نتایج متفاوتی را به همراه داشته باشند. بررسی دقیق این راه‌حل‌ها و ارزیابی کارایی آن‌ها می‌تواند به بهبود سیستم‌های یادگیری فدرال و غلبه بر مشکلات مرتبط با داده‌های غیرمستقل و غیریکنواخت کمک شایانی کند.



### ۲-۳ نگرش برپایه داده

جهت درک و حل چالش‌های ناشی از داده‌های non-IID در یادگیری فدرال، استفاده از نگرشی بر پایه داده می‌تواند بسیار مؤثر باشد. این نگرش با تاکید بر اشتراک‌گذاری، انتخاب و بهبود داده‌ها<sup>۱</sup>، به کاهش تفاوت‌های موجود بین داده‌های محلی کمک کرده و کارایی فرآیند آموزش را افزایش می‌دهد. با به کارگیری تکنیک‌های پیشرفته و مدیریت هوشمندانه داده‌ها، می‌توان به دقت و هماهنگی بهتری در مدل‌های نهایی دست یافت و همزمان حریم شخصی کاربران را حفظ کرد.

#### ۱-۲-۳ اشتراک‌گذاری داده

مشکل اصلی الگوریتم FedAvg در مواجهه با داده‌های غیرمستقل و غیریکنواخت، تفاوت وزن‌های اولیه در شروع فرآیند آموزش است. این تفاوت‌ها می‌توانند باعث شوند که مدل‌های محلی در هر گره به‌طور قابل توجهی متفاوت از یکدیگر باشند و در نتیجه منجر به مشکلات همگرایی و کاهش کارایی مدل نهایی شود.

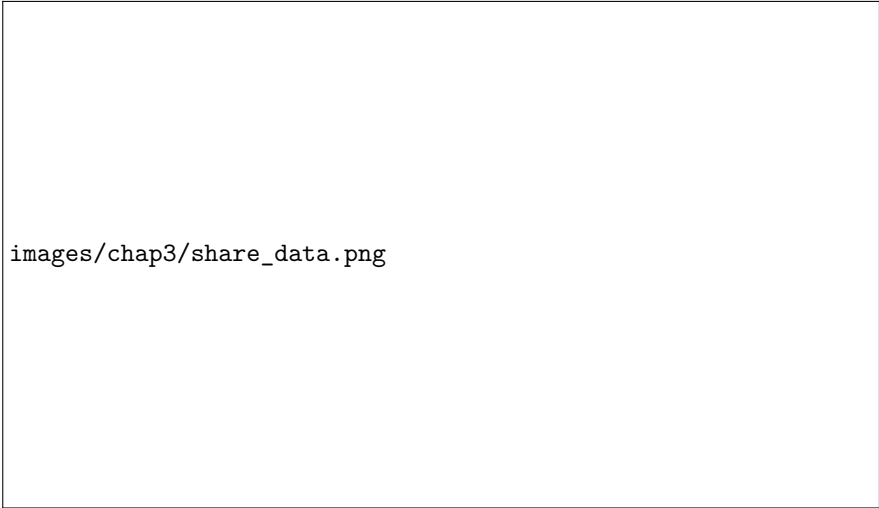
برای رفع این مشکل، در [۲۳] روشی بر پایه اشتراک‌گذاری داده پیشنهاد شده است که در آن ابتدا سرور مرکزی مقدار کمی از داده‌ها را به‌صورت محلی آموزش می‌دهد. در این مرحله، سرور مرکزی با استفاده از این داده‌ها، یک مدل اولیه را آموزش داده و وزن‌های اولیه آن را تنظیم می‌کند. سپس، این وزن‌های اولیه به همراه داده‌های آموزش دیده شده به تمامی کاربران ارسال می‌شود. این اقدام باعث می‌شود که تمام کاربران در ابتدای فرآیند آموزش با مجموعه‌ای از داده‌های مشترک و وزن‌های اولیه مشابه روبه‌رو شوند.

نقطه قوت این روش در این است که به دلیل انجام این عملیات تنها در آغاز فرآیند آموزش، هزینه زیادی به شبکه تحمیل نمی‌شود. در واقع، انتقال داده‌ها و وزن‌ها فقط در ابتدا انجام شده و پس از آن کاربران به‌صورت مستقل به آموزش مدل‌های محلی خود ادامه می‌دهند. این اقدام منجر به کاهش اختلافات ناشی از ناهمگنی داده‌ها شده و فرآیند همگرایی مدل نهایی سریع‌تر و با دقت بیشتری انجام می‌شود.

در شکل ۱-۳، نحوه اجرای این روش و مراحل مختلف آن به تصویر کشیده شده است. این تصویر نشان می‌دهد که چگونه سرور مرکزی ابتدا داده‌های کمی را آموزش می‌دهد، وزن‌های اولیه را تنظیم می‌کند و سپس این وزن‌ها و داده‌ها را به کاربران ارسال می‌کند تا فرآیند آموزش محلی با یک نقطه شروع مشترک برای همه کاربران آغاز شود.

یکی دیگر از روش‌های مطرح شده در زمینه یادگیری فدرال به این صورت است که کاربران بتوانند نتایج آموزش تعدادی داده اشتراکی را با یکدیگر به اشتراک بگذارند و از نتایج دیگر کاربران بر روی این داده‌های

<sup>۱</sup>Data Enhancement



images/chap3/share\_data.png

شکل ۳-۱: نمایش نحوه به اشتراک گذاری داده در یادگیری فدرال [۲۳].

اشتراکی مطلع شوند. در این روش، کاربران نتایج به دست آمده از آموزش داده‌های مشترک را با هم مبادله می‌کنند که این کار منجر به بهبود عملکرد مدل‌های محلی و در نهایت مدل سراسری می‌شود [۲۴]. براساس بررسی‌های انجام شده، برای مثال در مجموعه داده‌ای تنها با ۶۰,۰۰۰ داده ورودی، اگر حدود ۵ درصد از داده‌ها به صورت اشتراکی در اختیار کاربران قرار گیرد، دقت مدل تا حدود ۳۰ درصد افزایش خواهد یافت. این افزایش دقت به دلیل همگرایی بهتر مدل‌ها و کاهش تفاوت‌های آماری بین داده‌های محلی است. به عبارتی دیگر، این روش کمک می‌کند که مدل‌ها با یکدیگر هماهنگ‌تر شوند و نتایج دقیق‌تری ارائه دهند.

با این حال، باید توجه داشت که اشتراک‌گذاری داده‌ها بین کاربران می‌تواند مسائل حریم شخصی را به همراه داشته باشد. به عبارت دیگر، هنگامی که داده‌های اشتراکی بین کاربران مبادله می‌شود، احتمال نقض حریم شخصی کاربران افزایش می‌یابد. بنابراین، هنگام پیاده‌سازی این روش، ضروری است که اقدامات لازم برای حفظ حریم شخصی کاربران به طور جدی مد نظر قرار گیرد. این اقدامات می‌تواند شامل استفاده از تکنیک‌های رمزنگاری، ناشناس‌سازی داده‌ها، یا روش‌های دیگر برای محافظت از اطلاعات حساس کاربران باشد [۳].

در نهایت، روش به اشتراک‌گذاری داده‌ها بین کاربران، اگرچه می‌تواند به بهبود دقت و کارایی مدل‌ها کمک کند، اما نیازمند دقت و توجه ویژه‌ای به مسائل حریم شخصی است. پژوهشگران و توسعه‌دهندگان باید با در نظر گرفتن این چالش‌ها، راهکارهایی را برای حفظ امنیت و حریم شخصی کاربران در هنگام اجرای این روش‌ها ارائه دهند.

### ۳-۲-۲ بهبود داده

ابتدا، کاربران تعدادی از داده‌های خود را به سمت سرور ارسال می‌کنند. سرور، با استفاده از داده‌های دریافتی، یک مدل شبکه مولد رقابتی<sup>۱</sup> ایجاد می‌کند و این مدل را برای تمامی کاربران ارسال می‌نماید. کاربران با استفاده از این شبکه مولد رقابتی و با توجه به داده‌های خود، تعدادی داده جدید تولید کرده و در مراحل بعدی آموزش از این داده‌ها نیز استفاده می‌کنند. به این ترتیب، شبکه مولد رقابتی به کاربران کمک می‌کند تا داده‌های بیشتری برای آموزش مدل‌های خود در اختیار داشته باشند و از این داده‌ها برای بهبود عملکرد مدل‌های خود استفاده کنند. در شکل ۳-۲ نحوه عملکرد این روش به تصویر کشیده شده است.

این روش، به دلیل استفاده از داده‌های تولید شده، نسبت به روش‌های اشتراک‌گذاری داده‌ها از نظر حفظ حریم شخصی کاربران بهتر عمل می‌کند. به این معنی که، به جای ارسال داده‌های خام کاربران به سرور یا دیگر کاربران، از داده‌های تولید شده توسط شبکه مولد رقابتی استفاده می‌شود که احتمال نقض حریم شخصی را کاهش می‌دهد. به عبارت دیگر، حتی اگر داده‌ها در طول انتقال یا در سرور مورد دسترسی غیرمجاز قرار گیرند، به دلیل رمزگذاری، اطلاعات واقعی کاربران فاش نخواهد شد. این ویژگی، امنیت و حریم شخصی کاربران را به‌طور قابل توجهی افزایش می‌دهد و از اطلاعات حساس آنان در برابر تهدیدات محافظت می‌کند [۲۵].

بنابراین، روش‌های بهبود داده که مبتنی بر رمزگذاری و رمزگشایی داده‌ها هستند، در حالی که تلاش می‌کنند عملکرد مدل‌های یادگیری را بهبود بخشند، از حریم شخصی کاربران نیز حفاظت می‌نمایند. این ترکیب از امنیت



شکل ۳-۲: استفاده از شبکه مولد رقابتی جهت تولید داده [۲۵].

<sup>۱</sup> Generative Adversarial Network (GAN)

و کارایی، این روش‌ها را به گزینه‌های مناسبی برای استفاده در سیستم‌های یادگیری فدرال تبدیل کرده است.

### ۳-۲-۳ انتخاب داده

در هنگام انتخاب کاربران برای فرآیند آموزش، می‌توان از الگوریتم‌هایی که بر پایه کیفیت داده‌ها عمل می‌کنند، استفاده نمود. به عنوان نمونه، می‌توان از الگوریتم حریصانه کوله‌پشتی برای اولویت‌بندی کاربران بهره برد، به نحوی که کاربران با داده‌های غنی و گسترده‌تر، اولویت بالاتری جهت انتخاب داشته باشند. این رویکرد به بهبود کیفیت آموزش کمک می‌کند، زیرا داده‌های با کیفیت بالاتر تاثیر مثبتی بر نتایج نهایی مدل خواهند داشت [۲۶].

علاوه بر این، می‌توان از روش‌های یادگیری عمیق برای تخمین زمان اجرای مدل در سمت کاربران استفاده کرد. این روش‌ها می‌توانند زمان مورد نیاز برای اجرای مدل را پیش‌بینی کنند و بر اساس این پیش‌بینی، از بین ویژگی‌های مختلف جهت آموزش، تنها آن‌هایی را انتخاب نمایند که تاثیر بیشتری بر خروجی خواهند داشت. به این ترتیب، با بهینه‌سازی انتخاب ویژگی‌ها، می‌توان زمان و منابع محاسباتی را به شکل موثرتری مدیریت کرد. یکی از نکات کلیدی در استفاده از این روش‌های انتخاب داده این است که هیچ کدام از آن‌ها تغییری بر روی داده‌ها و کاربران ایجاد نمی‌کنند. به عبارت دیگر، این روش‌ها به گونه‌ای طراحی شده‌اند که داده‌های موجود و وضعیت کاربران بدون تغییر باقی می‌مانند، اما فرآیند انتخاب و استفاده از داده‌ها بهینه‌تر و کارآمدتر می‌شود. این ویژگی، استفاده از این راه‌حل‌ها را در برنامه‌های مختلف بسیار کاربردی و موثر می‌سازد [۲۷].

در نتیجه، استفاده از الگوریتم‌های مبتنی بر کیفیت داده‌ها و روش‌های یادگیری عمیق برای تخمین زمان اجرا، می‌تواند به‌طور قابل توجهی فرآیند آموزش در سیستم‌های یادگیری فدرال را بهبود بخشد. این روش‌ها نه تنها کیفیت داده‌های مورد استفاده را افزایش می‌دهند، بلکه با بهینه‌سازی منابع محاسباتی و زمان اجرا، کارایی سیستم را نیز بهبود می‌بخشند. این ترکیب از بهینه‌سازی داده‌ها و مدیریت منابع، به ویژه در محیط‌های با منابع محدود، اهمیت ویژه‌ای دارد و می‌تواند به نتایج بهتری در آموزش مدل‌ها منجر شود.

### ۳-۳ نگرش برپایه مدل

در نگرش بر پایه مدل، تمرکز بر بهبود و بهینه‌سازی مدل‌های یادگیری فدرال است تا به عملکرد و دقت بالاتری دست یابد. این رویکرد با استفاده از تکنیک‌هایی مانند تجمیع و به‌روزرسانی مدل‌ها<sup>۱</sup>، بهینه‌سازی تطبیقی<sup>۲</sup> و بهینه‌سازی منظم، سعی در کاهش نوسانات و افزایش همگرایی مدل‌ها دارد. با بهره‌گیری از این روش‌ها، مدل

<sup>۱</sup>Models Update and Aggregation

<sup>۲</sup>Adaptive Optimization

نهایی می‌تواند به‌طور موثرتر به تفاوت‌ها و تنوع داده‌ها پاسخ دهد و نتایج دقیق‌تر و قابل اعتمادتری ارائه دهد.

### ۱-۳-۳ تجمیع و به‌روزرسانی مدل

هنگام اجرای الگوریتم در مراحل میانی، می‌توان با استفاده از ساختار شبکه‌های عصبی عمیق موجود، تفاوت گره‌های شبکه بین کاربران مختلف را بررسی نمود. این بررسی امکان بهبود ساختار مدل اصلی را بر اساس تفاوت‌ها و ویژگی‌های مختلف کاربران فراهم می‌کند و در نتیجه به ایجاد مدلی کارآمدتر منجر می‌شود. این فرآیند می‌تواند به بهینه‌سازی عملکرد مدل و افزایش دقت آن در مراحل بعدی کمک کند [۲۸].

روش دیگری برای بهبود عملکرد یادگیری فدرال این است که هم در سمت سرور و هم در سمت کاربران چندین مدل شبکه عصبی قرار داده شود. این شبکه‌ها به‌صورت جداگانه آموزش داده شده و به‌روزرسانی می‌شوند. پس از چند مرحله آموزش، می‌توان شبکه‌ها را با یکدیگر ترکیب کرد. این رویکرد به بهبود عملکرد کلی مدل کمک می‌کند و باعث می‌شود تا مدل نهایی از ویژگی‌ها و مزایای چندین شبکه عصبی بهره‌مند شود [۲۹]. در شکل ۳-۳ نحوه عملکرد این روش به تصویر کشیده شده است.

همچنین، مکانیزم یادگیری فدرال نیمه-ناهمزمان<sup>۱</sup> نیز یکی دیگر از روش‌های موثر در این حوزه است [۳۰]. در این روش، مدل‌های کاربران به ترتیبی که به سرور می‌رسند به‌روزرسانی می‌شوند. این رویکرد به خوبی با کاربران کند<sup>۲</sup> که ممکن است در گردش‌های مختلف به سرور پیوندند، سازگار است. با به‌روزرسانی و ترکیب مدل‌ها در مراحل مختلف، این مکانیزم به خوبی می‌تواند توازن را برای داده‌های ناهمگن برقرار کند و عملکرد مدل را بهینه سازد.

در نهایت، با استفاده از این رویکردها و الگوریتم‌ها می‌توان به‌طور موثرتری با چالش‌های موجود در یادگیری

images/chap3/multi\_local\_and\_multi\_global.png

شکل ۳-۳: چارچوب یک سیستم یادگیری فدرال چندمحلی و چندمرکزی برای کشف ناهنجاری‌ها [۲۹].

<sup>۱</sup>Semi-Asynchronous

<sup>۲</sup>Stragglers

فدرال مقابله کرد و مدل‌هایی با دقت و کارایی بالاتر ایجاد نمود. این روش‌ها نه تنها به بهبود ساختار مدل‌ها کمک می‌کنند، بلکه باعث می‌شوند تا فرآیند آموزش بهینه‌تر و سازگارتر با تنوع و ناهمگنی داده‌ها انجام شود.

### ۳-۳-۲ بهینه‌سازی تطبیقی

در این روش، الگوریتم پیش‌بینی میزان کار به گونه‌ای طراحی شده است که به‌صورت خودکار اطلاعات جامعی از سابقه آموزش هر کاربر را جمع‌آوری می‌کند. این اطلاعات شامل عملکرد کاربر در مراحل قبلی آموزش است. سپس بر اساس این سوابق، میزان پیچیدگی الگوریتم برای مرحله بعدی آموزش تعیین می‌شود تا برای کاربر مربوطه مناسب باشد. این رویکرد به بهینه‌سازی فرآیند آموزش کمک می‌کند و موجب می‌شود تا الگوریتم‌ها به شکل موثرتری با توانایی‌های هر کاربر هماهنگ شوند [۳۱].

یکی از روش‌های اولیه در بهینه‌سازی تطبیقی، استفاده از روش کاهش نرخ یادگیری است. در این روش، نرخ یادگیری برای هر کاربر به‌طور جداگانه و بر اساس عملکرد گذشته وی تعیین می‌شود. بر اساس این روش، ممکن است برای کاربرانی که عملکرد بهتری دارند، نرخ یادگیری افزایش یابد. در حالی که برای کاربرانی که با چالش‌هایی روبه‌رو هستند، نرخ یادگیری کاهش پیدا می‌کند تا به این وسیله فرآیند یادگیری بهینه‌تر شود [۳۲]. در طول سال‌های اخیر، بهینه‌سازی تطبیقی نشان داده است که می‌تواند تاثیر قابل‌توجهی بر بهبود عملکرد الگوریتم‌ها داشته باشد. به همین دلیل، محققان به سمت توسعه روش‌هایی رفته‌اند که امکان تغییر و تطبیق پارامترهای الگوریتم را در طول زمان فراهم کنند [۳۲]. این رویکرد باعث می‌شود تا هر کاربر بتواند در مراحل مختلف آموزش، پارامترهای مربوط به الگوریتم را متناسب با نیازها و شرایط خود تنظیم کند. این انعطاف‌پذیری به الگوریتم‌ها کمک می‌کند تا با گذشت زمان، کارایی بیشتری داشته باشند و به‌طور خاص‌تر با شرایط و نیازهای کاربران سازگار شوند.

به‌طور کلی، استفاده از الگوریتم‌های پیش‌بینی و بهینه‌سازی تطبیقی می‌تواند به شکل چشمگیری کیفیت آموزش و کارایی سیستم‌های یادگیری را بهبود بخشد. این روش‌ها با فراهم کردن امکان تنظیم پارامترهای آموزشی بر اساس سوابق و عملکرد کاربران، موجب می‌شوند تا فرآیند یادگیری به شکل دقیق‌تر و موثرتری انجام شود. در نتیجه، کاربران می‌توانند از تجربیات گذشته خود بهره ببرند و با شرایط بهتر و مناسبتری به یادگیری ادامه دهند.

### ۳-۳-۳ بهینه‌سازی منظم

از مهم‌ترین و پرکاربردترین روش‌های موجود جهت کنترل داده‌های غیرمستقل و غیریکنواخت، رویکردهای بهینه‌سازی منظم هستند. این رویکردها با هدف بهبود فرآیند یادگیری و کاهش نوسانات ناشی از تفاوت در

توزیع داده‌ها به کار گرفته می‌شوند. به عنوان مثال، یکی از روش‌های متداول در این زمینه، در نظر گرفتن نزدیک‌ترین همسایه است که طی آن تابع بهینه‌سازی محلی برای هر کاربر به‌روزرسانی می‌شود تا از نوسانات زیاد جلوگیری کند و هماهنگی بیشتری بین داده‌های مختلف کاربران ایجاد شود [۲۲].

یکی دیگر از روش‌های معروف در این زمینه، مکانیزم معلم-شاگرد<sup>۱</sup> است. در این روش، یک مدل به عنوان معلم و مدل‌های دیگر به عنوان شاگرد عمل می‌کنند. گرادینان‌ها برای هر کاربر توسط یک جمله اضافه شده به نام جمله منظم‌سازی<sup>۲</sup> تنظیم می‌شود. این جمله منظم‌سازی به منظور کاهش خطاها و بهبود دقت مدل‌ها افزوده می‌شود و از بیش‌برازش<sup>۳</sup> جلوگیری می‌کند [۳۳].

رویکردهای بهینه‌سازی منظم، همان‌طور که در حوزه‌های مختلف یادگیری ماشین و یادگیری عمیق توانسته‌اند کارایی خود را به اثبات برسانند، در یادگیری فدرال نیز عملکرد بسیار خوبی دارند. این رویکردها با تنظیم مدل‌ها به گونه‌ای که نوسانات ناشی از داده‌های مختلف را کاهش دهند، به بهبود عملکرد کلی سیستم کمک می‌کنند. همچنین، با جلوگیری از بیش‌برازش، مدل‌ها را به سمت تعمیم بهتر هدایت می‌کنند، که این امر در محیط‌هایی با داده‌های غیرمستقل و غیریکنواخت بسیار حیاتی است.

در مجموع، استفاده از روش‌های بهینه‌سازی منظم در یادگیری فدرال نه تنها باعث بهبود دقت مدل‌ها می‌شود، بلکه موجب می‌گردد تا فرآیند یادگیری با پایداری و کارایی بیشتری انجام شود. این رویکردها به دلیل توانایی‌شان در کنترل نوسانات و کاهش خطاها، از ابزارهای اساسی در یادگیری فدرال به شمار می‌آیند و به توسعه مدل‌های دقیق و قابل اعتماد کمک می‌کنند.

### ۴-۳ نگرش برپایه چهارچوب

در روش‌های یادگیری فدرال، نگرش برپایه چهارچوب نقش کلیدی در بهبود عملکرد مدل‌ها ایفا می‌کند. این نگرش با استفاده از تکنیک‌های مختلف مانند خوشه‌بندی مشابهت<sup>۴</sup>، تقطیر دانش<sup>۵</sup> و لایه‌های شخصی‌سازی<sup>۶</sup> تلاش دارد تا فرآیند یادگیری را بهینه‌تر و هماهنگ‌تر کند. با توجه به اهمیت بهینه‌سازی و کاهش هزینه‌های ارتباطی، این رویکردها می‌توانند تأثیر مثبتی بر کارایی سیستم‌های یادگیری فدرال در داده‌های non-IID داشته باشند و به بهبود همگرایی مدل‌ها کمک کنند.

<sup>1</sup>Teacher-Student

<sup>2</sup>Regularization Term

<sup>3</sup>Overfitting

<sup>4</sup>Similarity Clustering

<sup>5</sup>Knowledge Distillation

<sup>6</sup>Personalization Layers

### ۳-۴-۱ خوشه‌بندی مشابهت

خوشه‌بندی یکی از روش‌های بسیار پرکاربرد و مهم در حوزه یادگیری ماشین است که ایده‌های آن می‌توانند در یادگیری فدرال نیز مورد استفاده قرار گیرند. در این روش، هنگامی که کاربران مدل‌های خود را آموزش داده و به سرور ارسال می‌کنند، سرور بر اساس مدل‌های دریافتی شباهت‌های آن‌ها را بررسی کرده و کاربرانی که مدل‌های مشابه دارند را در یک خوشه قرار می‌دهد. این فرایند به سرور امکان می‌دهد تا در مراحل بعدی، مدل یکسانی را برای اعضای هر خوشه ارسال کند. این رویکرد باعث می‌شود که مدل‌های آموزش دیده شده توسط کاربران با داده‌های مشابه، به‌طور همزمان و هماهنگ بهبود یابند و از همگرایی بهتری برخوردار شوند [۳۴].

به‌طور معمول، پس از چندین دوره آموزشی، فرآیند خوشه‌بندی مجدداً تکرار می‌شود تا از به‌روزرسانی‌های جدید و تغییرات احتمالی در داده‌ها و مدل‌ها بهره‌برداری شود. در شکل ۳-۴، حالت کلی خوشه‌بندی شباهت در سیستم‌های فدرال به تصویر کشیده شده است.

با وجود تمام مزایایی که روش خوشه‌بندی مشابهت به همراه دارد، یکی از مهم‌ترین مشکلات آن هزینه بالای ارتباطات است. در این روش، نیاز است که ساختار خوشه‌بندی در مراحل مختلف ارسال و دریافت شود، که این فرایند می‌تواند هزینه زیادی را بر شبکه اعمال کند. به خصوص در محیط‌هایی با تعداد زیاد کاربران و داده‌های بزرگ، این هزینه‌ها به‌طور قابل توجهی افزایش می‌یابد و می‌تواند عملکرد کلی سیستم را تحت تأثیر قرار دهد. بنابراین، در حالی که خوشه‌بندی شباهت می‌تواند کارایی و دقت یادگیری فدرال را بهبود بخشد، باید به دقت هزینه‌های ارتباطی آن نیز مورد ارزیابی قرار گیرد و در صورت امکان، بهینه‌سازی‌های لازم انجام شود تا این هزینه‌ها کاهش یابند. به کارگیری روش‌های بهینه‌سازی ارتباطات و فشرده‌سازی داده‌ها می‌تواند در این زمینه مفید باشند و به حفظ تعادل بین کارایی و هزینه‌ها کمک کنند.

images/chap3/similarity\_clustering.png

شکل ۳-۴: روش خوشه‌بندی مشابهت [۳۴].



### ۳-۴-۲ تقطیر دانش

به طور کلی، هدف اصلی روش های تقطیر دانش، ساده سازی مدل های پیچیده و ارائه مدل هایی ساده اما کارآمد است. بنابراین یکی از مهم ترین مزایای استفاده از تقطیر دانش، کاهش چشمگیر سربار شبکه است. در واقع به دلیل این که در این روش به جای ارسال پارامترهای مدل های محلی، فقط خروجی نهایی مدل ها ارسال می شود، حجم داده های اسالی به طور قابل توجهی کاهش می یابد. این کاهش حجم داده ها نه تنها هزینه های ارتباطی را پایین می آورد بلکه سرعت پردازش و به روزرسانی مدل ها را نیز افزایش می دهد. به این ترتیب، بهره وری سیستم بهبود یافته و توان محاسباتی به نحو بهتری مدیریت می شود.

یکی از الگوریتم های مهم در این زمینه روش DS-FL<sup>۱</sup> است [۳۵]. در این روش، ابتدا هر کاربر محلی با استفاده از داده های خود، مدلی را آموزش می دهد. سپس به جای ارسال پارامترهای مدل به سرور مرکزی، تنها خروجی مدل روی داده های بدون برچسب به اشتراک گذاشته می شود. سرور مرکزی با تجمیع این خروجی ها، یک مدل سراسری به روز شده را ایجاد می کند و آن را برای کاربران ارسال می کند. این فرایند تکرار می شود تا مدل سراسری به بهینه ترین حالت ممکن برسد.

به کارگیری تقطیر دانش در یادگیری فدرال نه تنها به بهبود کارایی شبکه کمک می کند، بلکه امنیت و حریم خصوصی داده ها را نیز افزایش می دهد. به عبارت دقیق تر چون خروجی مدل ها اغلب اطلاعات حساس کمتری نسبت به پارامترهای مدل در خود دارند، احتمال افشای اطلاعات شخصی کاربران کاهش می یابد. این ویژگی به خصوص در محیط هایی که حفظ حریم شخصی کاربران اولویت بالایی دارد، از اهمیت ویژه ای برخوردار است.

به طور خلاصه، روش های تقطیر دانش مانند DS-FL با هدف ساده سازی مدل های پیچیده و کاهش هزینه های ارتباطی، به بهبود کارایی و امنیت در سیستم های یادگیری فدرال کمک می کنند. این روش ها با ارسال خروجی های مدل به جای پارامترها، سربار شبکه را کاهش داده و به تطبیق بهتر مدل ها با داده های غیرمستقل و غیریکنواخت کمک می کنند.

### ۳-۴-۳ لایه های شخصی سازی

روش لایه های شخصی سازی شده به این شکل عمل می کند که در ابتدا کاربران بر اساس معیارهایی مانند کارایی آموزش و سرعت اجرا به گروه های مختلفی تقسیم می شوند. سپس، این کاربران بر اساس معیارهای تعیین شده به صورت لایه ای مرتب می شوند. به این ترتیب، سرور هنگامی که مدل را به روزرسانی می کند و قصد دارد آن را در مرحله بعد به سمت کاربران ارسال نماید، سعی می کند کاربرانی را که در یک لایه مشترک حضور دارند

<sup>۱</sup> Distillation-base Semi-supervised Federated Learning

انتخاب کند. این انتخاب به سرور امکان می‌دهد تا گردش به‌روزرسانی‌ها را با سرعت و کارایی هماهنگ‌تری به پایان برساند و عملکرد بهتری از سیستم بگیرد [۳۶].

یکی از نکات کلیدی در اجرای این روش، تعیین میزان آستانه‌ای است که بر اساس آن، کاربران به لایه‌های مختلف تقسیم می‌شوند. این تقسیم‌بندی باید به گونه‌ای باشد که خروجی مدل بهینه باشد و در عین حال کارایی سیستم حفظ شود. تعیین این آستانه‌ها می‌تواند چالش برانگیز باشد و نیاز به سعی و خطا دارد تا بهترین ترکیب ممکن به دست آید.

به‌طور کلی، روش لایه‌های شخصی‌سازی شده با تقسیم‌بندی کاربران و مرتب‌سازی آن‌ها در لایه‌های مختلف، امکان بهبود هماهنگی و کارایی در گردش به‌روزرسانی‌ها را فراهم می‌کند. این رویکرد نه تنها باعث می‌شود که کاربران با سرعت مشابه در یک لایه قرار گیرند، بلکه به سرور کمک می‌کند تا با کاهش ناهماهنگی‌ها، به‌روزرسانی مدل‌ها را با کارایی بیشتری انجام دهد. انتخاب صحیح معیارهای تقسیم‌بندی و آستانه‌ها در این روش، از اهمیت بالایی برخوردار است و نیازمند تحلیل و ارزیابی دقیق است تا بهترین نتایج ممکن به دست آید.

### ۳-۵ نگرش برپایه الگوریتم

نگرش بر پایه الگوریتم در یادگیری فدرال به مجموعه‌ای از رویکردها اشاره دارد که هدف آن‌ها بهبود فرآیند آموزش مدل‌های توزیع شده با استفاده از الگوریتم‌های پیشرفته است. این روش‌ها با ترکیب الگوریتم‌هایی مانند فرایادگیری، یادگیری چندوظیفه‌ای و یادگیری مادام‌العمر<sup>۱</sup> تلاش می‌کنند تا مدل‌ها را با سرعت بیشتری بهینه‌سازی کنند و دقت بالاتری در مواجهه با داده‌های non-IID داشته باشند. به کارگیری این الگوریتم‌ها کمک می‌کند تا سیستم‌های یادگیری فدرال بهبود یابند و چالش‌های موجود در هماهنگی و همگرایی مدل‌ها بهتر مدیریت شود.

#### ۳-۵-۱ فرایادگیری

روش‌های فرایادگیری به دلیل توانایی‌شان در هماهنگی سریع با داده‌های جدید و تغییر پارامترهای مربوطه، مورد توجه قرار گرفته‌اند. این روش‌ها می‌توانند به سرعت با شرایط جدید سازگار شوند و پارامترهای مدل را بهبود بخشند. مدل ابتدایی فرایادگیری پیاده‌شده بر بستر یادگیری فدرال، در واقع از همان الگوریتم FedAvg بهره می‌برد و با ترکیب آن با روش فرایادگیری، تلاش دارد تا فرایند آموزش را بهینه‌سازی کرده و پارامترهای مناسب‌تری را به دست آورد [۳۷].

<sup>۱</sup>Life-Long Learning

الگوریتم اولیه-دوگانه<sup>۱</sup> (FedPD)، یکی از الگوریتم‌های کارا با استفاده از فرایادگیری است که حتی برای توابع غیرمحدب نیز مقاوم بوده و علاوه بر دستیابی به همگرایی مناسب، از نظر کاهش ارتباطات نیز بسیار کارآمد عمل می‌کند [۳۸]. با این حال، یکی از چالش‌های اصلی این روش‌ها مربوط به کاربران کند است. این کاربران ممکن است به دلیل محدودیت‌های سخت‌افزاری یا مشکلات ارتباطی، نتوانند به‌روزرسانی‌های سریع و هماهنگ را انجام دهند و این موضوع می‌تواند باعث اختلال در عملکرد مدل شود.

به‌طور کلی، مدل‌های فرایادگیری در بستر یادگیری فدرال با ترکیب روش‌های مختلف و بهره‌گیری از الگوریتم‌های بهینه‌سازی مانند FedAvg و FedPD، سعی دارند تا با بهبود فرآیندهای آموزش و کاهش هزینه‌های ارتباطی، به نتایج بهتری دست یابند. این روش‌ها با وجود چالش‌هایی که ممکن است در پیاده‌سازی و هماهنگی با کاربران کند داشته باشند، به دلیل قابلیت‌هایشان در بهینه‌سازی و هماهنگی سریع با داده‌های جدید، پتانسیل بالایی برای بهبود عملکرد سیستم‌های یادگیری فدرال دارند.

### ۳-۵-۲ یادگیری چندوظیفه‌ای

یادگیری چندوظیفه‌ای به این معناست که هر یک از کاربران شرکت‌کننده در فرآیند یادگیری فدرال، به دنبال یادگیری وظایف مختلفی هستند و تلاش می‌شود که در این مسیر، حریم شخصی کاربران به‌طور قابل‌توجهی حفظ شود. در یادگیری فدرال چندوظیفه‌ای، کاربران بر اساس داده‌های محلی خود، مدل را آموزش می‌دهند و نتایج آن را به سمت سرور مرکزی ارسال می‌کنند. سپس سرور، با تحلیل پارامترهای ارسال شده، روابط معناداری میان این مدل‌ها پیدا کرده و مدل به‌روز شده را دوباره به سمت کاربران بازمی‌گرداند [۳۹].

به عبارت دیگر، در این روش، هر کاربر ابتدا مدل را با استفاده از داده‌های محلی خود آموزش می‌دهد. این فرآیند موجب می‌شود که داده‌های شخصی کاربران از دستگاه‌های آنان خارج نشود و فقط نتایج به دست آمده از مدل‌های محلی به سرور ارسال شود. سرور مرکزی با جمع‌آوری این نتایج، به دنبال یافتن الگوها و روابطی است که بتواند مدل کلی را بهبود بخشد. این مدل بهبود یافته سپس به کاربران ارسال می‌شود تا مجدداً با داده‌های محلی آنان آموزش داده شود.

در شکل ۳-۵، نمایی کلی از نحوه عملکرد یادگیری چندوظیفه‌ای در سیستم‌های فدرال به نمایش گذاشته شده است. این شکل به خوبی نشان می‌دهد که چگونه هر کاربر با استفاده از داده‌های محلی خود مدل را آموزش داده و نتایج را به سرور ارسال می‌کند و سرور با تحلیل این نتایج، مدل بهبود یافته را به کاربران بازمی‌گرداند.

به‌طور کلی، یادگیری چندوظیفه‌ای فدرال، به دلیل توانایی‌اش در تطبیق با داده‌های متنوع و محافظت از حریم شخصی کاربران، یک رویکرد بسیار مؤثر و کارآمد در زمینه یادگیری فدرال محسوب می‌شود.

<sup>۱</sup> Primal-Dual

images/chap3/multi\_tasking.png

شکل ۳-۵: یادگیری فدرال چندوظیفه‌ای [۳].

### ۳-۵-۳ یادگیری مادام‌العمر

رویکرد اصلی یادگیری مادام‌العمر به این صورت است که تلاش می‌کند در هر مرحله از الگوریتم، کاربرانی که برای اجرا انتخاب می‌شوند را به خاطر بسپارد. همان‌طور که پیش‌تر مطرح شد، در یادگیری فدرال ممکن است در هر مرحله تعداد کمی از کاربران انتخاب شوند. این مسئله باعث می‌شود که وزن‌ها و مدل‌هایی که برای کاربران جدید ارسال می‌شوند، لزوماً کارایی لازم را نداشته باشند. اما الگوریتم یادگیری مادام‌العمر تلاش دارد تا کاربران را به خاطر بسپارد و مدل‌های متناسب با هر کدام را ایجاد و به سمت آن‌ها ارسال کند [۴۰].

این رویکرد به این صورت عمل می‌کند که در هر مرحله از یادگیری، سوابق کاربران انتخاب شده را ذخیره می‌کند و از این سوابق برای بهبود و تطبیق مدل‌های آینده استفاده می‌کند. به این ترتیب، زمانی که کاربر جدیدی وارد فرآیند یادگیری می‌شود، الگوریتم می‌تواند از اطلاعات ذخیره شده قبلی استفاده کند و مدل بهتری را برای او ارسال کند. این روش باعث می‌شود که مدل‌ها به مرور زمان بهینه‌تر شده و عملکرد بهتری داشته باشند.

البته یکی از چالش‌های مهم در یادگیری مادام‌العمر، همین حفظ و به‌خاطر سپاری کاربران است. یادگیری فدرال به‌طور معمول با تعداد زیادی از کاربران سروکار دارد و حفظ سوابق همه این کاربران به‌طور همزمان می‌تواند منابع زیادی را مصرف کند و پیچیدگی‌های فنی زیادی را به همراه داشته باشد.

در نتیجه، یادگیری مادام‌العمر با ذخیره و استفاده از اطلاعات کاربران در طول زمان، می‌تواند به‌طور مؤثری به مدیریت چالش‌های مربوط به داده‌های غیرمستقل و غیریکنواخت در یادگیری فدرال کمک کند و همچنین به حفظ و بهبود کارایی مدل‌های یادگیری فدرال کمک نماید.

### ۳-۶ جمع‌بندی

یکی از چالش‌های مهم در یادگیری فدرال، تفاوت‌های موجود در داده‌های دستگاه‌های مختلف است که می‌تواند به تضعیف عملکرد مدل‌ها منجر شود. این مسئله محور اصلی این پژوهش را تشکیل می‌دهد. برای مقابله با این چالش، راه‌حل‌های مختلفی پیشنهاد شده است. از جمله این راه‌حل‌ها می‌توان به اشتراک‌گذاری داده‌ها، بهبود کیفیت داده‌ها با استفاده از شبکه‌های مولد و انتخاب داده‌ها بر اساس معیارهای کیفیت اشاره کرد. هر یک از این روش‌ها مزایا و محدودیت‌های خاص خود را دارند. برای مثال، اشتراک‌گذاری داده‌ها می‌تواند دقت مدل‌ها را افزایش دهد، اما ممکن است تهدیداتی برای حفظ حریم شخصی کاربران به همراه داشته باشد.

از سوی دیگر، راهکارهایی نیز بر پایه بهبود و به‌روزرسانی مدل‌ها وجود دارد. یکی از این روش‌ها استفاده از شبکه‌های عصبی عمیق و بهینه‌سازی تطبیقی است که با تنظیم نرخ یادگیری و بهینه‌سازی پارامترها بر اساس عملکرد کاربران، فرآیند آموزش را بهبود می‌بخشند. همچنین، روش‌های خوشه‌بندی مشابهت و تقطیر دانش برای کاهش پیچیدگی مدل‌ها و بهینه‌سازی ارتباطات در شبکه به کار گرفته می‌شوند. این رویکردها به گونه‌ای طراحی شده‌اند که ضمن بهبود عملکرد و دقت مدل‌ها، مصرف منابع و هزینه‌های ارتباطی را نیز کاهش دهند و فرآیند یادگیری را موثرتر سازند.

## فصل چهارم

### روش پیشنهادی برای جابه‌جایی مدل‌های شبکه عصبی بین کاربران

#### ۴-۱ مقدمه

در فصل گذشته، به بررسی روش‌های گوناگون برای مقابله با مشکل داده‌های non-IID در یادگیری فدرال پرداخته شد. در این فصل، تلاش خواهد شد با استفاده از جابه‌جایی مدل‌های شبکه عصبی بین کاربران نهایی، راهکاری برای پیاده‌سازی یادگیری فدرال بر روی داده‌های non-IID ارائه شود. شکل ۲-۲ به خوبی مفهوم داده‌های non-IID را توضیح می‌دهد، اما برای این که روشن شود چگونه جابه‌جایی مدل‌ها می‌تواند در این زمینه مؤثر باشند، در این بخش یک مثال مناسب ارائه می‌شود.

فرض کنید که هدف، آموزش مدلی برای شناسایی اشیائی مانند علائم ترافیکی و تابلوهای فروشگاهی است. اگر وسایل نقلیه، یکی در بزرگراه و دیگری در مرکز شهر حرکت کنند، داده‌های ویدیویی آن‌ها توزیع‌های متفاوتی از این علائم و تابلوها را ثبت خواهند کرد. به این صورت که داده‌های جمع‌آوری شده از بزرگراه، احتمالاً تابلوهای فروشگاهی کمتری را شامل می‌شود، در حالی که داده‌های ثبت شده در مرکز شهر حاوی تعداد بیشتری از هر دو نوع علائم و تابلوها هستند. این اختلاف توزیع داده‌ها در دستگاه‌های مختلف، می‌تواند منجر به مشکل انحراف در وزن‌دهی مدل شود.

برای حل این مسئله در پژوهش‌های پیشین، عملیات جابه‌جایی مدل‌های شبکه عصبی بین کاربران پیشنهاد

شده است. این رویکرد، مدل‌ها را بین دستگاه‌های نهایی جابه‌جا می‌کند تا تنوع داده‌ها در دستگاه‌های مختلف کاهش یابد. این فرایند با تحمیل هزینه اندک به منابع محاسباتی و ارتباطی، باعث بهبود مدل در برخورد با داده‌های non-IID می‌شود.

در این فصل، ابتدا به معرفی روش جابه‌جایی مدل‌ها پرداخته می‌شود که شامل دو نوع جابه‌جایی تصادفی و بر پایه شباهت است. روش جابه‌جایی بر پایه شباهت، در واقع همان رویکرد پیشنهادی این پژوهش محسوب می‌شود. سپس، تاثیرات این جابه‌جایی‌ها بر ترافیک شبکه و حفظ حریم شخصی، مورد بررسی قرار می‌گیرد. در ادامه، معیارهای شباهت و پایداری آن‌ها تعریف و ارزیابی می‌شوند. در پایان، شاخص‌های شباهت بین شبکه‌های عصبی مشخص شده و دو روش اصلی برای انتخاب کاربران جهت جابه‌جایی مدل‌ها تحلیل می‌شوند.

## ۴-۲ روش جابه‌جایی فدرال<sup>۱</sup> (FedSwap)

در این روش، یک عملیات جدید به نام جابه‌جایی فدرال یا FedSwap پیشنهاد شده است که به عنوان جایگزینی برای برخی از دوره‌های FedAvg در یادگیری فدرال به کار می‌رود. این عملیات با هدف بهبود فرآیند یادگیری فدرال و کاهش تاثیرات منفی داده‌های non-IID طراحی شده است [۹].

در روش یادگیری فدرال، پس از هر تکرار، مدل‌های محلی از دستگاه‌های نهایی جمع‌آوری شده و یک مدل جامع از ترکیب آن‌ها ساخته می‌شود. اما در روش FedSwap، به جای این که این ادغام در هر تکرار انجام شود، سرور مدل‌های محلی را در گام‌های مشخصی بین دستگاه‌ها جابه‌جا می‌کند. در واقع، در انتهای برخی مراحل، عملیات جابه‌جایی و در انتهای برخی دیگر، ادغام مدل‌ها صورت می‌گیرد. این مراحل بر اساس پارامترهایی که به عنوان ورودی تعریف می‌شوند، تعیین می‌گردند. جهت درک بهتر این ساختار به شکل ۴-۱ توجه نمایید.

برای حفظ توازن در این فرآیند، از یک استراتژی چرخشی<sup>۲</sup> استفاده می‌شود. در این استراتژی، به‌طور منظم، دو دستگاه نهایی به یکدیگر اجازه می‌دهند که مدل‌های خود را تبادل کنند. این کار باعث می‌شود که همه دستگاه‌های نهایی به‌طور مساوی در فرآیند تبادل مدل‌ها شرکت کنند و هیچ دستگاهی از مزایای این تبادل محروم نماند.

علاوه بر این، انتظار می‌رود که این عملیات جابه‌جایی مدل بین دستگاه‌های نهایی، به هر مدل دید گسترده‌تری از کل مجموعه داده‌ها بدهد. به عبارت دیگر، هر مدل محلی با تبادل مدل با دیگر دستگاه‌ها، می‌تواند اطلاعات بیشتری از داده‌های مختلف دریافت کند. این امر به کاهش انحراف وزن‌ها کمک می‌کند، زیرا مدل‌ها با داده‌های متنوع‌تری آموزش می‌بینند و به تدریج به یک مدل جامع‌تر و دقیق‌تر نزدیک می‌شوند.

<sup>۱</sup>Federated Swapping

<sup>۲</sup>Cyclic

به‌طور خلاصه، روش FedSwap با تبادل مدل‌های محلی بین دستگاه‌های نهایی، نه تنها به بهبود دقت و عملکرد مدل‌ها کمک می‌کند، بلکه مشکلات ناشی از داده‌های non-IID را نیز کاهش می‌دهد. این روش به عنوان یک رویکرد موثر در یادگیری فدرال می‌تواند باعث بهبود قابل توجهی در نتایج نهایی شود [۹].

جزئیات عملیات FedSwap در الگوریتم ۴-۱ ارائه شده است. همچنین در جدول ۴-۱ نمادهای مختص این الگوریتم به نمایش در آمده است. در این الگوریتم،  $w_t^k$  به عنوان وزن مدل در دستگاه نهایی  $k$  پس از گام  $t$  تنظیم می‌شود. در ابتدا، دستگاه‌های نهایی چندین به‌روزرسانی محلی انجام می‌دهند تا مدل‌های خود را بهبود بخشند. پس از هر  $h_1$  مرحله، سرور وارد عمل شده و عملیات جابه‌جایی را اجرا می‌کند. در این مرحله، مدل‌های محلی بین دستگاه‌های نهایی تبادل می‌شوند تا هر دستگاه بتواند از مدل‌های متنوع‌تری برای آموزش استفاده کند.

این تبادل مدل‌ها به کاهش تنوع داده‌ها بین دستگاه‌های مختلف کمک می‌کند و باعث می‌شود که مدل‌ها با داده‌های مختلفی آموزش ببینند. پس از انجام  $h_2$  عملیات جابه‌جایی، سرور وارد عمل شده و عملیات میانگین‌گیری را اجرا می‌کند. در این مرحله، سرور مدل‌های محلی را جمع می‌کند تا یک مدل مشترک ایجاد شود که از داده‌های تمام دستگاه‌ها بهره می‌برد.

برای تعیین مقادیر  $h_1$  و  $h_2$ ، ابتدا آزمایش‌های مختلفی انجام شده و بر اساس نتایج به دست آمده به‌صورت تجربی، مقادیر نهایی انتخاب شده‌اند. در این آزمایش‌ها، چند نکته مهم مشاهده شده است. ابتدا، مقدار  $h_1$  به عملکرد به‌روزرسانی مدل محلی در دستگاه‌های نهایی وابسته است. از آن جایی که وظیفه یادگیری معمولاً یک وظیفه عمومی مثل طبقه‌بندی است، مقدار  $h_1$  بر اساس مقدار گام تعریف‌شده در روش میانگین‌گیری فدرال سنتی

images/chap4/federated\_swapping.png

شکل ۴-۱: روش جابه‌جایی فدرال [۹].



---

```

1 Initialize all clients model with weight  $w_0$ ;
2 for  $t = 1, 2, \dots, T$  do
3   for each client  $k = 1, 2, \dots, K$  in parallel do
4      $w_t^k = w_{t-1}^k - \eta \nabla F(w_{t-1}^k)$ ;
5   end
6   if  $t|h_1 = 0$  and  $t|h_1 h_2 \neq 0$  then
7     for each client  $k = 1, 2, \dots, K$  do
8        $w_t^k \leftarrow \text{Swapping}(k, \{w_t^k\}_{k \in K})$ ;
9     end
10  end
11  if  $t|h_1 h_2 = 0$  then
12     $w_t \leftarrow \text{WeightedAvg}(\{w_t^k\}_{k \in K})$ ;
13    for each client  $k = 1, 2, \dots, K$  in parallel do
14       $w_t^k \leftarrow w_t$ ;
15    end
16  end
17 end
18 Function  $\text{Swapping}(k, \{w_t^k\}_{k \in K})$ :
19    $r$  represent a random client in  $K$ ;
20    $w_t \leftarrow w_t^r$ ;
21    $w_t^r \leftarrow w_t^k$ ;
22   return  $w_t$ ;
23 end
24 Function  $\text{WeightedAvg}(\{w_t^k\}_{k \in K})$ :
25    $w_t \leftarrow \sum_{k=1}^K \frac{n_k}{n} w_t^k$ ;
26   return  $w_t$ ;
27 end

```

---

جدول ۴-۱: نمادهای مختص الگوریتم FedSwap

متغیر	توضیحات
$h_1$	تعداد گام‌ها بین هر جابه‌جایی
$h_2$	تعداد جابه‌جایی‌ها بین هر میانگین‌گیری

تنظیم می‌شود [۹].

علاوه بر این، مقدار  $h_2$  نقش مهمی در توازن بین سربار ارتباطی و همگرایی مدل ایفا می‌کند. با افزایش مقدار  $h_2$ ، تعداد دفعات جابه‌جایی فدرال بین دستگاه‌های نهایی بیشتر خواهد شد. این امر می‌تواند با کاهش تعداد دفعات ادغام فدرال، صرفه‌جویی بیشتری در پهنای باند ارتباطی ایجاد کند. با این حال، این امر ممکن است باعث افزایش احتمال انحراف وزن‌ها و کاهش دقت همگرایی مدل سراسری شود. به عبارت دیگر، هرچه مقدار  $h_2$  بزرگتر باشد، تعداد دفعاتی که مدل‌ها بین دستگاه‌های نهایی جابه‌جا می‌شوند بیشتر است و این ممکن است به بهبود عملکرد مدل‌ها در مواجهه با داده‌های non-IID کمک کند، اما ریسک انحراف وزن‌ها نیز بیشتر خواهد شد، زیرا مرحله ادغام مدل‌ها به تاخیر خواهد افتاد.

از سوی دیگر، اگر مقدار  $h_2$  کوچکتر باشد، فراوانی جابه‌جایی فدرال بین دستگاه‌های نهایی کاهش می‌یابد.

این امر منجر به افزایش سربارهای ارتباطی می‌شود، زیرا نیاز به ادغام مکرر مدل سراسری خواهد بود. بنابراین، مقدار  $h_2$  باید به گونه‌ای تنظیم شود که توازن مناسبی بین کاهش سربار ارتباطی و حفظ دقت مدل ایجاد کند. در مجموع، روش FedSwap با جابه‌جایی مدل‌های محلی بین دستگاه‌های نهایی بدون نیاز به هزینه‌های محاسباتی و ارتباطی اضافی، می‌تواند به بهبود عملکرد مدل‌ها در مواجهه با داده‌های non-IID کمک کند [۹].

#### ۳-۴ نحوه جابه‌جایی مدل‌ها در یادگیری فدرال

در این بخش، دو روش متفاوت برای جابه‌جایی مدل‌ها در یادگیری فدرال مورد بررسی قرار می‌گیرد. روش جابه‌جایی تصادفی که مدل‌ها را بر حسب تصادف بین دستگاه‌ها جابه‌جا می‌کند و روش مبتنی بر شباهت که با مقایسه مدل‌ها بر اساس شباهت‌ها و تفاوت شبکه‌های عصبی، تصمیم به مبادله می‌گیرد. هر دو روش به منظور ارتقای عملکرد مدل‌ها در شرایطی که داده‌ها non-IID هستند، طراحی شده‌اند.

##### ۱-۳-۴ روش جابه‌جایی فدرال به صورت تصادفی

در الگوریتم FedSwap، مدل‌های محلی به‌طور کاملاً تصادفی بین دستگاه‌های نهایی مبادله می‌شوند. به بیان دیگر، هر بار که قرار است دو دستگاه مدل‌های خود را با یکدیگر مبادله کنند، انتخاب این دستگاه‌ها به صورت تصادفی صورت می‌گیرد. این باعث می‌شود که هیچ الگوی ثابتی در جابه‌جایی مدل‌ها وجود نداشته باشد و در هر مرتبه، ترکیب جدیدی از دستگاه‌ها در فرآیند تبادل مدل‌ها شرکت کنند.

یکی از ویژگی‌های مهم الگوریتم FedSwap، این است که تمام دستگاه‌های نهایی به صورت مساوی در فرآیند جابه‌جایی شرکت می‌کنند. به عبارت دیگر، همه دستگاه‌های شرکت کننده در این مرحله از اجرا، در فرآیند جابه‌جایی قرار می‌گیرند، اما انتخاب دستگاه‌ها برای جابه‌جایی به صورت تصادفی صورت می‌پذیرد. این باعث می‌شود که تمامی دستگاه‌ها فرصت مساوی برای تبادل مدل‌ها و بهبود دقت و عملکرد خود داشته باشند.

با استفاده از این رویکرد، الگوریتم FedSwap قادر است به بهبود عملکرد مدل‌های محلی کمک کند، زیرا تبادل تصادفی مدل‌ها بین دستگاه‌ها باعث می‌شود که هر دستگاه به داده‌ها و اطلاعات بیشتری دسترسی پیدا کند. این امر به کاهش انحراف وزن‌ها و بهبود همگرایی مدل سراسری کمک می‌کند.

##### ۲-۳-۴ روش پیشنهادی جابه‌جایی فدرال بر پایه شباهت<sup>۱</sup> (SimFedSwap)

همان‌گونه که پیش‌تر بیان شد، هدف اصلی این پژوهش، انتخاب دستگاه‌های نهایی بر اساس میزان شباهت مدل‌های شبکه عصبی و جابه‌جایی آن‌ها با یکدیگر است. برای این منظور، باید به‌طور کامل با ساختار شبکه

<sup>۱</sup> Similarity-based Federated Swapping

عصبی آشنا بوده و مدل‌های مختلف را با یکدیگر مقایسه کرد. این مقایسه امکان ارزیابی میزان شباهت و تفاوت بین مدل‌های شبکه عصبی را فراهم می‌کند.

بعد از بررسی و تعیین میزان شباهت مدل‌ها، باید تصمیم گرفت که کدام یک از آن‌ها را با یکدیگر جابه‌جا نمود. در روش پیشنهادی SimFedSwap بهترین انتخاب برای جابه‌جایی، مدلی است که کمترین شباهت را با مدل شبکه عصبی دستگاه فعلی دارد. دلیل این انتخاب این است که اگر مدل دستگاه فعلی با مدل دستگاه مقصد شباهت زیادی داشته باشد، جابه‌جایی آن‌ها مؤثر نخواهد بود. این شباهت بالا به این معناست که این دو دستگاه نهایی داده‌های مشابهی داشته و در طول زمان آموزش‌های مشابهی دیده‌اند، در نتیجه جابه‌جایی مدل‌ها تأثیر قابل‌توجهی بر بهبود یادگیری نخواهد داشت.

بنابراین، برای توضیح دلیل ارائه این روش، می‌توان به این نکته اشاره کرد که جابه‌جایی مدل‌هایی که کمترین شباهت را بین دستگاه‌های مختلف دارند، به بهبود فرآیند یادگیری کمک خواهند کرد. این فرض بر این اساس است که دستگاه‌هایی با مدل‌های متفاوت، احتمالاً داده‌هایی با ساختارهای متفاوت دارند. پس جابه‌جایی مدل‌ها بین این دستگاه‌ها، مدل‌ها را با داده‌های جدیدی روبه‌رو می‌کند که می‌تواند به یادگیری بهتر و متنوع‌تر کمک کند. در نتیجه، مدل سراسری سریع‌تر به سمت مسیر بهینه همگرا می‌شود و دقت و کارایی آن افزایش می‌یابد.

این روش نه تنها تنوع داده‌ها را در فرآیند یادگیری افزایش می‌دهد، بلکه به کاهش انحراف وزن‌ها نیز کمک می‌کند. با داشتن دید گسترده‌تری از داده‌ها و تجربیات مختلف، مدل‌ها می‌توانند بهتر و جامع‌تر آموزش ببینند. این امر در نهایت منجر به بهبود عملکرد کلی مدل در شرایط واقعی می‌شود و کمک می‌کند که مدل‌های یادگیری فدرال بتوانند با چالش‌های داده‌های non-IID به نحو بهتری مقابله کنند.

نحوه مدیریت بار محاسباتی در روش SimFedSwap به این صورت است که ابتدا تمام مدل‌های شبکه عصبی از دستگاه‌های نهایی به سمت سرور ارسال می‌شوند. سپس سرور بر اساس معیارهای مشخصی، شباهت بین این مدل‌ها را بررسی و اقدام به جابه‌جایی آن‌ها می‌کند. در این روش، تمامی عملیات پردازشی بر روی سرور انجام شده و تصمیم‌گیری درباره تبادل مدل‌ها نیز به عهده سرور خواهد بود. این با فرض محدودیت دستگاه‌های نهایی از لحاظ سخت‌افزار و منابع در دسترس سازگار است.

با انجام عملیات بر روی سرور، دستگاه‌های نهایی تنها به تبادل داده‌های لازم و اجرای به‌روزرسانی‌های محلی سبک می‌پردازند. این رویکرد باعث خواهد شد که فرآیند یادگیری بهینه‌تری ایجاد شود و مدل‌ها به‌طور مؤثر و کارآمدتری آموزش ببینند. در نتیجه، مشکلات محاسباتی به حداقل می‌رسد و عملکرد کلی سیستم بهبود خواهد یافت.

این روش نه تنها به حفظ منابع محدود دستگاه‌های نهایی کمک می‌کند، بلکه بهره‌وری بالاتری نیز از قدرت پردازشی سرور به دست می‌آید. به این ترتیب، می‌توان اطمینان داشت که عملیات‌های پیچیده و محاسبات سنگین به درستی و با سرعت مناسب انجام می‌شوند، بدون این که فشار اضافی بر دستگاه‌های نهایی وارد شود. به این ترتیب، این روش می‌تواند به‌طور موثری در محیط‌های مختلف با دستگاه‌های متنوع و منابع محدود پیاده‌سازی شود و نتایج قابل اعتمادی ارائه دهد.

#### ۴-۴ تعریف معیار شباهت

فرض کنید  $X$  ماتریسی با ابعاد  $n \times p_1$  باشد که  $X \in \mathbb{R}^{n \times p_1}$  شامل  $n$  نمونه و  $p_1$  ویژگی است. همچنین،  $Y$  ماتریسی با ابعاد  $n \times p_2$  باشد که  $Y \in \mathbb{R}^{n \times p_2}$  شامل  $n$  نمونه و  $p_2$  ویژگی است. فرض می‌شود که  $p_1$  کمتر یا مساوی  $p_2$  است.

هدف طراحی و تحلیل یک شاخص شباهت عددی  $s(X, Y)$  است که بتواند بازنمایی‌های<sup>۱</sup> موجود در ماتریس‌های  $X$  و  $Y$  را هم درون یک شبکه عصبی و هم بین شبکه‌های عصبی مختلف مقایسه کند. چنین شاخصی به درک بهتر تأثیر عوامل مختلف در یادگیری عمیق کمک می‌کند.

به عنوان مثال، در بررسی شبکه‌های عصبی، ماتریس  $X$  می‌تواند نمایانگر فعال‌سازهای<sup>۲</sup> نورون‌ها در یک لایه خاص برای  $n$  نمونه ورودی باشد و ماتریس  $Y$  می‌تواند نمایانگر فعال‌سازهای نورون‌ها در لایه‌ای دیگر یا حتی در یک شبکه عصبی دیگر برای همان  $n$  نمونه باشد. مقایسه این دو ماتریس اطلاعات مهمی درباره نحوه یادگیری و بازنمایی داده‌ها توسط شبکه عصبی ارائه می‌دهد.

شاخص  $s(X, Y)$  باید توانایی اندازه‌گیری شباهت‌ها و تفاوت‌های بین بازنمایی‌های مختلف را داشته باشد. این شاخص می‌تواند به پژوهشگران کمک کند تا نحوه تغییر بازنمایی‌ها در اثر عوامل مختلف مانند تغییرات در داده‌های ورودی، تغییرات در معماری شبکه یا تغییرات در پارامترهای آموزش را بهتر درک کنند. طراحی و تحلیل این شاخص شباهت می‌تواند به درک بهتر از نحوه عملکرد شبکه‌های عصبی کمک نموده و ابزار مفیدی برای بهبود روش‌های آموزش و بهینه‌سازی شبکه‌های عصبی فراهم کند.

#### ۴-۵ پایداری در معیارهای شباهت

در این بخش، ویژگی‌های ضروری برای معیارهای مقایسه بازنمایی‌های شبکه عصبی مورد بررسی قرار می‌گیرند. این بررسی شامل تحلیل پایداری شاخص‌های شباهت و تأثیرات آن‌ها در ارزیابی شباهت بازنمایی‌های شبکه

<sup>1</sup>Representations

<sup>2</sup>Activations

عصبی است. همچنین، به اهمیت معیارهای شباهتی پرداخته می‌شود که نسبت به تبدیل‌های متعامد<sup>۱</sup> و مقیاس‌بندی یکسان<sup>۲</sup> پایدار هستند. این ویژگی‌ها به معیار شباهت امکان می‌دهند تا بازنمایی‌های شبکه عصبی را به درستی مقایسه کرده و تأثیرات مختلف در فرآیند آموزش شبکه عصبی را بهتر درک کند.

#### ۴-۵-۱ پایداری نسبت به تبدیل متعامد

پایداری نسبت به تبدیل‌های متعامد به این معناست که اگر  $s(X, Y)$  یک شاخص شباهت بین دو ماتریس  $X$  و  $Y$  باشد، این شاخص باید در مقابل تغییرات متعامد نیز پایدار باقی بماند. به عبارت دیگر، اگر  $U$  و  $V$  ماتریس‌های متعامد با رتبه کامل<sup>۳</sup> باشند که شرط  $U^T U = I$  و  $V^T V = I$  را برآورده کنند، باید  $s(X, Y) = s(XU, YV)$  باشد. این ویژگی تضمین می‌کند که حتی در صورتی که ابعاد  $p$  بزرگتر از  $n$  باشند، شاخص شباهت همچنان به‌طور مطلوب عمل می‌کند. علاوه بر این، تبدیل‌های متعامد خواص مهمی از جمله حفظ حاصل ضرب‌های عددی و فاصله‌های اقلیدسی<sup>۴</sup> بین نمونه‌ها را نیز حفظ می‌کنند. این امر باعث می‌شود که مقایسه‌های انجام شده توسط این شاخص‌ها دقیق و قابل اعتماد باشند [۴۱].

پایداری نسبت به تبدیل‌های متعامد برای شبکه‌های عصبی که با استفاده از روش نزول گرادیان آموزش داده می‌شوند، بسیار مطلوب است. این ویژگی نه تنها پایداری نسبت به تغییرات متعامد را تضمین می‌کند بلکه شامل پایداری نسبت به جایگشت نیز می‌شود. جایگشت در یک ماتریس به معنای این است که مقدارهای درون ماتریس فقط جابه‌جا می‌شوند و ارزش‌های آن‌ها ثابت باقی می‌مانند. این پایداری برای تطبیق تقارن‌های شبکه‌های عصبی ضروری است [۴۲، ۴۳].

در حالت خطی، اگر ورودی‌ها با یک تبدیل متعامد تغییر کنند، روند آموزش با روش نزول گرادیان تحت تأثیر قرار نمی‌گیرد. برای شبکه‌های عصبی که با وزن‌های متقارن و تصادفی شروع می‌شوند، تبدیل‌های متعامد بر فعال‌سازها باعث می‌شود که روند آموزشی مشابه حالت بدون تغییر باقی بماند. اما اگر یک تغییر خطی دلخواه انجام شود، این ویژگی حفظ نمی‌شود و ممکن است روند آموزش تحت تأثیر منفی قرار گیرد [۴۴].

به‌طور کلی، پایداری نسبت به تبدیل‌های متعامد در شبکه‌های عصبی اهمیت زیادی دارد زیرا این ویژگی کمک می‌کند تا شبکه‌های عصبی در مواجهه با تغییرات متقارن در داده‌ها، به درستی عمل کنند و دقت و کارایی آن‌ها در فرآیند آموزش بهینه باقی بماند.

<sup>1</sup> Orthogonal Transformation

<sup>2</sup> Isotropic Scaling

<sup>3</sup> Full Rank

<sup>4</sup> Euclidean Distances

#### ۴-۵-۲ پایداری نسبت به مقیاس بندی یکسان

شاخص های شباهت باید هنگام مقیاس بندی یکسان ورودی ها، ثابت بمانند. به این معنا که اگر ورودی ها در اعداد مثبتی مانند  $\alpha$  و  $\beta$  ضرب شوند، نباید تغییری در شاخص شباهت ایجاد شود. به عبارت دیگر، مقدار  $s(X, Y)$  باید همانند  $s(\alpha X, \beta Y)$  باقی بماند و برای هر  $\alpha$  و  $\beta$  مثبت، درست باشد.

این ویژگی اهمیت خاصی دارد، زیرا تضمین می کند که مقایسه بازنمایی های شبکه های عصبی تحت تأثیر مقیاس بندی یکسان قرار نمی گیرد و دقت شاخص حفظ می شود. در واقع، این شاخص ها قادرند در شرایطی که شبکه های عصبی تحت تغییرات یکسان مقیاس قرار می گیرند، همچنان به درستی بازنمایی های مختلف را مقایسه کنند [۴۱].

به عنوان مثال، وقتی شبکه های عصبی در معرض تغییرات یکسان مقیاس قرار می گیرند، این شاخص ها همچنان قادر خواهند بود بازنمایی های مختلف را به درستی مقایسه کنند. این ویژگی امکان فهم بهتر تأثیرات گوناگون در طول آموزش شبکه های عصبی و استفاده از این شاخص ها برای تحلیل و بهبود عملکرد مدل ها را فراهم می کند. بنابراین، شاخص های مقاوم در برابر مقیاس بندی یکسان می توانند ابزار مفیدی برای ارزیابی و بهینه سازی شبکه های عصبی باشند.

#### ۴-۶ مقایسه ساختارهای مشابهت

یکی از چالش های اصلی در تحلیل بازنمایی های شبکه های عصبی، مقایسه ویژگی های چندگانه هر نمونه در بازنمایی های مختلف است. این روش ممکن است پیچیده و زمان بر باشد و نتایج گمراه کننده ای ایجاد کند. برای حل این مشکل، می توان از رویکردی استفاده کرد که به جای مقایسه مستقیم ویژگی های هر نمونه، ساختارهای شباهتی بین نمونه ها را بررسی کند.

ایده اصلی این است که به جای مقایسه مستقیم ویژگی های چندگانه هر نمونه در دو بازنمایی مختلف، می توان ابتدا شباهت بین هر جفت نمونه در هر بازنمایی را به صورت جداگانه سنجید و سپس این ساختارهای شباهتی را با هم مقایسه کرد [۴۱].

برای درک بهتر این موضوع، تصور کنید که به جای مقایسه مستقیم ویژگی های چندبعدی دو نمونه، ابتدا میزان شباهت هر یک از این نمونه ها به سایر نمونه ها بررسی می شود. سپس، این ماتریس های شباهت، که میزان شباهت هر نمونه به دیگر نمونه ها را نشان می دهند، با یکدیگر مقایسه می شوند.

نکته مهم این است که اگر برای اندازه گیری شباهت از ضرب داخلی استفاده شود، شباهت بین ماتریس های بازنمایی به یک مفهوم دیگر و قابل درک از شباهت بین ویژگی های جفتی تبدیل می شود. به عبارت دیگر، این

روش امکان دستیابی به درک دقیق‌تری از شباهت بین ویژگی‌ها را بدون مقایسه مستقیم ویژگی‌های چندگانه هر نمونه فراهم می‌کند. این رویکرد می‌تواند به‌طور قابل توجهی در تحلیل و درک بازنمایی‌های شبکه‌های عصبی و داده‌های پیچیده مؤثر باشد، زیرا ساختارهای پیچیده را به شیوه‌ای ساده‌تر و قابل فهم‌تر بررسی می‌کند [۴۱].

#### ۴-۶-۱ ضرب داخلی<sup>۱</sup>

یک رابطه ساده وجود دارد که ضرب داخلی بین نمونه‌ها را با ضرب داخلی بین ویژگی‌ها مرتبط می‌سازد:

$$\langle \text{vec}(XX^T), \text{vec}(YY^T) \rangle = \text{tr}(XX^T YY^T) = \|Y^T X\|_F^2 \quad (۱-۴)$$

که در آن عناصر  $XX^T$  و  $YY^T$  نشان‌دهنده ضرب داخلی بین بازنمایی نمونه‌های  $i$  و  $j$  هستند و شباهت بین این نمونه‌ها را بر اساس شبکه‌های مربوطه نشان می‌دهند.

به بیان دیگر، بخش چپ رابطه (۱-۴)، میزان مشابهت بین الگوهای شباهت، میان نمونه‌ها را ارزیابی می‌کند. این در حالی است که سمت راست، با جمع کردن مربعات ضرب‌های داخلی بین هر جفت از نمونه‌ها، به همان نتیجه مشابه می‌رسد و شباهت بین ویژگی‌های  $X$  و  $Y$  را اندازه‌گیری می‌کند.

این رابطه نشان می‌دهد که می‌توان به‌جای مقایسه مستقیم ویژگی‌ها، از شباهت‌های بین نمونه‌ها استفاده کرد تا به فهم بهتری از بازنمایی‌های شبکه‌های عصبی و داده‌های پیچیده دست یافت. به این ترتیب، تحلیل و درک داده‌ها ساده‌تر و مؤثرتر می‌شود، زیرا این روش امکان دستیابی به نتایج دقیق‌تر را با استفاده از شباهت‌های موجود بین نمونه‌ها به‌صورت غیرمستقیم فراهم می‌کند.

#### ۴-۶-۲ انتخاب هسته<sup>۲</sup>

در معیارهای مشابهت و اندازه‌گیری وابستگی، مفهوم هسته یا kernel نقش بسیار مهمی دارد. هسته در واقع یک تابع ریاضی است که برای محاسبه شباهت بین داده‌های ورودی استفاده می‌شود. این تابع، داده‌ها را به یک فضای ویژگی بالاتر نگاشت می‌کند تا بتوان همبستگی‌ها و مشابهت‌های پیچیده‌تر بین آن‌ها را بهتر سنجید.

به بیان ساده‌تر، تابع هسته  $k$  یک تابع مثبت معین است که دو بردار ورودی  $x_i$  و  $x_j$  را گرفته و یک عدد حقیقی تولید می‌کند که نشان‌دهنده میزان شباهت بین این دو بردار است. هسته‌ها می‌توانند به شکل‌های مختلفی باشند که هر کدام ویژگی‌ها و کاربردهای خاص خود را دارند. در این جا با اقتباس از [۴۱] به چند نمونه رایج اشاره می‌شود.

<sup>۱</sup>Inner Product

<sup>۲</sup>Kernel

- هسته خطی<sup>۱</sup>: این هسته به سادگی ضرب داخلی دو بردار ورودی را محاسبه می‌کند.

$$k(x_i, x_j) = x_i^T x_j \quad (۲-۴)$$

- هسته چندجمله‌ای<sup>۲</sup>: این هسته ضرب داخلی را با یک توان مثبت، بالا می‌برد.

$$k(x_i, x_j) = (x_i^T x_j + c)^d \quad (۳-۴)$$

که در آن  $c$  یک ثابت و  $d$  درجه چندجمله‌ای است.

- هسته گاوسی (RBF)<sup>۳</sup>: این هسته فاصله اقلیدسی بین دو بردار را در یک تابع نمایی قرار می‌دهد که باعث می‌شود داده‌هایی که نزدیک به هم هستند شباهت بیشتری داشته باشند.

$$k(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right) \quad (۴-۴)$$

که در آن  $\sigma$  پارامتر پهنای باند و تنظیم‌کننده میزان تاثیر فاصله می‌باشد.

برای هسته RBF، چندین استراتژی مختلف برای انتخاب پهنای باند  $\sigma$  وجود دارد که میزان تأکید بر شباهت فواصل کوچک نسبت به فواصل بزرگ را کنترل می‌کند. پارامتر  $\sigma$  به عنوان کسری از فاصله میانه، بین نمونه‌ها تنظیم می‌شود. در عمل، مشاهده می‌شود که هسته‌های RBF و خطی در بیشتر آزمایش‌ها نتایج مشابهی ارائه می‌دهند [۴۱].

استفاده از توابع هسته‌ای در معیارهای شباهت و وابستگی، این امکان را فراهم می‌کند که داده‌های ورودی به یک فضای ویژگی بالاتر نگاشت شوند، جایی که روابط پیچیده و غیرخطی بین داده‌ها می‌توانند به صورت ساده‌تری مدل‌سازی شوند. به این ترتیب، معیارها با بهره‌گیری از هسته‌ها می‌توانند تحلیل دقیق‌تر و کارآمدتری از داده‌های پیچیده ارائه دهند. این ویژگی باعث می‌شود که هسته‌ها ابزار قدرتمندی در تحلیل داده‌ها باشند.

## ۷-۴ معیارهای سنجش شباهت

در این بخش به بررسی روش‌های مختلفی که برای سنجش شباهت بین بازنمایی‌های شبکه‌های عصبی استفاده می‌شود، پرداخته خواهد شد. یکی از روش‌های مهم در این زمینه استفاده از پایه‌های متعامد است. به طور خاص، فرض می‌شود که  $Q_X$  و  $Q_Y$  پایه‌های متعامدی برای ستون‌های ماتریس‌های  $X$  و  $Y$  هستند. به این معنا که  $Q_X$  و  $Q_Y$  به صورت  $Q_X = X(X^T X)^{-1/2}$  و  $Q_Y = Y(Y^T Y)^{-1/2}$  تعریف شده‌اند. این پایه‌های متعامد امکان

<sup>1</sup>Linear Kernel

<sup>2</sup>Polynomial Kernel

<sup>3</sup>Radial Basis Function Kernel



تحلیل مؤثرتر بازنمایی‌های شبکه عصبی و سنجش دقیق‌تر شباهت‌های آن‌ها را فراهم می‌کنند. تمامی رابطه‌های مورد استفاده در بخش ۴-۷، از مرجع [۴۱] برگرفته شده‌اند.

استفاده از پایه‌های متعامد این امکان را فراهم می‌کند که بدون نگرانی از وابستگی‌های خطی بین ستون‌ها، شباهت‌ها به صورت مستقیم مقایسه شوند. این رویکرد به ویژه زمانی مفید است که هدف بررسی نحوه استخراج و بازنمایی ویژگی‌های مختلف داده‌ها توسط شبکه عصبی باشد. با این روش، می‌توان تحلیل‌های دقیقی انجام داد تا تأثیر تغییرات در داده‌های ورودی یا ساختار شبکه بر بازنمایی‌های داخلی بهتر درک شود. چنین تحلیل‌هایی می‌توانند در بهبود و بهینه‌سازی شبکه‌های عصبی و الگوریتم‌های یادگیری عمیق نقش بسزایی داشته باشند.

#### ۴-۷-۱ قرینه مجموع اختلاف مطلق<sup>۱</sup> (OSAD)

معیار سنجش شباهت باید به گونه‌ای تعریف شود که علاوه بر داشتن پایداری در مقابل تبدیل‌های متعامد و مقیاس‌بندی یکسان، قادر باشد ماتریس‌هایی با ابعاد مختلف را نیز پوشش دهد. با این حال، در صورتی که فرض شود ماتریس‌های مورد بررسی دارای ساختار یکسان و ابعاد ثابت هستند و همچنین مقدارهای اولیه این ماتریس‌ها یکسان در نظر گرفته شوند، می‌توان از معیار OSAD استفاده نمود.

در صورتی که ابعاد به شکل  $n \times p$  تعریف شوند، از رابطه زیر برای مقایسه و اندازه‌گیری شباهت استفاده می‌شود:

$$OSAD = - \sum_{i=1}^n \sum_{j=1}^{p_1} Z_{ij} \quad \text{where} \quad Z = |X - Y| \quad (۵-۴)$$

در این رابطه،  $X$  و  $Y$  ماتریس‌های ورودی هستند و  $Z$  ماتریسی است که از اختلاف مطلق بین این دو به دست می‌آید. در نهایت قرینه مجموع مقادیر ماتریس  $Z$  که با OSAD نشان داده می‌شود، به عنوان معیاری برای سنجش میزان شباهت مورد استفاده قرار می‌گیرد.

با توجه به این که در این روش فرض بر این است که ابعاد ماتریس‌ها و مقدارهای اولیه ثابت و یکسان هستند، معیار OSAD به عنوان ابزاری مفید و قابل اعتماد برای مقایسه و ارزیابی شباهت بین دو ماتریس مختلف مطرح می‌شود. این معیار به سادگی اختلاف‌های موجود در مقادیر را محاسبه کرده و قرینه مجموع آن‌ها را به عنوان نتیجه ارائه می‌دهد، که در تحلیل‌های مختلف بسیار کاربردی خواهد بود.

<sup>۱</sup> Opposite Sum of Absolute Difference

#### ۴-۷-۲ تحلیل همبستگی کانونی<sup>۱</sup> (CCA)

برای درک بهتر تحلیل همبستگی کانونی، از یک مثال ساده استفاده می‌شود. فرض کنید دو مجموعه داده مختلف در اختیار است، مجموعه‌ای شامل اطلاعاتی همچون قد و وزن افراد و مجموعه دیگر حاوی اطلاعاتی مانند سن و درآمد آن‌ها باشد. هدف تحلیل همبستگی کانونی، این است که ارتباط‌های پنهان بین این دو مجموعه داده را کشف کند. به بیان ساده، CCA به دنبال شناسایی ترکیب‌هایی از ویژگی‌ها در هر مجموعه داده است که با مقایسه آن‌ها، بیشترین همبستگی حاصل شود. تلاش بر این است که مشخص شود کدام ترکیب قد و وزن با کدام ترکیب سن و درآمد بیشترین ارتباط را دارد.

ابتدا داده‌ها استاندارد می‌شوند، به این صورت که میانگین هر ویژگی صفر شده و داده‌ها به گونه‌ای تغییر می‌کنند که انحراف معیارشان یک شود. سپس، CCA بردارهای وزنی را برای هر مجموعه داده محاسبه می‌کند تا ترکیب‌های خطی از این داده‌ها ایجاد کند. این ترکیب‌ها به گونه‌ای انتخاب می‌شوند که حداکثر همبستگی بین آن‌ها وجود داشته باشد. به عنوان مثال، CCA می‌خواهد ترکیبی از قد و وزن (مثلاً  $0.5 \times \text{وزن} + 0.5 \times \text{قد}$ ) و ترکیبی از سن و درآمد (مثلاً  $0.7 \times \text{درآمد} + 0.3 \times \text{سن}$ ) را بیابد که بیشترین ارتباط را با هم داشته باشند. با استفاده از این بردارهای وزنی، ترکیب‌های جدیدی از داده‌ها ایجاد می‌شوند و سپس همبستگی بین این ترکیب‌ها محاسبه می‌شود. CCA به دنبال یافتن پایه‌هایی برای دو ماتریس است به طوری که وقتی ماتریس‌های اصلی بر روی این پایه‌ها توزیع می‌شوند، همبستگی به حداکثر برسد. برای هر  $i$  که بین ۱ تا  $p_1$  (تعداد ویژگی‌ها) قرار دارد، ضریب همبستگی کانونی  $\rho_i$  به صورت زیر تعریف می‌شود:

$$\rho_i = \max_{\mathbf{w}_X^i, \mathbf{w}_Y^i} \text{corr}(X\mathbf{w}_X^i, Y\mathbf{w}_Y^i) \quad (۴-۶)$$

با در نظر گرفتن بردارهای  $\mathbf{w}_X^i \in \mathbb{R}^{p_1}$  و  $\mathbf{w}_Y^i \in \mathbb{R}^{p_2}$ ، ضریب همبستگی کانونی  $\rho_i$  هدفش این است که همبستگی بین ترکیب خطی  $X\mathbf{w}_X^i$  و  $Y\mathbf{w}_Y^i$  را به حداکثر برساند.

برای اطمینان از این که ترکیب‌های جدید داده‌ها مستقل و متفاوت از هم باشند، شرط‌های زیر باید رعایت شوند:

$$\begin{aligned} \forall j < i \quad X\mathbf{w}_X^i &\perp X\mathbf{w}_X^j \\ \forall j < i \quad Y\mathbf{w}_Y^i &\perp Y\mathbf{w}_Y^j \end{aligned} \quad (۴-۷)$$

این شرط‌ها اطمینان می‌دهند که ترکیب‌های جدید از داده‌ها با ترکیب‌های قبلی همپوشانی نداشته و متعامد باقی بمانند.

در نهایت برای مقایسه دو شبکه عصبی و اندازه‌گیری شباهت بین آن‌ها، از معیاری به نام  $R_{CCA}^2$  استفاده

<sup>۱</sup> Canonical Correlation Analysis

می‌شود. این معیار نشان می‌دهد که چقدر از اطلاعات داده‌ها، توسط ترکیب‌های خطی حاصل از روش CCA توضیح داده می‌شود. رابطه این معیار به این صورت است:

$$R_{CCA}^2 = \frac{\sum_{i=1}^{p1} \rho_i^2}{p1} = \frac{\|Q_Y^T Q_X\|_F^2}{p1} \quad (۸-۴)$$

که با محاسبه و جمع کردن مربعات ضرایب همبستگی کانونی و سپس تقسیم آن‌ها بر تعداد ضرایب، میزان شباهت بین دو شبکه عصبی را ارزیابی می‌کند.

با استفاده از روش CCA، می‌توان ترکیب‌های خطی از ویژگی‌های دو مجموعه داده مختلف را شناسایی کرد که بالاترین همبستگی را با هم دارند. این روش کمک می‌کند تا روابط پنهان و مهم بین هر دو مجموعه داده آشکار شود و تحلیل‌های دقیق‌تری صورت گیرد.

#### ۳-۷-۴ معیار استقلال هیلبرت-اشمیت<sup>۱</sup> (HSIC)

برای بررسی میزان وابستگی و شباهت بین دو مجموعه داده، می‌توان از معیار استقلال هیلبرت-اشمیت استفاده کرد. این معیار به‌طور خاص برای اندازه‌گیری همبستگی بین داده‌های مختلف طراحی شده است. به‌طور دقیق‌تر، برای داده‌های مرکزیت‌یافته (میانگین صفر در هر ستون)  $X$  و  $Y$ ، رابطه زیر برقرار است:

$$\frac{1}{(n-1)^2} \text{tr}(XX^T YY^T) = \|\text{cov}(X^T, Y^T)\|_F^2 \quad (۹-۴)$$

معیار HSIC این معادله را با استفاده از فضایی خاص تعمیم می‌دهد و امکان بررسی مؤثر وابستگی‌ها را فراهم می‌کند. به بیان دیگر، این معیار با بهره‌گیری از توابع هسته‌ای، همبستگی بین ماتریس‌های داده را اندازه‌گیری می‌کند. به این نحو که عناصر  $K_{ij}$  و  $L_{ij}$  به ترتیب از طریق  $k(x_i, x_j)$  و  $l(y_i, y_j)$  محاسبه می‌شوند، که در این جا  $k$  و  $l$  توابع هسته‌ای هستند [۴۵].

برآورد HSIC به‌صورت زیر تعریف می‌شود:

$$\text{HSIC}(K, L) = \frac{1}{(n-1)^2} \text{tr}(KHLH) \quad (۱۰-۴)$$

که در آن  $H$  ماتریس مرکزیت‌دهنده است و به شکل  $H_n = I_n - \frac{1}{n}11^T$  تعریف می‌شود.

نکته جالب این است که اگر هسته‌های خطی  $k$  و  $l$  به‌صورت  $k(x, y) = l(x, y) = x^T y$  باشند، HSIC به همان معادله اولیه برمی‌گردد. این بدین معناست که معیار HSIC به‌طور دقیق و قابل اعتماد، میزان وابستگی و شباهت بین داده‌ها را ارزیابی کرده و به درک بهتر ساختارهای پیچیده داده‌ها کمک می‌کند.

<sup>۱</sup> Hilbert-Schmidt Independence Criterion

#### ۴-۷-۴ هم‌ترازی هسته مرکزی<sup>۱</sup> (CKA)

معیار HSIC در اندازه‌گیری همبستگی‌ها با مشکل عدم پایداری نسبت به مقیاس‌بندی یکسان ویژگی‌ها مواجه است. این بدان معناست که در صورت تغییر مقیاس ویژگی‌ها، نتیجه HSIC ممکن است دچار تغییراتی شود که به درستی شباهت‌های بین داده‌ها را نشان ندهد. برای رفع این مورد، از فرم نرمال‌شده‌ای به نام هم‌ترازی هسته مرکزی استفاده می‌شود.

در حقیقت CKA یک شاخص نرمال شده است که تأثیر مقیاس‌بندی یکسان را حذف می‌کند و به این ترتیب، دقت و پایداری بیشتری در مقایسه بازنمایی‌های شبکه‌های عصبی فراهم می‌آورد [۴۶، ۴۷]. رابطه CKA به شکل زیر تعریف می‌شود:

$$CKA(K, L) = \frac{HSIC(K, L)}{\sqrt{HSIC(K, K) \cdot HSIC(L, L)}} \quad (۱۱-۴)$$

که در آن عبارت  $HSIC(K, L)$  میزان همبستگی بین دو ماتریس هسته‌ای  $K$  و  $L$  را اندازه‌گیری می‌کند. صورت کسر، همان HSIC اصلی است که میزان همبستگی بین دو ماتریس داده را نشان می‌دهد. اما برای نرمال‌سازی این مقدار و حذف تأثیر مقیاس‌بندی، مخرج کسر به کار می‌رود که شامل ضرب دو HSIC مربوط به هر یک از ماتریس‌ها با خودشان است. این نرمال‌سازی باعث می‌شود که نتیجه نهایی مستقل از مقیاس‌بندی ویژگی‌ها باشد و شباهت‌های واقعی بین داده‌ها را بهتر منعکس کند.

استفاده از CKA در مقایسه با HSIC، به‌ویژه در تحلیل بازنمایی‌های شبکه‌های عصبی و داده‌های پیچیده، کارایی بهتری دارد. زیرا این شاخص نرمال‌شده، نه تنها شباهت‌های بین داده‌ها را دقیق‌تر می‌سنجد، بلکه در برابر تغییرات مقیاس‌بندی نیز مقاوم است. به این ترتیب، می‌توان از CKA به عنوان یک ابزار قدرتمند برای درک و تحلیل بازنمایی‌های مختلف در یادگیری عمیق و دیگر زمینه‌های مرتبط استفاده کرد.

#### ۴-۷-۵ هم‌ترازی هسته مرکزی بدون مداخله<sup>۲</sup> (dCKA)

فرض می‌شود  $X$  یک مجموعه داده ورودی با  $n$  نمونه و  $p$  ویژگی باشد. نمایش لایه‌های  $m_1$  و  $m_2$  از دو شبکه عصبی که به ترتیب  $f_1$  و  $f_2$  نامیده می‌شوند، به صورت  $X_{f_1}^{m_1}$  و  $X_{f_2}^{m_2}$  هستند. شبکه‌های عصبی  $f_1$  و  $f_2$  دو مدل متفاوت یادگیری ماشین با معماری و ساختار مخصوص به خود هستند. لایه‌های  $m_1$  و  $m_2$  نیز لایه‌های خاصی در این شبکه‌ها هستند که مورد بررسی قرار می‌گیرند. برای مثال،  $m_1$  می‌تواند لایه دوم از شبکه  $f_1$  و  $m_2$  می‌تواند لایه چهارم از شبکه  $f_2$  باشد [۴۸]. تمامی رابطه‌های مورد استفاده در بخش ۴-۷-۵، از مرجع [۴۸] برگرفته شده‌اند.

<sup>۱</sup>Centered Kernel Alignment

<sup>۲</sup>Deconfounded Centered Kernel Alignment

برای مقایسه نمایش‌های دو شبکه عصبی، ساختارهای شباهت در هر شبکه بررسی می‌شود. این کار با محاسبه شباهت بین هر جفت از نمونه‌ها در  $X_{f_1}^{m_1}$  و  $X_{f_2}^{m_2}$  با استفاده از یک معیار شباهت  $k(\cdot, \cdot)$  انجام می‌شود:

$$K_{f_1}^{m_1} = k(X_{f_1}^{m_1}, X_{f_1}^{m_1}), \quad K_{f_2}^{m_2} = k(X_{f_2}^{m_2}, X_{f_2}^{m_2}) \quad (۱۲-۴)$$

در این رابطه ماتریس‌های  $K_{f_1}^{m_1}$  و  $K_{f_2}^{m_2}$  نشان‌دهنده شباهت بین هر جفت از نمونه‌ها در لایه‌های  $m_1$  و  $m_2$  از شبکه‌های  $f_1$  و  $f_2$  هستند. در مرحله بعد، برای مقایسه این دو ماتریس از معیاری به نام  $s(\cdot, \cdot)$  استفاده می‌شود:

$$s_{f_1, f_2}^{m_1, m_2} = s(K_{f_1}^{m_1}, K_{f_2}^{m_2}) \quad (۱۳-۴)$$

این مقدار بیانگر میزان شباهت بین دو نمایش شبکه‌های عصبی است [۴۸]. این روش امکان درک دقیقی از شباهت‌ها و تفاوت‌های بین دو شبکه عصبی را فراهم می‌کند و می‌توان از آن برای بهبود عملکرد مدل‌های یادگیری ماشین بهره برد.

روش‌های فعلی برای مقایسه نمایش‌های شبکه‌های عصبی از معیارهای شباهت متفاوتی در دو مرحله استفاده می‌کنند. برای مثال روش CKA در مرحله اول از یک تابع هسته‌ای برای اندازه‌گیری شباهت استفاده می‌کند که با  $k(\cdot, \cdot)$  نشان داده می‌شود. در مرحله دوم، برای اندازه‌گیری شباهت بین تابع‌های هسته‌ای از معیاری به نام HSIC که با  $s(\cdot, \cdot)$  نشان داده می‌شود، بهره می‌برد.

حال به بررسی تأثیر متغیرهای مداخله‌گر<sup>۱</sup> در ارزیابی شباهت پرداخته می‌شود. فرض کنید مجموعه داده‌ای به نام  $X$  در اختیار است که شباهت بین نمونه‌ها در آن با ماتریسی به نام  $K^0$  نمایش داده می‌شود. این ماتریس شباهت، شباهت‌های اولیه بین داده‌ها در فضای ورودی را نشان می‌دهد. حالا وقتی پیش‌بینی با استفاده از شبکه‌های عصبی انجام می‌شود، می‌توان این پیش‌بینی‌ها را به صورت مرحله به مرحله در نظر گرفت.

در روش CKA، شباهت بین دو ماتریس  $K_{f_1}^{m_1}$  و  $K_{f_2}^{m_2}$ ، میزان شباهت بین دو شبکه عصبی را نشان می‌دهد. اما هر دو ماتریس  $K_{f_1}^{m_1}$  و  $K_{f_2}^{m_2}$  تحت تأثیر ماتریس شباهت اولیه  $K^0$  قرار می‌گیرند. این مسئله ممکن است باعث شود که شباهت بین شبکه‌ها بیش از حد بالا برود و واقعی نباشد. به طور کلی، داده‌های مشابه در فضای ورودی احتمالاً در لایه‌های ابتدایی شبکه‌های عصبی نیز مشابه خواهند بود، حتی اگر عملکرد شبکه‌های عصبی متفاوت باشد. بنابراین، CKA ممکن است تحت تأثیر ویژگی‌های خاص مجموعه داده قرار بگیرد و نتایج ناسازگاری بین مجموعه داده‌های مختلف ایجاد کند.

برای حل این مشکل، شباهت ورودی  $K^0$  از ماتریس‌های  $K_{f_1}^m$  و  $K_{f_2}^m$  حذف می‌شود. این فرآیند، حذف مداخله‌گر نامیده می‌شود. با این روش، تنها عملکرد شبکه عصبی علت شباهت‌های موجود در فضای پنهان

<sup>۱</sup> Confounded

خواهد بود و این شباهت‌ها ناشی از شباهت اولیه داده‌ها نخواهد بود. به این ترتیب، معیار بدون مداخله به مقایسه عملکرد واقعی شبکه‌های عصبی می‌پردازد و کمتر تحت تأثیر ساختار اولیه مجموعه داده قرار می‌گیرد [۴۸]. جهت رفع تأثیر متغیر مداخله‌گر و تنظیم شباهت‌های نادرست ناشی از آن، از روشی استفاده می‌شود که در آن ساختار شباهت ورودی از ساختار شباهت بازنمایی حذف می‌گردد. این روش به شکل زیر تعریف شده است:

$$dK_{f_1}^{m_1} = K_{f_1}^{m_1} - \hat{\alpha}_{f_1}^{m_1} K^0, \quad dK_{f_2}^{m_2} = K_{f_2}^{m_2} - \hat{\alpha}_{f_2}^{m_2} K^0 \quad (۱۴-۴)$$

در این جا،  $K^0$  نمایانگر ساختار شباهت ورودی است و  $\hat{\alpha}_{f_1}^{m_1}$  و  $\hat{\alpha}_{f_2}^{m_2}$  ضرایبی هستند که برای حداقل کردن اندازه ماتریس‌های  $dK_{f_1}^{m_1}$  و  $dK_{f_2}^{m_2}$  در نظر گرفته می‌شوند. حرف  $d$  قبل از ماتریس شباهت، نشان می‌دهد که این ماتریس بدون تأثیر متغیر مداخله‌گر است [۴۸].

ساختار شباهت ورودی  $K^0$  تأثیری خطی و قابل جمع شدن بر  $K_f^m$  دارد:

$$\text{vec}(K_f^m) = \alpha_f^m \text{vec}(K^0) + \epsilon_f^m \quad (۱۵-۴)$$

که در این جا تابع  $\text{vec}(\cdot)$  یک ماتریس را به یک بردار تبدیل می‌کند. نویز  $\epsilon_f^m$  نیز مستقل از متغیر مداخله‌گر فرض شده است و معادلات زیر را شامل می‌شود:

$$\hat{\epsilon}_f^m = \text{vec}(dK_f^m) \quad (۱۶-۴)$$

$$\hat{\alpha}_f^m = (\text{vec}(K^0)^T \text{vec}(K^0))^{-1} \text{vec}(K^0)^T \text{vec}(K_f^m) \quad (۱۷-۴)$$

در این جا برای حذف تأثیر یک ماتریس از روی دیگری، از معکوس آن ماتریس استفاده می‌شود. معکوس ماتریس  $A$ ، ماتریسی است که وقتی در  $A$  ضرب شود، نتیجه یک ماتریس همانی<sup>۱</sup> است. در این مرحله، معکوس ماتریس  $(\text{vec}(K^0)^T \text{vec}(K^0))$  برای حذف تأثیرات اولیه استفاده می‌شود. سپس، ضرب داخلی  $\text{vec}(K^0)$  با  $\text{vec}(K_f^m)$  انجام می‌گیرد تا شباهت‌های واقعی بین داده‌ها استخراج شوند. در نهایت، معکوس ماتریس مرحله قبل در این عبارت ضرب می‌شود تا تأثیرات مزاحم حذف شده و شباهت‌های خالص نمایان گردند.

پس از به دست آوردن ساختارهای شباهت، بدون تأثیر متغیر مداخله‌گر، از همان معیار شباهت در رابطه

(۱۳-۴) استفاده می‌شود:

$$ds_{f_1, f_2}^{m_1, m_2} = s(dK_{f_1}^{m_1}, dK_{f_2}^{m_2}) \quad (۱۸-۴)$$

این روش امکان بررسی دقیق و معتبر شباهت‌های بین ساختارهای بازنمایی را فراهم می‌کند، بدون این که نتایج تحت تأثیر متغیرهای مداخله‌گر قرار بگیرند.

<sup>۱</sup> Identity Matrix

#### ۴-۸ شاخص شباهت بین شبکه‌های عصبی

برای بررسی و مقایسه کامل دو شبکه عصبی، ارزیابی جداگانه تمامی لایه‌های آن‌ها و ارائه یک شاخص شباهت کلی ضروری است. این کار با استفاده از معیارهای شباهت برای هر لایه انجام شده و سپس نتایج این معیارها به صورت میانگین ترکیب می‌شوند تا یک نمای کلی از شباهت بین دو شبکه عصبی به دست آید.

در لایه‌های کاملاً متصل<sup>۱</sup> که تمامی نرون‌ها به هم متصل هستند، ساختار لایه‌ها به صورت ماتریس‌های دو بعدی می‌باشند. بنابراین مقایسه آن‌ها با استفاده از معیارهای شباهت به راحتی امکان‌پذیر است. اما در لایه‌های پیچشی<sup>۲</sup> که دارای ساختار چند بعدی هستند، ابتدا باید ابعاد این لایه‌ها به دو بعد تبدیل شوند تا امکان مقایسه فراهم شود. این تبدیل معمولاً با مسطح کردن<sup>۳</sup> لایه‌ها انجام می‌گیرد. پس از تبدیل، از معیارهای مشابه لایه‌های کاملاً متصل استفاده می‌شود.

برای مثال، دو شبکه عصبی فرض می‌شود که هر کدام شامل چندین لایه مختلف هستند. ابتدا برای هر لایه از شبکه اول، لایه متناظر در شبکه دوم پیدا می‌شود. سپس با استفاده از معیارهای شباهت، میزان شباهت بین این دو لایه محاسبه می‌شود و این فرآیند برای تمامی لایه‌ها تکرار می‌شود.

پس از محاسبه معیارهای شباهت برای تمامی لایه‌ها، این معیارها به صورت میانگین ترکیب می‌شوند تا یک شاخص کلی از شباهت بین دو شبکه عصبی به دست آید. این شاخص میانگین می‌تواند نشان دهد که دو شبکه چقدر به یکدیگر شبیه هستند.

این روش به درک بهتر از عملکرد و ساختار داخلی شبکه‌های عصبی کمک می‌کند. با داشتن این شاخص شباهت، شناسایی تفاوت‌ها و شباهت‌های دو شبکه آسان‌تر شده و می‌توان بر اساس آن‌ها تصمیمات بهتری برای بهبود مدل‌های یادگیری گرفت.

#### ۴-۹ بررسی تأثیرات جابه‌جایی مدل‌ها

در این بخش، تأثیرات مختلف جابه‌جایی مدل‌ها بر ترافیک شبکه و حریم شخصی بررسی شده است. ابتدا نشان داده می‌شود که روش‌های جابه‌جایی مدل‌ها هزینه ترافیک شبکه را نسبت به روش‌های سنتی افزایش نمی‌دهند و در برخی موارد حتی به کاهش آن کمک می‌کنند. سپس به بررسی تأثیرات این جابه‌جایی‌ها بر حریم شخصی کاربران پرداخته می‌شود که در آن خطرات مرتبط با تبادل مدل‌ها بین کاربران، با واسطه یا بدون واسطه سرور ارزیابی می‌شود.

<sup>1</sup>Fully connected

<sup>2</sup>Convolutional Layers

<sup>3</sup>Flattening

#### ۴-۹-۱ تاثیرات جابه‌جایی مدل‌ها بر ترافیک شبکه

در این بخش نشان داده می‌شود که جابه‌جایی مدل‌ها، هیچ هزینه اضافی در ترافیک شبکه نسبت به روش مرسوم FedAvg به همراه ندارد. به بیان دقیق‌تر، این روش نه تنها هزینه‌ای بیشتر ایجاد نمی‌کند، بلکه در بهبود ترافیک شبکه نیز مؤثر است و می‌تواند به سادگی در شرایط مختلف اعمال شود.

فرض می‌شود که تعداد کل کاربران با  $K$ ، اندازه مدل با  $s$ ، تعداد گام‌های بین هر جابه‌جایی مدل با  $h_1$  و تعداد جابه‌جایی‌ها بین هر میانگین‌گیری با  $h_2$  نشان داده شود. در روش FedAvg، در هر یک از گام‌های  $h_1 \times h_2$ ، ابتدا همه کاربران مدل‌های خود را به سرور ارسال می‌کنند. سپس سرور پس از انجام میانگین‌گیری، مدل‌های جدید را به کاربران بازمی‌گرداند. هزینه این فرآیند رفت و برگشت در هر گام برابر با  $2Ks$  است. در مقابل، روش FedSwap به گونه‌ای عمل می‌کند که در  $h_2 - 1$  مرحله اولیه، کاربران فقط مدل‌های خود را با یکدیگر مبادله می‌کنند و هیچ مدلی به سمت سرور ارسال نمی‌شود. این مبادله میان کاربران، هزینه‌ای معادل  $Ks$  دارد و تنها در گام نهایی، میانگین‌گیری انجام می‌شود.

در روش SimFedSwap، در تمام مراحل جابجایی که شامل  $h_2 - 1$  مرحله است، تمامی مدل‌ها به سرور ارسال می‌شوند. سپس سرور پس از بررسی شباهت مدل‌ها، آن‌ها را به کاربران بازمی‌گرداند. در نهایت، میانگین‌گیری در آخرین مرحله صورت می‌پذیرد. در این روش، در هر گام، رفت و برگشت کامل مدل‌ها انجام می‌شود که هزینه آن در تمامی مراحل برابر با  $2Ks$  خواهد بود.

بنابراین، در هر چرخه  $h_1 \times h_2$ ، هزینه ارتباطات برای روش‌های مختلف به صورت زیر به دست می‌آید:

$$\begin{aligned} FedAvg : & \quad h_1 h_2 (2Ks) \\ FedSwap : & \quad (h_2 - 1)(Ks) + (2Ks) \\ SimFedSwap : & \quad h_2 (2Ks) \end{aligned} \quad (۱۹-۴)$$

بر این اساس، میزان کاهش هزینه‌های ارتباطی در روش‌های FedSwap و SimFedSwap در مقایسه با روش

FedAvg به صورت زیر محاسبه می‌شود:

$$\begin{aligned} \frac{FedAvg - FedSwap}{FedAvg} &= \frac{h_1 h_2 (2Ks) - ((h_2 - 1)(Ks) + (2Ks))}{h_1 h_2 (2Ks)} \\ &= \frac{2h_1 h_2 Ks - h_2 Ks + Ks - 2Ks}{2h_1 h_2 Ks} \\ &= \frac{Ks(2h_1 h_2 - h_2 + 1 - 2)}{2h_1 h_2 Ks} \\ &= \frac{2h_1 h_2 - h_2 - 1}{2h_1 h_2} \end{aligned} \quad (۲۰-۴)$$



$$\begin{aligned}
\frac{FedAvg - SimFedSwap}{FedAvg} &= \frac{h_1 h_2 (2Ks) - (h_2 (2Ks))}{h_1 h_2 (2Ks)} \\
&= \frac{2h_1 h_2 Ks - 2h_2 Ks}{2h_1 h_2 Ks} \\
&= \frac{2h_2 Ks (h_1 - 1)}{2h_1 h_2 Ks} \\
&= \frac{h_1 - 1}{h_1}
\end{aligned} \tag{۲۱-۴}$$

جهت فهم بهتر این ساختار و چگونگی تأثیر آن بر ترافیک شبکه، با در نظر گرفتن این که در این مثال پارامتر  $h_1$  مقدار ۵ و پارامتر  $h_2$  مقدار ۳ را دارند، به شکل ۴-۲ دقت کنید. به طور خاص، در شبیه‌سازی‌هایی که پارامترهای  $h_1$  برابر با ۵ و  $h_2$  برابر با ۳ در نظر گرفته شده‌اند، روش FedSwap موفق به کاهش ۸۶/۶۶ درصدی هزینه‌های شبکه شده است. همچنین، روش SimFedSwap توانسته این هزینه‌ها را تا ۸۰ درصد کاهش دهد.

#### ۴-۹-۲ تأثیر جابجایی مدل‌ها بر حریم شخصی در ارتباط بین سرور و کاربران

در روش یادگیری فدرال سنتی، مانند FedAvg، مدل‌ها پس از آموزش محلی توسط کاربران، به سرور مرکزی ارسال می‌شوند. سرور وظیفه تجميع این مدل‌ها را بر عهده دارد و سپس مدل به‌روزرسانی شده را به کاربران بازمی‌گرداند. در این روش، تعاملات فقط بین سرور و کاربران صورت می‌گیرد و هیچ ارتباط مستقیمی بین کاربران وجود ندارد. این امر باعث می‌شود که حریم شخصی کاربران به دلیل عدم تبادل مستقیم اطلاعات با یکدیگر، تا حد زیادی حفظ شود. همچنین کاربران نیازی به اعتماد به یکدیگر ندارند و تنها باید به سرور مرکزی اعتماد کنند.

در مقابل، در روش‌هایی که مدل‌ها بین خود کاربران جابه‌جا می‌شود، چه با واسطه سرور و چه بدون واسطه آن، خطرات بیشتری برای حریم شخصی وجود دارد. زمانی که مدل‌ها مستقیماً بین کاربران مبادله می‌شود، احتمال

images/chap4/compare\_swap\_net\_traffic.png

شکل ۴-۲: تأثیر نحوه جابه‌جایی مدل‌ها بر ترافیک شبکه.

این که یکی از کاربران بتواند از طریق تحلیل مدل دریافت شده اطلاعاتی در مورد داده‌های دیگر کاربران به دست آورد، افزایش می‌یابد. حتی اگر سرور به عنوان واسطه در این جابجایی‌ها عمل کند، همچنان خطراتی وجود خواهد داشت، زیرا سرور می‌تواند نقش ناظر را داشته باشد و از جابجایی مدل‌ها بین کاربران سوء استفاده کند. بنابراین، در هر دو حالت، جابجایی مدل‌ها بین کاربران نسبت به FedAvg با چالش‌های بیشتری در زمینه حفظ حریم شخصی مواجه است.

در نتیجه، در روش FedAvg، به دلیل عدم ارتباط مستقیم بین کاربران، حریم شخصی به شکل بهتری حفظ می‌شود و تنها سرور مرکزی باید ایمن باشد. اما در روش جابجایی مدل‌ها بین کاربران، خطر نشت اطلاعات بین کاربران افزایش می‌یابد و این روش به پروتکل‌های امنیتی پیچیده‌تر و اعتماد بیشتر بین کاربران نیاز دارد.

#### ۴-۹-۳ تاثیر جابجایی مدل‌ها بر حریم شخصی با توجه به واسطه بودن سرور

روش حریم خصوصی تفاضلی که در ۲-۳-۴ نیز به آن اشاره شد یکی از تکنیک‌های موثر برای حفظ حریم شخصی است که با اضافه کردن نویز به داده‌ها یا مدل‌ها، مانع از افشای اطلاعات حساس فردی می‌شود. این روش تضمین می‌کند که خروجی یک الگوریتم یادگیری به اندازه کافی تصادفی است، به‌طوری‌که حضور یا عدم حضور یک نمونه داده خاص در مجموعه داده‌ها، تاثیری بر خروجی نهایی نداشته باشد.

حال اگر از این روش در جابجایی مدل‌ها بین کاربران استفاده شود، تفاوت‌هایی بین دو رویکرد با واسطه سرور و بدون واسطه سرور وجود خواهد داشت. در حالتی که سرور واسطه باشد، اعمال حریم خصوصی تفاضلی می‌تواند به کنترل و مدیریت نویز اضافه شده کمک کند و سرور می‌تواند نقش فعالی در تضمین این که مدل‌ها به درستی ناشناس شده‌اند، ایفا کند. این امر باعث می‌شود که خطر نشت اطلاعات به دلیل واسطه‌گری سرور کاهش یابد. اما همچنان باید به سرور اعتماد کرد که این فرآیند را به درستی انجام دهد.

در رویکرد بدون واسطه سرور، اعمال حریم خصوصی تفاضلی چالش‌برانگیزتر است، زیرا کاربران به‌طور مستقل باید نویز لازم را به مدل‌های خود اضافه کنند و در عین حال مطمئن شوند که سطح مناسبی از حفظ حریم شخصی برقرار است. در این حالت، هرگونه خطا یا ناهماهنگی در اعمال حریم خصوصی تفاضلی ممکن است منجر به افشای اطلاعات حساس شود. همچنین، نبود یک ناظر مرکزی مانند سرور، کار را برای هماهنگی و اجرای صحیح این روش پیچیده‌تر می‌کند.

در نتیجه، در صورت استفاده از روش حریم خصوصی تفاضلی، رویکرد با واسطه سرور به دلیل نظارت و کنترل متمرکز، از لحاظ حفظ حریم شخصی مزیت بیشتری دارد. در مقابل، در رویکرد بدون واسطه سرور، چالش‌های بیشتری برای کاربران وجود دارد که می‌تواند منجر به کاهش اثربخشی حریم خصوصی تفاضلی شود. در نهایت، اعتماد به سرور و هماهنگی دقیق بین کاربران نقش کلیدی در تعیین سطح امنیت و حریم شخصی

خواهد داشت.

#### ۴-۱۰ نحوه تعیین کاربران نهایی جهت جابه‌جایی مدل‌ها در روش SimFedSwap

زمانی که همه مدل‌های شبکه عصبی در سرور مرکزی قرار دارند، وظیفه سرور این است که تصمیم بگیرد کدام کاربران مدل‌های خود را با یکدیگر جابه‌جا کنند. برای انجام این کار، ابتدا باید مدل‌ها با یکدیگر مقایسه شوند تا میزان شباهت بین آن‌ها مشخص شود. سپس، بر اساس این شباهت‌ها تعیین می‌شود که کدام کاربران مدل‌های شبکه عصبی خود را با یکدیگر مبادله کنند، یا به عبارت دیگر، سرور مشخص می‌کند کدام مدل به کدام کاربر ارسال شود.

نکته مهمی که باید مد نظر قرار داد این است که فرآیند بررسی شباهت بین مدل‌های شبکه عصبی ممکن است زمان‌بر باشد. بنابراین، برای تصمیم‌گیری سریع درباره جابه‌جایی مدل‌ها، باید از روش‌های مؤثری استفاده شود. در ادامه، دو روش برای تعیین کاربران نهایی جهت جابه‌جایی مدل‌ها معرفی می‌شود.

##### ۴-۱۰-۱ روش جابه‌جایی حریصانه<sup>۱</sup> (GS)

در روش حریصانه، از بین تمام کاربران موجود، یک کاربر به صورت تصادفی انتخاب می‌شود. سپس مدل شبکه عصبی این کاربر با مدل‌های تمامی کاربران دیگر مقایسه می‌شود تا میزان شباهت آن‌ها سنجیده شود. در این مرحله، کاربری که مدل شبکه عصبی او کمترین شباهت را با مدل کاربر انتخاب شده دارد، به عنوان کاربر مقصد برای جابه‌جایی مدل انتخاب می‌شود. پس از این انتخاب، سرور مدل‌های این دو کاربر را با یکدیگر جابه‌جا می‌کند و در نهایت این دو کاربر را از لیست انتخاب حذف خواهد کرد.

پس از انجام این جابه‌جایی، فرایند مشابهی برای کاربران باقی‌مانده تکرار می‌شود. ابتدا یک کاربر دیگر به صورت تصادفی انتخاب می‌شود و دقیقاً همان روند بالا برای آن تکرار خواهد شد. شبه کد کامل این روش در الگوریتم ۴-۲ ارائه شده است. علاوه بر این، نمادهای مختص به این الگوریتم در جدول ۴-۲ و همچنین تمامی نمادهای پایه در جدول ۲-۱ توضیح داده شده‌اند. هدف از این جداول، فراهم کردن درکی جامع از نحوه عملکرد و پیاده‌سازی الگوریتم می‌باشد.

در ادامه این مورد بررسی خواهد شد که تابع ModelSimilarity در الگوریتم ۴-۲ چند مرتبه اجرا می‌شود. برای ساده‌تر کردن موضوع و فهم بهتر آن، لیست  $LRS$  با  $n$  مقدار اولیه در نظر گرفته می‌شود (لیستی با  $n$  عضو) و همچنین مقدار  $NS$  برابر  $n/2$  لحاظ خواهد شد. دلیل این انتخاب این است که با انجام  $n/2$  جابه‌جایی، همه مدل‌ها یک بار جابه‌جا می‌شوند. سپس حلقه اصلی به تعداد  $NS$  تکرار می‌شود. در هر تکرار از این حلقه

<sup>۱</sup> Greedy Swapping

---

```

1 Function GreedySwapping():
2    $LRS = \text{copy of } U_t$ ;
3    $NS = (\text{length of } U_t // 2) * SP$ ;
4   for  $NS$  times do
5      $RandomIndex = \text{random integer between 0 and length of } LRS$ ;
6      $SCB = LRS[RandomIndex]$ ;
7     remove  $SCB$  from  $LRS$ ;
8     Initialize  $LstSimilarity$  as an empty list;
9     for each  $RC \in LRS$  do
10       $Sim = \text{ModelSimilarity}(w^{SCB}, w^{RC})$ ;
11      Append  $Sim$  to  $LstSimilarity$ ;
12    end
13     $MinSimilarityIndex = \text{index of the minimum value in } LstSimilarity$ ;
14     $SCD = LRS[MinSimilarityIndex]$ ;
15    remove  $SCD$  from  $LRS$ ;
16    Swap( $SCB, SCD$ );
17  end
18 end

```

---

جدول ۴-۲: نمادهای مختص الگوریتم جابه‌جایی حریصانه

متغیر	توضیحات
$LRS$ ( $LstRemainSwap$ )	لیست باقی‌مانده جابه‌جایی
$NS$ ( $NumSwaps$ )	تعداد جابه‌جایی‌ها
$SP$ ( $SwapPercentage$ )	ضریب کنترلی برای تعداد جابه‌جایی‌ها
$RandomIndex$	شاخص تصادفی
$SCB$ ( $SwapClientBase$ )	کاربر مبدا جابه‌جایی
$LstSimilarity$	لیست مشابهت
$RC$ ( $RemainClient$ )	کاربر باقی‌مانده
$Sim$	معیار مشابهت بین دو مدل شبکه عصبی
$SCD$ ( $SwapClientDest$ )	کاربر مقصد جابه‌جایی

یک عنصر تصادفی از  $LRS$  انتخاب و حذف خواهد شد. سپس برای هر عنصر باقی‌مانده در  $LRS$ ، تابع  $ModelSimilarity$  فراخوانی می‌شود.

تعداد تکرارهای حلقه داخلی که در آن تابع  $ModelSimilarity$  فراخوانی می‌شود وابسته به تعداد عناصر باقی‌مانده در  $LRS$  است. به‌طور دقیق‌تر، در اولین تکرار حلقه اصلی،  $LRS$  شامل  $n - 1$  عنصر است و در دومین تکرار،  $LRS$  شامل  $n - 3$  عنصر خواهد بود؛ زیرا در هر تکرار از حلقه اصلی، یک عنصر به‌صورت تصادفی و یک عنصر دیگر با کمترین شباهت حذف می‌شوند.

به این ترتیب، تعداد کل فراخوانی‌های تابع  $ModelSimilarity$  برابر است با مجموع تعداد عناصر باقی‌مانده در هر تکرار از حلقه اصلی:

$$\sum_{i=0}^{NS-1} (n - 1 - 2i) \quad (۲۲-۴)$$

که این مجموع برای  $NS = n/2$  به صورت زیر است:

$$\sum_{i=0}^{(n/2)-1} (n-1-2i) \quad (23-4)$$

این یک دنباله حسابی با مقدار اولیه  $a = n-1$  و قدر نسبت  $d = -2$  است و تعداد جملات آن برابر با  $NS$  است. مجموع این دنباله حسابی به صورت زیر محاسبه می شود:

$$S = NS \times \left( \frac{a+l}{2} \right) \quad (24-4)$$

که در آن  $l$  مقدار آخرین جمله است و به این شکل به دست می آید:

$$\begin{aligned} l &= n-1-2(NS-1) \\ &= n-1-2(n/2-1) \\ &= n-1-n+2 \\ &= 1 \end{aligned} \quad (25-4)$$

بنابراین مجموع نهایی به این صورت خواهد بود:

$$\begin{aligned} S &= (n/2) \times \left( \frac{(n-1)+1}{2} \right) \\ &= (n/2) \times \left( \frac{n}{2} \right) \\ &= \frac{n^2}{4} \end{aligned} \quad (26-4)$$

پس در نهایت تابع ModelSimilarity به تعداد  $\left\lfloor \frac{n^2}{4} \right\rfloor$  بار اجرا می شود.

#### ۴-۱۰-۲ مرتبه زمانی روش جابه جایی حریصانه

بر اساس فرضیات مطرح شده در بخش قبل، زمان اجرای الگوریتم جابه جایی حریصانه معادل  $O(kn^2)$  است. باید توجه داشت که هنگام محاسبه مرتبه زمانی، مدت زمان اجرای تابع ModelSimilarity برابر  $k$  در نظر گرفته شده است. این مقدار به معیار شباهتی که برای این تابع انتخاب می شود بستگی دارد و با توجه به نوع معیار، می تواند کاملاً متفاوت باشد.

همچنین با توجه به این که مقدار  $NS$  برابر با  $n/2$  فرض شده و هر بار اجرای حلقه اصلی شامل یک حذف از لیست  $LRS$  می شود که زمان اجرای آن  $O(n)$  است و با در نظر گرفتن این که حلقه داخلی نیز، طبق بررسی های انجام شده در بخش ۴-۱۰-۱، مرتبه زمانی  $O(n)$  دارد، می توان نتیجه گرفت که ترکیب این دو عامل موجب می شود که کل زمان اجرای الگوریتم برابر با  $O(kn^2)$  باشد. البته نکته مهم این است که محاسبات حلقه دوم می توانند به طور کامل به صورت موازی انجام شوند. به طور دقیق تر، عناصر موجود در لیست  $LstSimilarity$  به

یکدیگر وابستگی ندارند و می‌توان تمام آن‌ها را به‌طور همزمان محاسبه کرد. اگر سرور قابلیت اجرای موازی این عملیات را داشته باشد، محاسبه حلقه داخلی عملاً لحاظ نمی‌شود و مرتبه زمانی برابر  $O(kn)$  خواهد شد. در نهایت، میزان مصرف حافظه توسط لیست *LstSimilarity* در الگوریتم ۴-۲ بررسی می‌شود. همان‌طور که در بخش قبل توضیح داده شد، حلقه داخلی این الگوریتم در بیشترین حالت به تعداد  $n-1$  مرتبه اجرا می‌شود. بنابراین، از نظر حافظه، لیست *LstSimilarity* در مرتبه  $O(n)$  قرار دارد. با این حال، باید توجه داشت که برای یافتن کمترین مقدار در یک مجموعه، حافظه‌ای به میزان  $O(1)$  نیز کافی است. در این الگوریتم، استفاده از لیست *LstSimilarity* به منظور افزایش خوانایی و سادگی کد صورت گرفته است، اگرچه از لحاظ بهینه‌سازی حافظه می‌توان از روش‌های کم‌حافظه‌تری نیز استفاده کرد. به بیان دیگر، به‌جای نگهداری همه شباهت‌ها در یک لیست و سپس پیدا کردن کمترین مقدار، می‌توان به‌صورت مستقیم در همان حلقه داخلی کمترین شباهت را دنبال کرد و در هر تکرار، فقط مقدار کمینه فعلی را به‌روزرسانی کرد. این روش نیاز به حافظه کمتری دارد و با حافظه  $O(1)$  قابل انجام است.

با این حال، استفاده از لیست *LstSimilarity* در این جا به منظور ساده‌تر و قابل فهم‌تر کردن کد انجام شده است. این انتخاب به توسعه‌دهندگان امکان می‌دهد تا الگوریتم را بهتر درک کنند و روند مقایسه شباهت‌ها را به وضوح مشاهده کنند، با این شرط که بهینه‌سازی حافظه در اولویت نباشد.

#### ۴-۱۰-۳ روش جابه‌جایی حداقل شباهت<sup>۱</sup> (MSS)

در این روش، برای این که حداقل شباهت ممکن بین تمامی مدل‌های شبکه عصبی به دست آید، لازم است تمامی مدل‌ها با یکدیگر مقایسه شوند و دو مدلی که کمترین شباهت را دارند با هم جابه‌جا شوند. به عنوان مثال، اگر  $n$  مدل وجود داشته باشد، باید یک ماتریس  $n \times n$  برای بررسی میزان شباهت‌ها ایجاد شود. در این ماتریس، شباهت مدل ۱ با مدل ۲ برابر با شباهت مدل ۲ با مدل ۱ در نظر گرفته می‌شود. همچنین به دلیل این که مقایسه یک مدل با خودش بی‌معنی است، ماتریس نهایی به شکل یک ماتریس بالا مثلثی<sup>۲</sup> (بدون قطر اصلی) تبدیل می‌شود. به این صورت که تنها حدود نیمی از ماتریس، شامل شباهت‌های مورد نیاز برای مقایسه است. در نتیجه ماتریس مورد نظر به شکل زیر در خواهد آمد.

$$\begin{bmatrix} \infty & a^{12} & a^{13} & \dots & a^{1n} \\ \infty & \infty & a^{23} & \dots & a^{2n} \\ \infty & \infty & \infty & \dots & a^{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \infty & \infty & \infty & \dots & \infty \end{bmatrix} \quad (۴-۲۷)$$

ابتدا، تمامی شباهت‌ها بین مدل‌ها محاسبه می‌شود و سپس از ماتریس ایجاد شده، کمترین مقدار شباهت

<sup>۱</sup>Minimum Similarity Swapping

<sup>۲</sup>Upper Triangular Matrix

انتخاب می‌شود. شماره سطر و ستون متناظر با این مقدار نشان می‌دهد که این دو مدل باید با یکدیگر جابه‌جا شوند. پس از انجام این جابه‌جایی، تمامی مقدارهای مربوط به سطر و ستون متناظر با این دو مدل باید به بی‌نهایت تغییر داده شوند تا در مراحل بعدی مجدد انتخاب نشوند. همچنین باید دقت کرد که هر دو سطر و ستون مرتبط با این دو مدل باید به بی‌نهایت تغییر داده شوند.

برای درک بهتر، فرض کنید کمترین مقدار شباهت در سطر سوم و ستون ششم ماتریس قرار دارد. در این حالت، مدل‌های سوم و ششم باید با یکدیگر جابه‌جا شوند. پس از این جابه‌جایی، لازم است که تمامی مقادیر در سطرهای سوم و ششم و همچنین ستون‌های سوم و ششم به بی‌نهایت تغییر کنند تا این دو مدل دیگر مورد بررسی قرار نگیرند. به این ترتیب، در دور بعدی، کوچک‌ترین مقدار شباهت از ماتریس انتخاب می‌شود و چون مقادیر مربوط به مدل‌های سوم و ششم به بی‌نهایت تغییر کرده‌اند، دیگر در این مرحله حضور نخواهند داشت و انتخاب نمی‌شوند. شبه‌کد کامل این روش در الگوریتم ۴-۳ ارائه شده است. علاوه بر این، نمادهای مختص به این الگوریتم در جدول ۴-۳ توضیح داده شده‌اند.

در این الگوریتم، تعداد دفعات اجرای تابع  $\text{ModelSimilarity}$  به تعداد کل جابه‌جایی‌ها بین کاربران نهایی وابسته نیست. همان‌طور که در الگوریتم ۴-۳ مشاهده می‌شود، ماتریس شباهت تنها یک بار محاسبه خواهد شد. با توجه به ساختار ماتریس بالا مثلی در رابطه ۴-۲۷ و با فرض این که تعداد مدل‌های کاربران برابر  $n$  در نظر گرفته شود، مقدار  $L$  در الگوریتم نیز برابر  $n$  خواهد شد. در نتیجه، تعداد دفعات اجرای تابع  $\text{ModelSimilarity}$  در الگوریتم جابه‌جایی حداقل شباهت، برابر با  $\frac{n(n-1)}{2}$  خواهد بود.

#### ۴-۱۰-۴ مرتبه زمانی و نحوه پیاده‌سازی روش جابه‌جایی حداقل شباهت

بر اساس فرضیات مطرح‌شده در بخش قبلی، زمان اجرای الگوریتم جابه‌جایی حداقل شباهت برابر با  $O(kn^2 + n^3)$  است. همچنین، مطابق با آنچه در بخش ۴-۱۰-۲ بیان شد، مرتبه زمانی تابع  $\text{ModelSimilarity}$  برابر  $k$  در نظر گرفته شده و طبق فرضیات بخش قبل، این تابع به تعداد  $O(n^2)$  بار اجرا می‌شود. بنابراین، بخش اول محاسبات با مرتبه زمانی  $O(kn^2)$  مشخص می‌شود.

برای محاسبات بخش دوم، پیدا کردن عنصر کمینه در ماتریس شباهت، خود از مرتبه زمانی  $O(n^2)$  است. بنابراین، وقتی که حلقه اصلی  $n/2$  بار اجرا شود و هر بار نیاز به پیدا کردن عنصر کمینه در ماتریس شباهت داشته باشد، مرتبه زمانی بخش دوم الگوریتم برابر  $O(n/2) \times O(n^2)$  خواهد بود که به  $O(n^3)$  ساده‌سازی می‌شود. پس با ترکیب بخش اول و بخش دوم مرتبه زمانی کل الگوریتم برابر  $O(kn^2 + n^3)$  خواهد شد.

لازم به ذکر است که محاسبات هر یک از عناصر ماتریس شباهت، مستقل از یکدیگر هستند. اگر سرور

الگوریتم ۴-۳: جابه‌جایی حداقل شباهت (Minimum Similarity Swapping)

```

1 Function MinSimilaritySwapping():
2   Initialize SimArray;
3   L = length of Ut;
4   for each row from 0 to L do
5     for each col from (row + 1) to L do
6       Sim = ModelSimilarity(wrow, wcol);
7       Append Sim to SimArray;
8     end
9   end
10  NS = (L // 2) * SP;
11  for NS times do
12    MI = index of minimum value in SimArray;
13    row, col = find row and col number, based on MI;
14    Set all values of SimArray[row, col] to ∞;
15    Swap(wrow, wcol);
16  end
17 end

```

جدول ۴-۳: نمادهای مختص الگوریتم جابه‌جایی حداقل شباهت

متغیر	توضیحات
<i>SimArray</i>	آرایه شباهت
<i>L</i>	طول مجموعه‌ای از کاربران در گام <i>t</i>
<i>MI</i> ( <i>MinIndex</i> )	شاخص مقدار کمینه در آرایه شباهت
<i>row</i>	شماره سطر بر اساس دید ماتریس شباهت
<i>col</i>	شماره ستون بر اساس دید ماتریس شباهت

قابلیت اجرای موازی این عملیات را داشته باشد، مرتبه زمانی محاسبه بخش اول برابر  $O(k)$  و در نتیجه کل الگوریتم برابر با  $O(k + n^3)$  خواهد شد.

ماتریس شباهت، در حالت عادی، دارای  $n^2$  عنصر است. با بررسی دقیق‌تر الگوریتم و تعداد اجراهای حلقه دوم، در صورتی که از ماتریس شباهت در پیاده‌سازی استفاده شود، از نظر زمان اجرا رابطه زیر به دست می‌آید:

$$\left(\frac{n}{2}\right) \times (n^2 + 4n) = \left(\frac{n^3}{2}\right) + 2n^2 \quad (۴-۲۸)$$

$$\stackrel{\times 4}{=} 2n^3 + 8n^2$$

در این رابطه، تعداد اجراهای حلقه دوم برابر  $n/2$ ، پیدا کردن مقدار کمینه در ماتریس شباهت برابر با  $n^2$  و انتساب مقدار بی‌نهایت برای دو سطر و ستون ماتریس شباهت برابر با  $4n$  است.

با توجه به رابطه ۴-۲۷، بیش از نصف ماتریس، شامل مقادیر بی‌نهایت می‌باشد. بنابراین، با پیاده‌سازی ماتریس به صورت یک آرایه یک‌بعدی و تنها ذخیره‌سازی مقادیر بالا مثالی، می‌توان با انجام چند عملیات ساده ریاضی به مقدار سطر و ستون مورد نظر در ماتریس شباهت دست یافت.

در این پیاده‌سازی، آرایه یک‌بعدی جدید دارای  $\frac{n(n-1)}{2}$  عنصر خواهد بود. اگر حلقه دوم الگوریتم، مجدد



بررسی شود، زمان اجرای آن به صورت زیر خواهد بود:

$$\begin{aligned}
 \left(\frac{n}{2}\right) \times \left(\frac{n(n-1)}{2} + 4n\right) &= \left(\frac{n^2(n-1)}{4}\right) + 2n^2 \\
 &= \left(\frac{n^3}{4} - \frac{n^2}{4}\right) + 2n^2 \\
 &\stackrel{\times 4}{=} n^3 - n^2 + 8n^2 \\
 &= n^3 + 7n^2
 \end{aligned}
 \tag{۲۹-۴}$$

در این عبارت، تعداد اجراهای حلقه دوم برابر  $n/2$ ، پیدا کردن مقدار کمینه در آرایه برابر  $\frac{n(n-1)}{2}$  و در نهایت انتساب مقدار بی نهایت در آرایه مربوطه برابر با  $4n$  خواهد بود.

همان طور که مشاهده می شود، این پیاده سازی تقریباً سرعت اجرای الگوریتم را دو برابر می کند. اگرچه از نظر مرتبه زمانی بهبودی حاصل نشد، اما افزایش سرعت اجرا به میزان دو برابر، بهبود قابل توجهی در روند آموزش محسوب می شود. همچنین از نظر حافظه نیز بهبود حاصل شده است، زیرا در صورت استفاده از ماتریس مشابهت نیاز به ذخیره سازی  $n^2$  عنصر است، در حالی که با به کارگیری آرایه مشابهت تنها  $\frac{n(n-1)}{2}$  عنصر ذخیره خواهد شد. بنابراین، استفاده از ساختار آرایه یک بعدی می تواند بسیار مفید بوده و به کارایی الگوریتم کمک کند.

#### ۴-۱۱ جمع بندی

روش جابه جایی فدرال به جای ادغام مدل های محلی در هر مرحله از یادگیری فدرال، این مدل ها را بین دستگاه ها جابه جا می کند. این روش با هدف کاهش تأثیرات منفی ناشی از داده های non-IID و بهبود دقت مدل ها طراحی شده است. جابه جایی مدل ها به دستگاه های مختلف اجازه می دهد تا به داده های متنوع تری دسترسی پیدا کنند و عملکرد مدل سراسری بهبود یابد. همچنین، جابه جایی فدرال در دو حالت تصادفی و بر پایه شباهت معرفی شد که در حالت دوم، مدل هایی که کمترین شباهت را دارند جابه جا می شوند تا تنوع داده ها افزایش یابد و یادگیری بهینه تر صورت گیرد.

جابه جایی مدل ها در روش های FedSwap و SimFedSwap می تواند هزینه های ارتباطی را به میزان قابل توجهی کاهش دهد و از این نظر بر ترافیک شبکه تأثیر مثبتی داشته باشد. از سوی دیگر، این جابه جایی ها می توانند چالش های جدیدی را در زمینه حفظ حریم شخصی ایجاد کنند، به ویژه زمانی که مدل ها مستقیماً بین کاربران جابه جا می شوند. استفاده از روش های حریم خصوصی تفاضلی می تواند به کاهش این خطرات کمک کند، اما اجرای صحیح آن ها، به ویژه بدون واسطه گری سرور، چالش برانگیز است.

برای حفظ دقت در مقایسه های شبکه های عصبی در شرایط مختلف، لازم است شاخص های شباهت در برابر تغییرات مقاوم باشند. برای تحلیل عمیق تر، استفاده از روش های مختلفی مانند معیارهای نرمال شده یا همان

CKA پیشنهاد می‌شود. ارزیابی لایه‌های شبکه به صورت جداگانه و ترکیب نتایج آن‌ها به بهینه‌سازی شاخص انتخاب شده کمک می‌کند. در نهایت، در روش SimFedSwap، سرور دو مدل با کمترین شباهت را انتخاب و با یکدیگر جابه‌جا می‌کند که این فرآیند بر اساس روش حریصانه یا حداقل شباهت انجام می‌گیرد.

## فصل پنجم

### پیاده‌سازی و بررسی نتایج

#### ۵-۱ مقدمه

در این فصل به پیاده‌سازی شبکه‌های عصبی و تحلیل نتایج آن‌ها پرداخته می‌شود. ابتدا، دو مدل اصلی که در این پژوهش استفاده شده‌اند، معرفی و پیاده‌سازی می‌شوند. جزئیات هر مدل، شامل ابعاد لایه‌ها، توابع فعال‌سازی و ساختار کلی آن‌ها توضیح داده شده است. سپس مجموعه داده‌ها معرفی شده و نتایج اجرای مدل‌ها با روش SimFedSwap بررسی می‌گردد. در این بررسی، مقایسه‌هایی با دیگر روش‌های مرسوم از نظر تعداد تکرار و جابه‌جایی بر اساس روش حریصانه و حداقل شباهت انجام می‌شود.

#### ۵-۲ پیاده‌سازی مدل‌های شبکه عصبی

در این بخش، دو ساختار و مدل اصلی شبکه‌های عصبی، شامل شبکه عصبی پرسپترون چندلایه<sup>۱</sup> (MLP) و شبکه عصبی پیچشی<sup>۲</sup> (CNN) مورد بررسی قرار خواهند گرفت. این بررسی به منظور فراهم آوردن درکی جامع از نحوه عملکرد هر مدل، نقش آن‌ها در بررسی نتایج و مقایسه روش‌ها در پژوهش انجام خواهد شد. به عبارت دیگر استفاده از این مدل‌ها به تحلیل و ارزیابی دقیق‌تر نتایج کمک کرده و مقایسه‌ای جامع از روش‌های مختلف را

---

<sup>۱</sup>MultiLayer Perceptron

<sup>۲</sup>Convolutional Neural Network

ممکن می‌سازد.

### ۵-۲-۱ مدل MLP

در شبکه عصبی چندلایه، ابتدا لایه‌های شبکه عصبی در قالب یک ساختار ترتیبی<sup>۱</sup> ایجاد می‌شوند. این ساختار ترتیب‌دار باعث می‌شود که لایه‌ها به‌صورت متوالی اجرا شوند و خروجی هر لایه به عنوان ورودی به لایه بعدی منتقل شود.

اولین لایه، یک لایه کاملاً متصل است که تعداد نوروں‌های ورودی آن برابر با تعداد ویژگی‌های ورودی مدل به‌صورت مسطح‌شده و تعداد نوروں‌های خروجی آن ۲۵۶ است. این لایه تمام اتصالات ممکن بین نوروں‌های ورودی و خروجی را دارد. پس از این لایه، یک تابع فعال‌سازی ReLU قرار دارد که وظیفه آن این است که تمامی مقادیر منفی خروجی را به صفر تبدیل کند و مقادیر مثبت را بدون تغییر نگه دارد.

لایه دوم، یک لایه کاملاً متصل دیگر است که ۲۵۶ نوروں ورودی و ۱۲۸ نوروں خروجی دارد. پس از این لایه نیز یک تابع فعال‌سازی ReLU قرار دارد که مشابه تابع فعال‌سازی قبلی عمل می‌کند. سومین لایه نیز دقیقاً مشابه لایه دوم است با این تفاوت که ۱۲۸ نوروں به عنوان ورودی و ۶۴ نوروں به عنوان خروجی دارد و در ادامه آن هم تابع فعال‌سازی ReLU وجود دارد.

چهارمین و آخرین لایه، یک لایه کاملاً متصل است که ۶۴ نوروں ورودی و تعداد نوروں‌های خروجی آن برابر با تعداد کلاس‌های موجود در مسئله طبقه‌بندی است. این لایه، خروجی‌های نهایی شبکه را تولید می‌کند که نشان‌دهنده میزان تعلق هر ورودی به هر یک از کلاس‌ها است.

در نهایت، یک لایه Softmax اضافه شده است که وظیفه آن تبدیل خروجی‌های نهایی شبکه به توزیع احتمالاتی است. این لایه کمک می‌کند تا بتوان احتمال تعلق هر ورودی به هر کلاس را به‌صورت عددی بین صفر و یک به دست آورد که جمع کل این احتمالات برای همه کلاس‌ها برابر با یک خواهد بود. این توزیع احتمالاتی برای انجام پیش‌بینی‌های نهایی مورد استفاده قرار می‌گیرد. در پایان می‌توانید ساختار این مدل را در شکل ۵-۱ مشاهده نمایید.

### ۵-۲-۲ مدل CNN

در شبکه عصبی پیچشی، ابتدا یک بلوک ترتیبی شامل لایه‌های مختلف تعریف شده است. این لایه‌ها به ترتیب وظایف مختلفی در استخراج ویژگی‌ها و انجام طبقه‌بندی نهایی دارند. اولین لایه، یک لایه پیچشی تعریف شده است که تعداد کانال‌های ورودی تصویر را به ۳۲ کانال خروجی تبدیل می‌کند. این لایه از یک هسته پیچشی

<sup>۱</sup> Sequential

images/chap5/mlp.png

شکل ۵-۱: ساختار مدل MLP.

با اندازه  $3 \times 3$  استفاده می‌کند. این لایه وظیفه دارد تا ویژگی‌های ابتدایی تصویر ورودی را استخراج کند. پس از این لایه، یک تابع فعال‌سازی ReLU قرار دارد که مقادیر منفی را به صفر تبدیل کرده و مقادیر مثبت را بدون تغییر نگه می‌دارد که این کار باعث ایجاد غیرخطی شدن شبکه می‌شود.

سپس یک لایه تجمیع حداکثر<sup>۱</sup> قرار دارد که اندازه فضای ویژگی‌های خروجی را کاهش می‌دهد و به کم کردن تعداد پارامترها و افزایش کارایی مدل کمک می‌کند. این لایه با انتخاب حداکثر مقدار در هر ناحیه کوچک  $2 \times 2$ ، منجر به کاهش ابعاد تصاویر خواهد شد.

در ادامه، یک لایه پیچشی دیگر قرار دارد که تعداد کانال‌های خروجی را به ۶۴ کانال افزایش می‌دهد. این لایه نیز از یک هسته پیچشی با اندازه  $3 \times 3$  استفاده می‌کند و وظیفه استخراج ویژگی‌های پیچیده‌تر را بر عهده دارد. پس از این لایه نیز یک تابع فعال‌سازی ReLU قرار دارد که مشابه قبل عمل می‌کند.

سپس یک لایه تجمیع حداکثر دیگر قرار دارد که اندازه فضای ویژگی‌های خروجی را مجدداً کاهش می‌دهد. این لایه نیز با انتخاب حداکثر مقدار در هر ناحیه کوچک  $2 \times 2$ ، به کاهش ابعاد تصاویر کمک می‌کند.

پس از این لایه‌ها، یک لایه مسطح‌کننده قرار دارد که ویژگی‌های چند بعدی خروجی را به یک بردار یک بعدی تبدیل می‌کند. این کار برای آماده‌سازی داده‌ها جهت ورود به لایه‌های کاملاً متصل انجام می‌شود.

در مرحله بعد، یک لایه کاملاً متصل قرار دارد که بردار ویژگی‌ها را به یک بردار با ۱۰۰ نورون تبدیل می‌کند. این لایه تمام اتصالات ممکن بین نورون‌های ورودی و خروجی را دارد. پس از این لایه، یک تابع فعال‌سازی ReLU وجود دارد که مشابه توابع فعال‌سازی قبلی عمل می‌کند و غیرخطی بودن را به شبکه اضافه می‌کند.

<sup>۱</sup>Max Pooling

در پایان، یک لایه کاملاً متصل دیگر قرار دارد که بردار ویژگی‌ها را به یک بردار جدید با تعداد نوروتهایی برابر با تعداد کلاس‌ها تبدیل می‌کند. این لایه، خروجی نهایی شبکه را تولید می‌کند که نشان می‌دهد هر ورودی به چه میزان به هر یک از کلاس‌ها تعلق دارد.

در انتها، یک لایه Softmax قرار داده شده که خروجی‌های نهایی شبکه را به توزیع احتمالاتی تبدیل می‌کند. این لایه باعث می‌شود که احتمال تعلق هر ورودی به هر کلاس به صورت عددی بین صفر و یک محاسبه شود، به طوری که مجموع این احتمالات برای همه کلاس‌ها برابر با یک باشد. این توزیع احتمالاتی در انتها برای انجام پیش‌بینی‌های نهایی به کار می‌رود. در پایان جهت درک بهتر می‌توانید ساختار این مدل را در شکل ۵-۲ مشاهده نمایید.

### ۵-۳ مجموعه داده MNIST<sup>۱</sup>

مجموعه داده MNIST یکی از مشهورترین و پر استفاده‌ترین مجموعه داده‌ها در زمینه یادگیری ماشین است. این مجموعه شامل تصاویر دست‌نویس از اعداد ۰ تا ۹ می‌باشد و به‌طور گسترده‌ای برای آموزش و ارزیابی مدل‌های مختلف یادگیری ماشین به کار گرفته می‌شود. مجموعه داده MNIST در دهه ۱۹۹۰ توسط یان لکون<sup>۲</sup>، کورینا کورتس<sup>۳</sup> و کریستوفر برجس<sup>۴</sup> ایجاد شد [۱۲]. هدف اصلی این مجموعه داده، فراهم کردن یک مجموعه

images/chap5/cnn.png

شکل ۵-۲: ساختار مدل CNN.

<sup>۱</sup>Modified National Institute of Standards and Technology

<sup>۲</sup>Yann LeCun

<sup>۳</sup>Corinna Cortes

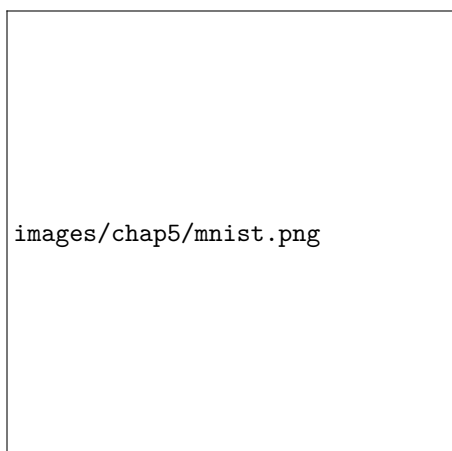
<sup>۴</sup>Christopher Burges

استاندارد برای ارزیابی الگوریتم‌های یادگیری ماشین و بینایی کامپیوتر بود.

مجموعه داده MNIST شامل ۷۰,۰۰۰ تصویر از ارقام دست‌نویس است که به دو بخش شامل مجموعه آموزش با ۶۰,۰۰۰ تصویر و مجموعه تست با ۱۰,۰۰۰ تصویر تقسیم می‌شود. هر تصویر دارای ابعاد  $28 \times 28$  پیکسل است که به‌صورت خاکستری<sup>۱</sup> ذخیره شده‌اند و هر پیکسل دارای مقداری بین ۰ (سیاه) تا ۲۵۵ (سفید) است. همچنین تمامی تصاویر با یک برچسب عددی بین ۰ تا ۹ همراه هستند که نمایانگر رقم موجود در تصویر می‌باشد. چند نمونه از اعضای این مجموعه داده در شکل ۵-۳، نمایش داده شده‌اند.

داده‌ها معمولاً در قالب دو فایل باینری شامل یکی برای تصاویر و دیگری برای برچسب‌ها ذخیره می‌شوند. هر تصویر به‌صورت یک بردار از اعداد بین ۰ تا ۲۵۵ با طول ۷۸۴ ( $28 \times 28$ ) ذخیره می‌شود. به دلیل یکنواختی تصاویر و اندازه کوچک آن‌ها، نیاز به پیش‌پردازش پیچیده‌ای ندارند. یکی از مراحل پیش‌پردازش شامل نرمال‌سازی یا همان تبدیل مقادیر پیکسل‌ها به مقادیر بین ۰ و ۱ می‌باشد.

مجموعه داده MNIST به عنوان یک نقطه شروع استاندارد برای آموزش و ارزیابی مدل‌های مختلف یادگیری عمیق و شبکه‌های عصبی استفاده می‌شود. محققان اغلب از MNIST برای مقایسه کارایی الگوریتم‌های جدید با الگوریتم‌های موجود استفاده می‌کنند. این مجموعه شامل نمونه‌های متنوعی از ارقام دست‌نویس از افراد مختلف است که موجب می‌شود به عنوان یک معیار استاندارد برای مقایسه مدل‌ها و الگوریتم‌ها مورد استفاده قرار گیرد. مجموعه داده MNIST دارای مزایای زیادی از جمله سادگی، در دسترس بودن، استاندارد بودن و پراکندگی داده‌ها می‌باشد. با این حال، این مجموعه داده دارای معایبی نیز هست. به عنوان مثال، برای مسائل پیچیده‌تر و واقعی‌تر ممکن است MNIST خیلی ساده باشد و نتواند چالش‌های واقعی را نشان دهد. همچنین، این مجموعه داده شامل تنها اعداد ۰ تا ۹ است و برای سایر کاربردهای دسته‌بندی تصویر ممکن است کافی نباشد.



شکل ۵-۳: چند نمونه از اعضای مجموعه داده MNIST [۴۹].

<sup>۱</sup> Grayscale

کاربردهای عملی این مجموعه داده شامل آموزش شبکه‌های عصبی متفاوت برای بهبود دقت دسته‌بندی، تست و ارزیابی مدل‌های مختلف یادگیری عمیق و الگوریتم‌های بهینه‌سازی است. بسیاری از مدل‌ها و الگوریتم‌های پیشرفته امروزی با استفاده از مجموعه داده MNIST توسعه و ارزیابی شده‌اند.

به‌طور کلی، مجموعه داده MNIST با توجه به دلایل ذکر شده، یکی از مهم‌ترین و پراستفاده‌ترین مجموعه داده‌ها در زمینه یادگیری ماشین و بینایی کامپیوتر است. این مجموعه به محققان و دانشجویان کمک می‌کند تا مفاهیم پایه‌ای یادگیری ماشین را به خوبی درک کرده و الگوریتم‌های جدید را ارزیابی کنند.

اکنون نتایج مربوط به مجموعه داده MNIST بررسی خواهد شد که شکل ۵-۴ نتایج مقایسه روش SimFedSwap با سایر روش‌های مرجع را با استفاده از مدل MLP به تصویر می‌کشد. همچنین، پارامترهای به‌کاررفته در این اجرا در جدول ۵-۱ به نمایش درآمده‌اند. نکته قابل توجه این است که منحنی‌های FedAvg و FedSwap به عنوان منحنی‌های نهایی و مرجع بر روی سایر منحنی‌ها قرار گرفته‌اند. در صورتی که رنگ متفاوتی در نمودار دیده شود، این موضوع نشان‌دهنده اختلاف عملکرد روش مربوطه در آن نقطه خواهد بود. این تغییر ممکن است نشان‌دهنده عملکرد بهتر یا ضعیف‌تر در مقایسه با دیگر روش‌ها باشد و می‌تواند به عنوان مبنایی برای مقایسه و تحلیل مورد توجه قرار گیرد.

همان‌طور که در شکل ۵-۴ مشاهده می‌شود، روش‌های مبتنی بر جابه‌جایی به شکل بسیار ناچیزی از روش FedAvg نتایج مطلوب‌تری را ارائه داده‌اند. نکته قابل توجه این است که همه روش‌های مبتنی بر جابه‌جایی عملکردی مشابه داشته‌اند. برای بررسی دقیق‌تر این اجرا، منحنی‌های خطا در شکل آ-۱ پیوست، قابل مشاهده هستند.

در شکل ۵-۵ همان آزمایش قبلی تکرار شده، با این تفاوت که این مرتبه از مدل شبکه عصبی CNN استفاده شده است. همان‌طور که دیده می‌شود، تقریباً تمامی روش‌ها عملکرد مشابهی داشته‌اند. این نکته نشان می‌دهد که وقتی شبکه به راحتی به دقت بالایی می‌رسد، تفاوتی در نتایج بین روش‌ها دیده نمی‌شود. برای جزئیات بیشتر این اجرا، منحنی‌های خطا در شکل آ-۲ پیوست، آمده‌اند.

#### ۴-۵ مجموعه داده CIFAR-10<sup>۱</sup>

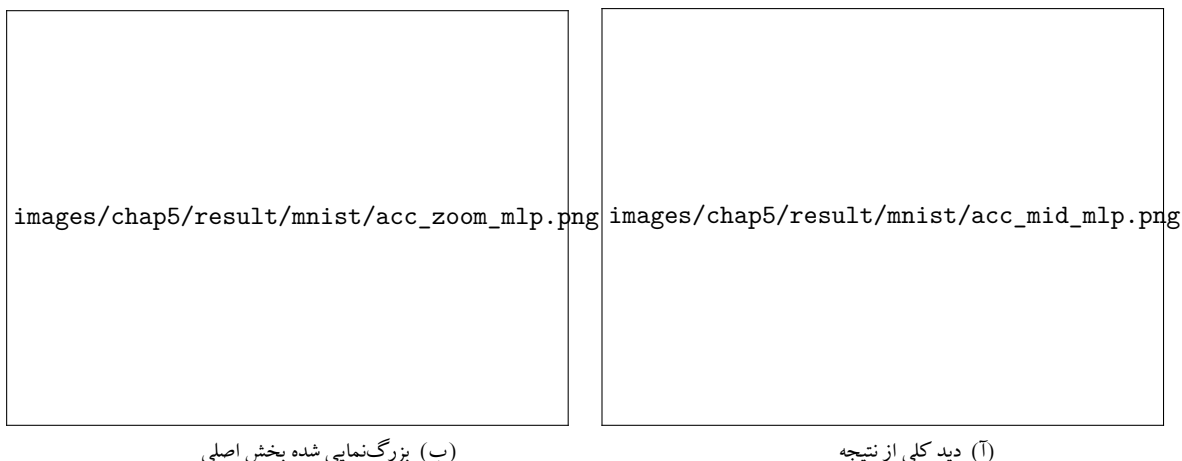
مجموعه داده CIFAR-10 یکی از معروف‌ترین و پرکاربردترین مجموعه داده‌های مورد استفاده در حوزه یادگیری ماشین و بینایی کامپیوتر است. این مجموعه داده توسط گروهی به سرپرستی الکس کریژفسکی<sup>۲</sup> و جفری هینتون<sup>۳</sup>

<sup>۱</sup>Canadian Institute For Advanced Research

<sup>۲</sup>Alex Krizhevsky

<sup>۳</sup>Geoffrey Hinton





(ب) بزرگ‌نمایی شده بخش اصلی

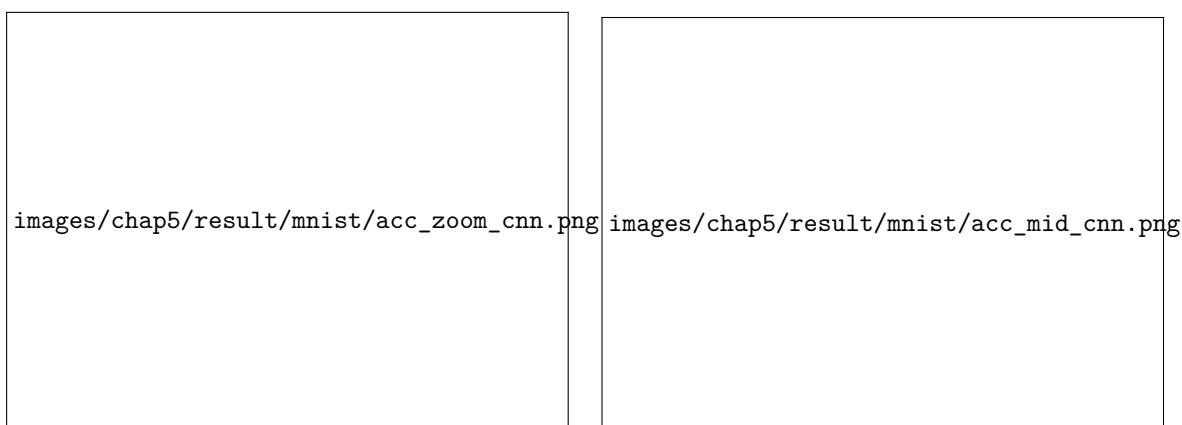
(T) دید کلی از نتیجه

شکل ۴-۵: مقایسه منحنی‌های دقت در مجموعه داده MNIST با استفاده از مدل MLP.

جدول ۵-۱: پارامترهای اجرا در مجموعه داده MNIST

مجموعه داده	نحوه جابه‌جایی	توزیع داده	$K$	$B$	$C$	$SP$	$\eta$	$E$	$h_1$	$h_2$
MNIST	MSS	نرمال	10	32	1.0	1.0	0.001	1	5	3

در دانشگاه تورنتو گردآوری شده و برای ارزیابی و آزمایش مدل‌های یادگیری عمیق به کار می‌رود [۵۰]. مجموعه داده CIFAR-10 شامل ۶۰,۰۰۰ تصویر رنگی با اندازه  $32 \times 32$  پیکسل است که به ۱۰ کلاس مختلف تقسیم شده‌اند. هر کلاس شامل ۶۰۰۰ تصویر است که به‌صورت مساوی بین مجموعه‌های آموزشی و آزمایشی توزیع شده‌اند. این کلاس‌ها شامل مواردی مانند هواپیما، اتومبیل، پرنده، گربه، گوزن، سگ، قورباغه، اسب، کشتی و کامیون هستند. هر یک از این کلاس‌ها دارای تصاویری است که تنوع بالایی از زوایا، پس‌زمینه‌ها و شرایط نوری مختلف را شامل می‌شود. در شکل ۵-۶، چند نمونه از هر کلاس در مجموعه داده CIFAR-10 به نمایش درآمده است.



(ب) بزرگ‌نمایی شده بخش اصلی

(T) دید کلی از نتیجه

شکل ۵-۵: مقایسه منحنی‌های دقت در مجموعه داده MNIST با استفاده از مدل CNN.

یکی از ویژگی‌های مهم مجموعه داده CIFAR-10 تنوع بالای تصاویر در هر کلاس است. این تنوع باعث می‌شود که مدل‌های یادگیری عمیق نیاز به توانایی تعمیم‌دهی بالا برای تشخیص صحیح کلاس‌ها داشته باشند. این مجموعه داده برای آموزش و ارزیابی مدل‌های مختلفی مورد استفاده قرار می‌گیرد و بسیاری از پژوهش‌ها و مقالات علمی از آن به عنوان مبنای مقایسه عملکرد مدل‌ها استفاده کرده‌اند.

مجموعه داده CIFAR-10 به دو بخش آموزشی و آزمایشی تقسیم شده است. بخش آموزشی شامل ۵۰,۰۰۰ تصویر و بخش آزمایشی شامل ۱۰,۰۰۰ تصویر است. این تقسیم‌بندی، استاندارد برای ارزیابی مدل‌ها فراهم می‌کند، به طوری که مدل‌ها می‌توانند بر روی مجموعه آموزشی، آموزش دیده و سپس بر روی مجموعه آزمایشی ارزیابی شوند. این روش به محققان امکان می‌دهد تا عملکرد مدل‌ها را به صورت عینی و قابل تکرار مقایسه کنند.

به دلیل اندازه کوچک تصاویر ( $32 \times 32$  پیکسل)، پردازش و آموزش مدل‌ها بر روی CIFAR-10 نسبتاً سریع و کم هزینه است. این ویژگی باعث شده تا مجموعه داده CIFAR-10 برای آزمایش مدل‌ها بسیار مناسب باشد. بسیاری از ابزارها و چارچوب‌های<sup>۱</sup> یادگیری ماشین مانند TensorFlow و PyTorch شامل توابع و ابزارهای آماده برای بارگذاری و استفاده از این مجموعه داده هستند که این امر نیز به سهولت استفاده از آن کمک می‌کند. در نهایت، مجموعه داده CIFAR-10 با ارائه تصاویری متنوع و چالش‌برانگیز در کلاس‌های مختلف، ابزاری

images/chap5/cifar10.png

شکل ۵-۶: چند نمونه از هر کلاس در مجموعه داده CIFAR-10 [۵۱].

<sup>۱</sup> Frameworks

قدرتمند برای آموزش و ارزیابی مدل‌های یادگیری عمیق فراهم می‌کند. این مجموعه داده نه تنها در پژوهش‌های دانشگاهی بلکه در صنعت نیز به عنوان معیاری برای ارزیابی پیشرفت‌ها در حوزه بینایی کامپیوتر استفاده می‌شود. اکنون نتایج مربوط به مجموعه داده CIFAR-10 بررسی خواهد شد که شکل ۵-۷ نتایج مقایسه روش SimFedSwap با سایر روش‌های مرجع را با توزیع داده یکنواخت بین کاربران به تصویر می‌کشد. پارامترهای استفاده شده در این آزمایش نیز در جدول ۵-۲ به نمایش درآمده‌اند.

همان‌طور که در شکل ۵-۷ مشاهده می‌شود، روش‌های مبتنی بر جابه‌جایی نسبت به روش FedAvg عملکرد متمایزی داشته‌اند. با این حال، این روش‌ها در یک سطح عملکردی نزدیک به هم قرار گرفته‌اند. به‌طور کلی، با وجود اختلافات جزئی، روش‌های مبتنی بر شباهت در مقایسه با روش FedSwap کمی بهتر عمل کرده‌اند. برای آگاهی از جزئیات بیشتر، به منحنی‌های خطا در شکل آ-۳ پیوست، توجه نمایید.

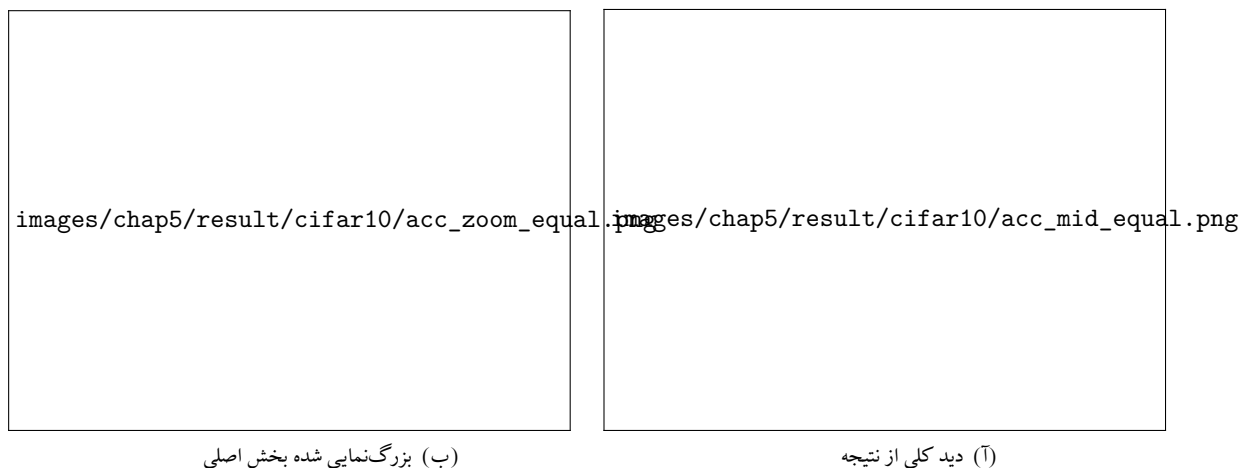
در شکل ۵-۸، آزمایش قبلی دوباره اجرا شده، اما این بار از توزیع داده نرمال استفاده شده است. مشاهده می‌شود که در این وضعیت نیز روش‌های مبتنی بر جابه‌جایی، عملکرد بهتری نسبت به روش FedAvg داشته‌اند. البته، نتایج حاصل از روش‌های جابه‌جایی تقریباً مشابه بوده و تفاوت قابل توجهی بین آن‌ها دیده نمی‌شود. برای مشاهده جزئیات بیشتر، می‌توان به منحنی‌های خطا در شکل آ-۴ پیوست، مراجعه کرد.

## ۵-۵ مجموعه داده CINIC-10<sup>۱</sup>

مجموعه داده CINIC-10 یک مجموعه داده تصویری گسترده و متنوع است که برای ارزیابی عملکرد مدل‌های یادگیری ماشین به ویژه در زمینه‌های مرتبط با طبقه‌بندی تصاویر مورد استفاده قرار می‌گیرد [۵۲]. این مجموعه داده، ترکیبی از تصاویر موجود در مجموعه داده‌های معروف CIFAR-10 و ImageNet است. این ترکیب به منظور ایجاد مجموعه‌ای گسترده‌تر و متنوع‌تر از تصاویر انجام شده است که می‌تواند به ارزیابی دقیق‌تر و واقع‌گرایانه‌تر مدل‌ها کمک کند.

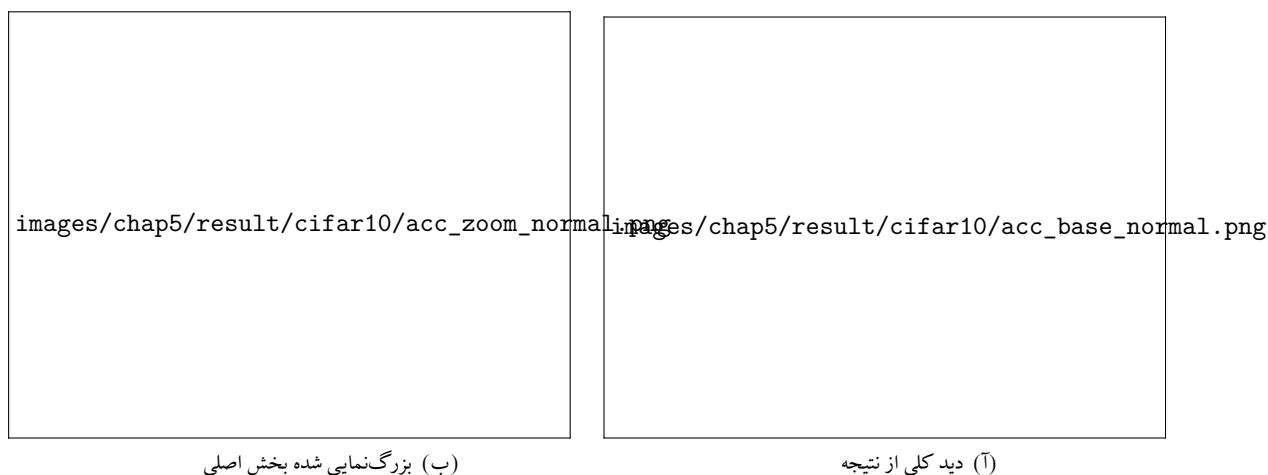
مجموعه داده CINIC-10 شامل ۲۷۰,۰۰۰ تصویر است که در ۱۰ کلاس مختلف دسته‌بندی شده‌اند. هر کلاس شامل ۲۷,۰۰۰ تصویر است که به دو بخش آموزشی و آزمایشی تقسیم شده‌اند. بخش آموزشی شامل ۱۸۰,۰۰۰ تصویر و بخش آزمایشی شامل ۹۰,۰۰۰ تصویر است. این تقسیم‌بندی منظم به محققان و مهندسان یادگیری ماشین این امکان را می‌دهد که به راحتی مدل‌های خود را آموزش داده، اعتبارسنجی و آزمایش کنند. تصاویر موجود در CINIC-10 دارای ابعاد  $۳۲ \times ۳۲$  پیکسل هستند که مشابه ابعاد تصاویر موجود در مجموعه داده CIFAR-10 است. این ویژگی باعث می‌شود که مدل‌های از پیش آموزش دیده بر روی CIFAR-10 بتوانند به

<sup>۱</sup> CIFAR-10 and ImageNet Combined



شکل ۵-۷: مقایسه منحنی‌های دقت در مجموعه داده CIFAR-10 با توزیع داده یکنواخت.

جدول ۵-۲: پارامترهای اجرا در مجموعه داده CIFAR-10										
$h_2$	$h_1$	$E$	$\eta$	$SP$	$C$	$B$	$K$	نحوه جابه‌جایی	شبکه عصبی	مجموعه داده
10	3	2	0.001	1.0	1.0	64	10	MSS	Conv	CIFAR-10



شکل ۵-۸: مقایسه منحنی‌های دقت در مجموعه داده CIFAR-10 با توزیع داده نرمال.

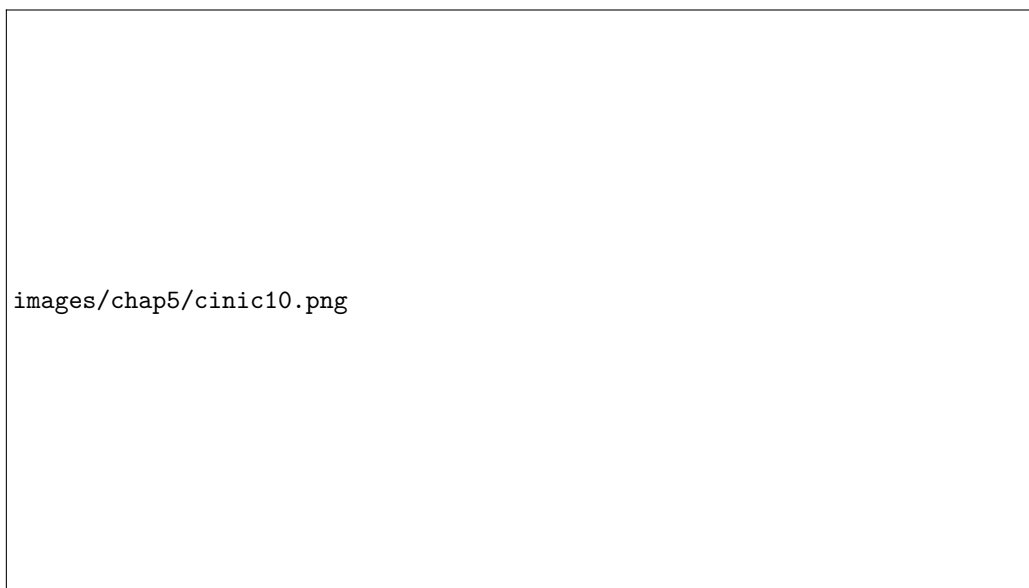
راحتی بر روی این مجموعه داده نیز مورد استفاده قرار گیرند و ارزیابی شوند. با این حال، تنوع بیشتر تصاویر در CINIC-10 نسبت به CIFAR-10 به دلیل ترکیب تصاویر از ImageNet، چالشی جدی‌تر برای مدل‌های یادگیری ماشین فراهم می‌کند. در شکل ۵-۹، تعدادی نمونه از کلاس خودرو در مجموعه داده CIFAR-10 به نمایش درآمده است.

یکی از اهداف اصلی ایجاد CINIC-10، افزایش تنوع و پیچیدگی تصاویر مورد استفاده برای آموزش و ارزیابی مدل‌ها بوده است. این مجموعه داده شامل تصاویری از دنیای واقعی است که در شرایط نوری مختلف و با پس‌زمینه‌های متنوع گرفته شده‌اند. این ویژگی به مدل‌ها کمک می‌کند تا به جای این که تنها بر روی مجموعه‌ای

محدود از تصاویر آموزش ببینند، توانایی تعمیم‌دهی خود را به تصاویر جدید و غیرمنتظره نیز افزایش دهند. در نهایت، CINIC-10 با هدف ارتقای استانداردهای ارزیابی مدل‌های یادگیری عمیق و بهبود عملکرد آن‌ها در مواجهه با داده‌های واقعی و متنوع ایجاد شده است. این مجموعه داده به محققان این امکان را می‌دهد که مدل‌های خود را در شرایط نزدیک به دنیای واقعی آزمایش کرده و نقاط ضعف و قوت آن‌ها را بهتر شناسایی کنند. به همین دلیل، CINIC-10 به عنوان یک ابزار ارزشمند در جامعه یادگیری ماشین شناخته می‌شود و به‌طور گسترده‌ای مورد استفاده قرار می‌گیرد.

اکنون نتایج مربوط به مجموعه داده CINIC-10 بررسی خواهد شد که شکل ۵-۱۰ نتایج مقایسه روش SimFedSwap با سایر روش‌های مرجع را به تصویر می‌کشد. همچنین، پارامترهای به کار رفته در این آزمایش در جدول ۵-۳ ارائه شده‌اند.

در شکل ۵-۱۰ به وضوح می‌توان مشاهده کرد که روش‌های مبتنی بر جابه‌جایی در مقایسه با روش FedAvg، عملکرد متفاوتی داشته‌اند. هرچند، این روش‌ها همچنان در یک سطح عملکردی نزدیک به هم قرار دارند و تفاوت‌های عمده‌ای میان آن‌ها دیده نمی‌شود. برای بررسی دقیق‌تر، منحنی‌های خطا در شکل ۵-۲ پیوست، به تفصیل آمده‌اند.



شکل ۵-۹: تعدادی نمونه از کلاس خودرو در مجموعه داده CINIC-10 [۵۲].

جدول ۵-۳: پارامترهای اجرا در مجموعه داده CINIC-10										
مجموعه داده	شبکه عصبی	نحوه جابه‌جایی	توزیع داده	$K$	$B$	$C$	$SP$	$\eta$	$E$	$h_1$ $h_2$
CINIC-10	Conv	MSS	نرمال	30	64	0.5	1.0	0.001	1	2 5

## ۵-۶ مجموعه داده FEMNIST<sup>۱</sup>

مجموعه داده FEMNIST یک مجموعه داده توسعه‌یافته از مجموعه مشهور MNIST است که برای کاربردهای یادگیری فدرال طراحی شده است [۵۳]. این مجموعه داده شامل ۸۱۴,۲۵۵ تصویر است که در ۶۲ کلاس مختلف دسته‌بندی شده‌اند و ۱۰ درصد این داده‌ها به بخش آزمایشی تعلق دارند. مجموعه داده FEMNIST از تصاویر دست‌نوشته به وجود آمده است که شامل اعداد و حروف الفبای انگلیسی می‌شود.

برخلاف مجموعه داده MNIST که تنها شامل اعداد دست‌نوشته از صفر تا نه است، مجموعه داده FEMNIST شامل حروف بزرگ و کوچک الفبای انگلیسی نیز می‌باشد. این ویژگی باعث می‌شود که FEMNIST نسبت به MNIST تنوع بیشتری داشته باشد و برای آزمایش مدل‌های پیچیده، مناسب‌تر باشد. چند نمونه از اعضای این مجموعه داده در شکل ۵-۱۱، نمایش داده شده‌اند. در این مجموعه داده، تعداد داده‌ها در هر کلاس یکسان نیست و کلاس‌های مختلف دارای تعداد متفاوتی از داده‌ها هستند. شکل ۵-۱۲، تعداد داده‌های هر کلاس و نحوه نام‌گذاری آن‌ها را نشان می‌دهد.

همان‌طور که در شکل ۵-۱۲ مشاهده می‌شود، تعداد کلاس‌های ۰ تا ۹ که به ارقام ۰ تا ۹ اشاره دارند، به‌طور قابل‌توجهی بیشتر از سایر کلاس‌هاست و هر کدام حدود ۴۰,۰۰۰ نمونه دارند. در بین کلاس‌های ۱۰ تا ۳۵ که مربوط به حروف بزرگ انگلیسی هستند، کلاس‌های S و O بیشترین تعداد نمونه را دارند. به نظر می‌رسد این سبک از جمع‌آوری داده به دلیل جلوگیری از اشتباه گرفتن کلاس O با عدد صفر و کلاس S با معادل حرف کوچک آن در کلاس‌های ۳۶ تا ۶۱ بوده باشد.

یکی از ویژگی‌های برجسته مجموعه داده FEMNIST، نحوه سازماندهی داده‌ها است. این مجموعه داده بر



(ب) بزرگ‌نمایی شده بخش اصلی

(آ) دید کلی از نتیجه

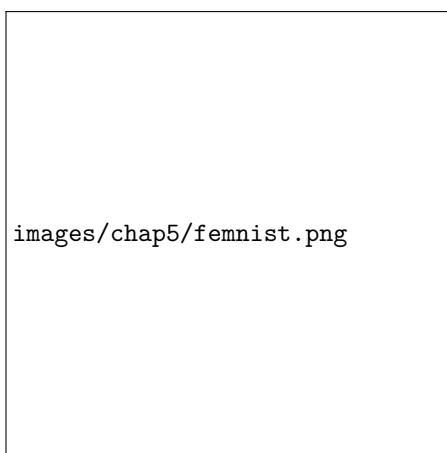
شکل ۵-۱۰: مقایسه منحنی‌های دقت در مجموعه داده CINIC-10.

<sup>۱</sup>Federated Extended MNIST

اساس کاربران مختلف تقسیم‌بندی شده است، به طوری که هر کاربر دارای مجموعه‌ای از داده‌های دست‌نوشته خود است. این سازماندهی امکان آزمایش و ارزیابی روش‌های یادگیری فدرال را فراهم می‌کند، زیرا در یادگیری فدرال داده‌ها به صورت محلی بر روی دستگاه‌های کاربران، نگه‌داری می‌شوند و مدل‌ها بر روی این داده‌ها آموزش می‌بینند. این ویژگی به محققان اجازه می‌دهد تا سناریوهای واقعی‌تری از یادگیری فدرال را شبیه‌سازی و بررسی کنند.

مجموعه داده FEMNIST به صورت پیش فرض شامل ۳۵۹۷ کاربر است که داده‌ها میان این کاربران توزیع شده‌اند. این توزیع، نه از لحاظ تعداد تصاویر بین کاربران و نه از لحاظ پوشش‌دهی کلاس‌ها در هر کاربر، یکسان نیست. با این حال، تعداد کاربران و نحوه توزیع داده‌ها میان آن‌ها را می‌توان به دلخواه تغییر داد. برای بررسی حالت پیش فرض، می‌توان مشاهده کرد که هر کاربر چه تعداد کلاس را پوشش داده است. در شکل ۵-۱۳ قابل مشاهده است که هر کاربر چند کلاس را شامل می‌شود. به عنوان مثال، این شکل نشان می‌دهد که حدود ۴۰۰ کاربر وجود دارند که هر کدام ۵۸ کلاس را پوشش داده‌اند. همچنین برای بررسی تعداد تصاویری که در هر کاربر وجود دارد، می‌توان به شکل ۵-۱۴ توجه کرد. این شکل نشان می‌دهد که حدوداً ۴۸۰ کاربر وجود دارند که هر کدام ۱۷۵ تصویر را شامل می‌شوند.

تصاویر در مجموعه داده FEMNIST به صورت سیاه و سفید و با اندازه  $28 \times 28$  پیکسل هستند. هر تصویر نمایانگر یک کاراکتر دست‌نوشته است. این تصاویر از مجموعه داده NIST استخراج شده‌اند و به صورت مناسبی برای کاربردهای یادگیری فدرال سازماندهی شده‌اند. در حقیقت این تصاویر شامل نویسه‌های مختلف از کاربران مختلف است که تنوع در سبک نوشتن و کیفیت دست‌نوشته‌ها را افزایش می‌دهد. به طور کلی، مجموعه داده FEMNIST یک ابزار قدرتمند برای تحقیقات در زمینه یادگیری فدرال است. با



شکل ۵-۱۱: چند نمونه از اعضای مجموعه داده FEMNIST [۴۹].

images/chap5/count\_all\_classes.png

شکل ۵-۱۲: تعداد داده‌های هر کلاس و نحوه نام‌گذاری در مجموعه داده FEMNIST.

ارائه تنوع بالای داده‌ها و سازماندهی مناسب برای سناریوهای یادگیری فدرال، این مجموعه داده به محققان کمک می‌کند تا روش‌ها و الگوریتم‌های جدید را در محیط‌های واقعی‌تر آزمایش کنند. این ویژگی‌ها باعث شده تا FEMNIST به عنوان یکی از مجموعه داده‌های مرجع در این حوزه شناخته شود و در بسیاری از تحقیقات علمی و صنعتی مورد استفاده قرار گیرد.

#### ۵-۶-۱ رویکردهای پایه در مجموعه داده FEMNIST

این مجموعه داده در دو رویکرد مختلف بررسی خواهد شد. در رویکرد اول، داده‌ها بدون توجه به کاربران اصلی و تنها بر اساس کلاس‌های آن‌ها تفکیک می‌شوند. به این صورت که تمام داده‌های مربوط به هر کلاس جمع‌آوری شده و طبق یک توزیع مشخص بین تعدادی کاربر تقسیم می‌شوند. این روش به عنوان رویکرد کلاس‌بندی یا FEMNISTclass شناخته می‌شود.

در رویکرد دوم، ساختار اصلی مجموعه داده تغییر نمی‌کند و تعداد کاربران همان تعداد پیش‌فرض باقی می‌ماند. همچنین داده‌ها دقیقاً به همان شیوه‌ای که به هر کاربر اختصاص داده شده‌اند، حفظ می‌شوند. این روش به نام رویکرد نویسندگان یا FEMNISTwriter نام‌گذاری شده است. در ادامه نتایج مربوط به هر کدام از این رویکردها به صورت مجزا بررسی خواهد شد.

#### ۵-۶-۲ مقایسه نتایج در رویکرد کلاس‌بندی (FEMNISTclass)

نتایج مقایسه روش SimFedSwap با دیگر روش‌های مرجع در شکل ۵-۱۵ نمایش داده شده‌اند که این آزمایش بر روی مجموعه داده FEMNISTclass انجام شده است. پارامترهای مورد استفاده در این آزمایش نیز در جدول



images/chap5/clients\_cover\_classes.png

شکل ۵-۱۳: تعداد کلاس‌های پوشش داده شده توسط کاربران در مجموعه داده FEMNIST.

images/chap5/clients\_images.png

شکل ۵-۱۴: تعداد تصاویر هر یک از کاربران در مجموعه داده FEMNIST.

۵-۴ ذکر شده‌اند.

از شکل ۵-۱۵ مشخص است که روش‌های مبتنی بر جابه‌جایی در مقایسه با FedAvg عملکرد متفاوتی نشان می‌دهند، اما همچنان تفاوت‌های عملکردی بین آن‌ها محدود و نزدیک به هم است. نکته قابل توجه این است

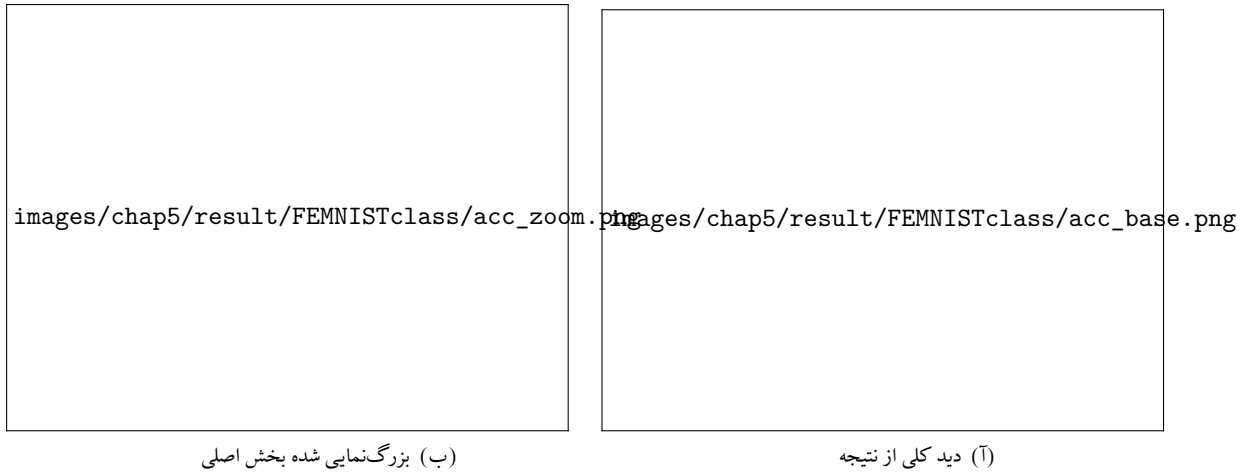
جدول ۵-۴: پارامترهای اجرا در مجموعه داده FEMNISTclass

مجموعه داده	شبکه عصبی	نحوه جابه‌جایی	توزیع داده	$K$	$B$	$C$	$SP$	$\eta$	$E$	$h_1$	$h_2$
FEMNISTclass	Conv	MSS	یکنواخت	200	1024	1.0	1.0	0.001	2	5	3

که بسیاری از تغییرات در نمودارها به لحظات جابه‌جایی یا میانگین‌گیری مربوط می‌شوند. برای جزئیات بیشتر و بررسی دقیق‌تر، می‌توان به منحنی‌های خطا که در شکل آ-۶ پیوست آمده‌اند، مراجعه کرد.

### ۳-۶-۵ مقایسه نتایج در رویکرد نویسندگان (FEMNISTwriter)

نتایج مربوط به مقایسه روش SimFedSwap با سایر روش‌های مرجع در شکل ۵-۱۶ قابل مشاهده است. این آزمایش بر روی مجموعه داده FEMNISTwriter اجرا شده و پارامترهای به‌کاررفته در آن نیز در جدول ۵-۵ ذکر شده‌اند.



شکل ۵-۱۵: مقایسه منحنی‌های دقت در مجموعه داده FEMNISTclass.



شکل ۵-۱۶: مقایسه منحنی‌های دقت در یک اجرا بر روی مجموعه داده FEMNISTwriter.

جدول ۵-۵: پارامترهای اجرا در مجموعه داده FEMNISTwriter

مجموعه داده	شبکه عصبی	نحوه جابه‌جایی	توزیع داده	$K$	$B$	$C$	$SP$	$\eta$	$E$	$h_1$	$h_2$
FEMNISTwriter	Conv	MSS	یکنواخت	3597	64	0.15	1.0	0.001	1	5	3

در شکل ۵-۱۶ مشاهده می‌شود که روش‌های مبتنی بر جابه‌جایی نتایجی متفاوت از روش FedAvg ارائه داده‌اند. اما نکته مهم، برتری قابل توجه روش‌های مبتنی بر شباهت نسبت به FedSwap است. این اولین آزمایشی است که در آن روش‌های شباهت محور توانسته‌اند عملکرد بهتری را به‌طور معناداری ارائه کنند. برای اطلاعات بیشتر و تحلیل دقیق‌تر، می‌توان به منحنی‌های خطا در شکل آ-۷ پیوست، مراجعه کرد.

با بهبود نتایج، این پرسش پیش می‌آید که آیا این برتری در شرایط استفاده از چندین Seed متفاوت، همچنان پابرجا خواهد بود. برای پاسخ به این سوال، آزمایش قبلی با پنج Seed مختلف تکرار شده و میانگین نتایج در شکل ۵-۱۷ نمایش داده شده‌اند. نکته قابل توجه این است که اختلاف روش مبتنی بر شباهت با روش FedSwap، دیگر به وضوح قبلی دیده نمی‌شود و تنها، بهبودی حدود یک درصد در میانگین پنج اجرا مشاهده می‌شود. همچنین باید به این نکته توجه شود که تغییر Seed در نمودارهای پیشین، تفاوت چشم‌گیری در خروجی ایجاد نمی‌کردند. برای بررسی جزئیات بیشتر، منحنی‌های خطا در شکل آ-۸ پیوست ارائه شده‌اند.

بنابراین می‌توان نتیجه گرفت که با افزایش تعداد کاربران و داده‌های مربوط به آن‌ها، روش‌های مبتنی بر شباهت، هرچند به میزان کم، می‌توانند عملکرد بهتری نشان دهند. باید به این نکته توجه داشت که دقت کلی که در این مجموعه داده به ۵۰ درصد رسید، وابسته به طراحی شبکه عصبی است. با بهینه‌سازی این شبکه، می‌توان به دقت بالاتری دست یافت. با این حال، این بهینه‌سازی تأثیری بر مقایسه بین روش‌ها نخواهد داشت، زیرا همه روش‌ها به‌طور همزمان به دقت بهتری دست خواهند یافت.

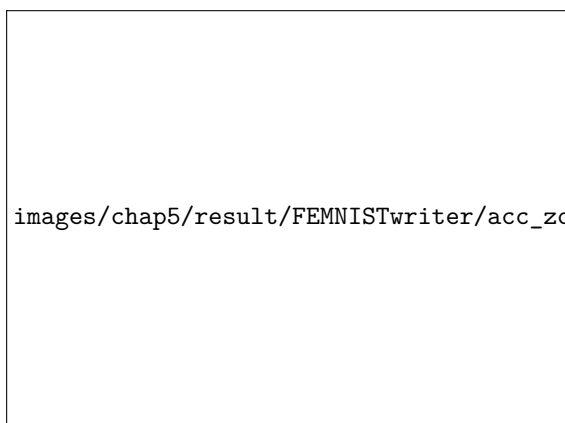
## ۵-۷ مقایسه جابه‌جایی حریصانه با جابه‌جایی حداقل شباهت در روش SimFedSwap

شکل ۵-۱۸ عملکرد دو روش جابه‌جایی حریصانه و جابه‌جایی حداقل شباهت را در الگوریتم SimFedSwap مقایسه می‌کند. این ارزیابی روی مجموعه داده CIFAR-10 و با توزیع یکنواخت داده‌ها میان کاربران انجام شده است. مشخصات پارامترهای این آزمایش در جدول ۵-۲ آمده است.

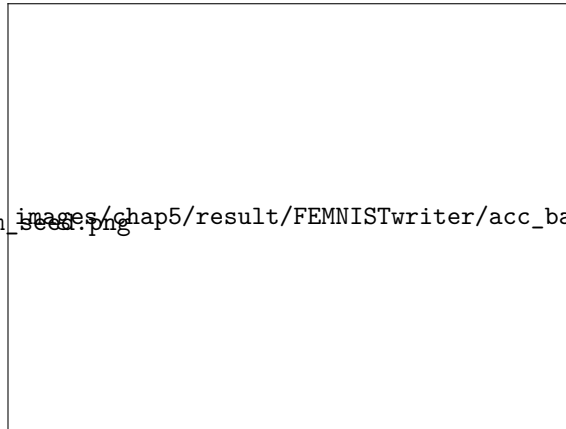
بر اساس نتایج شکل ۵-۱۸، عملکرد این دو روش بسیار مشابه بوده و تفاوت قابل توجهی مشاهده نمی‌شود. برای اطلاعات بیشتر، می‌توان به منحنی‌های خطا در شکل آ-۹ پیوست، مراجعه کرد.

شکل ۵-۱۹ عملکرد دو روش جابه‌جایی حریصانه و جابه‌جایی حداقل شباهت را در الگوریتم SimFedSwap مقایسه می‌کند. این آزمایش بر روی مجموعه داده FEMNISTwriter انجام شده و پارامترهای مربوط به این بررسی در جدول ۵-۵ آورده شده است.

نتایج به دست آمده از شکل ۵-۱۹ نشان می‌دهد که معیار OSAD با استفاده از جابه‌جایی حریصانه عملکردی مشابه روش FedSwap داشته است. اما روش‌های CKA با استفاده از هسته‌های خطی و گاوسی به تدریج عملکرد



(ب) بزرگ‌نمایی شده بخش اصلی



(آ) دید کلی از نتیجه

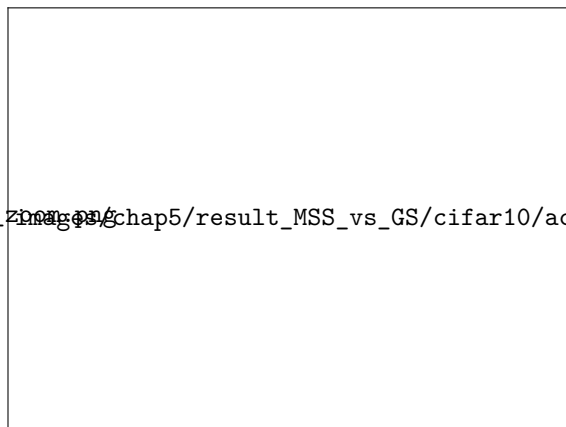
images/chap5/result/FEMNISTwriter/acc\_zoom.png

images/chap5/result/FEMNISTwriter/acc\_base\_seed.png

شکل ۵-۱۷: مقایسه منحنی‌های دقت در میانگین پنج اجرا بر روی مجموعه داده FEMNISTwriter.



(ب) بزرگ‌نمایی شده بخش اصلی



(آ) دید کلی از نتیجه

images/chap5/result\_MSS\_vs\_GS/cifar10/acc\_zoom.png

images/chap5/result\_MSS\_vs\_GS/cifar10/acc\_mid.png

شکل ۵-۱۸: مقایسه منحنی‌های دقت بین MSS و GS، در مجموعه داده CIFAR-10 با توزیع داده یکنواخت.

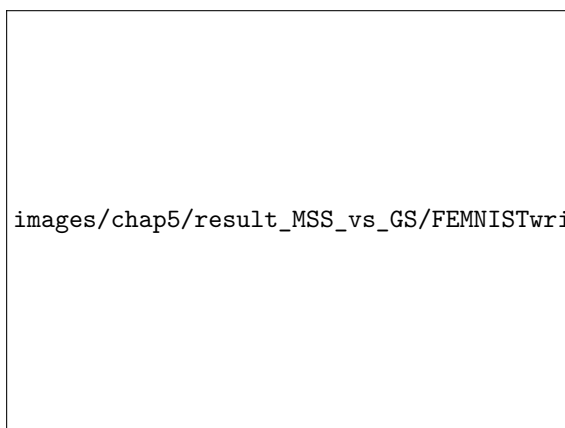
خود را بهبود بخشیده و به نتایج بهتری دست یافته‌اند. برای اطلاعات دقیق‌تر، منحنی‌های خطا در شکل آ-۱۰ پیوست، قابل بررسی هستند.

## ۵-۸ تحلیل کاهش تعداد کاربران در هر دور و افزایش تعداد کل دورها در روش SimFedSwap

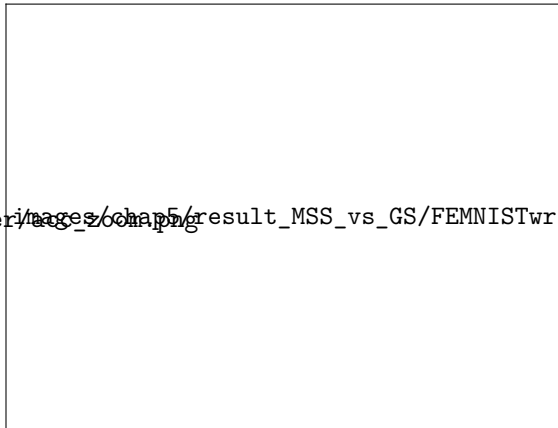
در این بخش بررسی می‌شود که اگر تعداد کاربران شرکت کننده در هر دور به‌طور قابل توجهی کاهش یابد و در مقابل تعداد کل دورها افزایش پیدا کند، آیا مدل سراسری همچنان آموزش دیده و به همگرایی خواهد رسید.

شکل ۵-۲۰ به تحلیل تأثیر کاهش تعداد کاربران در هر دور و افزایش تعداد دورهای کلی در الگوریتم SimFedSwap پرداخته است. این آزمایش با استفاده از مجموعه داده CINIC-10 انجام شده و پارامترهای مورد استفاده، در جدول ۵-۶ به نمایش درآمده‌اند.

بر اساس نتایج ارائه‌شده در شکل ۵-۲۰، حتی با وجود مشارکت کم کاربران در هر دور، مدل سراسری



(ب) بزرگ‌نمایی شده بخش اصلی

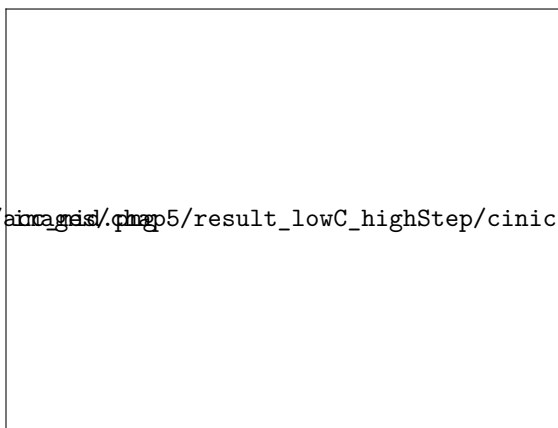


(آ) دید کلی از نتیجه

شکل ۵-۱۹: مقایسه منحنی‌های دقت بین MSS و GS، در مجموعه داده FEMNISTwriter.



(ب) بزرگ‌نمایی شده بخش اصلی



(آ) دید کلی از نتیجه

شکل ۵-۲۰: مقایسه منحنی‌های دقت در مجموعه داده CININC-10 با کاهش مشارکت کاربران و افزایش کل دورها.

توانسته است به همگرایی برسد. اما برای این کار به بیش از ۶۰۰۰ دور احتیاج داشته است. همچنین، تمامی روش‌ها تقریباً عملکرد مشابهی را ارائه داده‌اند. برای جزئیات بیشتر، منحنی‌های خطا در شکل آ-۱۱ پیوست، نشان داده شده‌اند.

شکل ۵-۲۱ تأثیر کاهش تعداد کاربران شرکت کننده در هر دور و افزایش تعداد کل دورها را در الگوریتم SimFedSwap بررسی می‌کند. این ارزیابی بر پایه مجموعه داده FEMNISTclass صورت گرفته و پارامترهای مورد استفاده نیز در جدول ۵-۴ آورده شده‌اند، با این تفاوت که مقدار  $C$  برابر با  $۰/۱$  تنظیم شده است.

نتایج موجود در شکل ۵-۲۱ نشان می‌دهد که حتی با کاهش تعداد کاربران در هر دور، مدل سراسری موفق به همگرایی می‌شود. با این حال، این فرآیند به حدود ۱۶۰ دور نیاز داشته است، در حالی که در شکل ۵-۱۵ (آ)، همگرایی تنها با ۴۵ دور به دست آمده بود. این در صورتی است که تمام روش‌های مبتنی بر جابه‌جایی عملکرد مشابهی از خود نشان داده‌اند. برای بررسی دقیق‌تر این نتایج، می‌توان به منحنی‌های خطا در شکل آ-۱۲ پیوست،

جدول ۵-۶: پارامترهای اجرا در مجموعه داده CINIC-10 با کاهش مشارکت کاربران و افزایش کل دورها

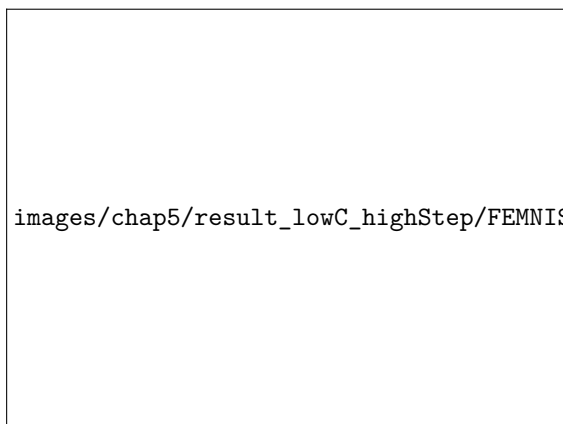
مجموعه داده	شبکه عصبی	نحوه جابه‌جایی	توزیع داده	$K$	$B$	$C$	$SP$	$\eta$	$E$	$h_1$	$h_2$
CINIC-10	Conv	MSS	نرمال	200	64	0.1	1.0	0.001	1	5	3

مراجعه کرد.

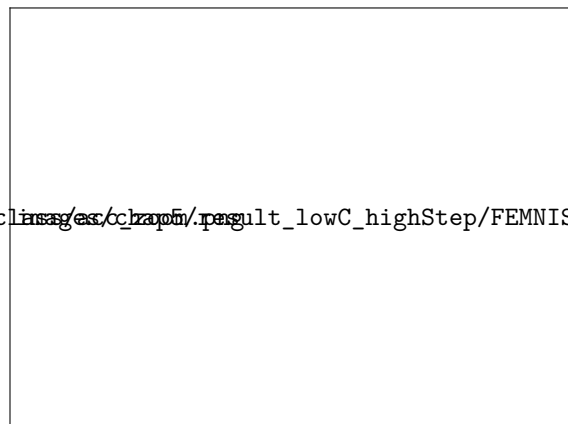
## ۵-۹ بررسی نحوه پیاده‌سازی کد، سخت‌افزار مورد استفاده و زمان اجرای کدها

تمامی کدهای مورد استفاده در این پژوهش با کتابخانه PyTorch پیاده‌سازی شده‌اند. همان‌طور که در پروپوزال بیان شده بود، ابتدا کتابخانه‌های TensorFlow Federated (TFF)، PySyft و Flower مد نظر بودند اما پس از پیاده‌سازی اولیه و رسیدن به مرحله مقایسه شبکه‌های عصبی و جابه‌جایی آن‌ها بین کاربران، مشخص شد که این کتابخانه‌ها ساختار مورد نیاز برای این کار را ندارند. همچنین، افزودن این ویژگی‌ها به کتابخانه‌های مذکور زمان‌بر بوده و این کتابخانه‌ها شامل بسیاری از امکانات غیرضروری برای این پژوهش بودند. به همین دلیل، تصمیم گرفته شد که پیاده‌سازی از پایه با استفاده از PyTorch انجام شود. همچنین کد این پژوهش به‌صورت متن باز در Github موجود می‌باشد.

در ابتدا، اجرای تمامی آزمایش‌ها با درصد کمی از مجموعه داده‌ها در محیط Google Colab صورت گرفت. برای بهینه‌سازی فرآیند، از چند حساب کاربری مختلف استفاده شد تا امکان ذخیره تنظیمات در پایان هر دور و ادامه اجراها در حساب‌های دیگر فراهم شود. پس از تکمیل کد، بررسی‌های نهایی توسط کلاستر محاسباتی که از سوی دانشکده فراهم شده بود، انجام گرفت. همچنین، لازم است از سخت‌افزارهای ارائه‌شده قدردانی شود. در نهایت، تمام نتایج مشاهده‌شده در نمودارها با سیستمی که مشخصات آن در جدول ۵-۷ آمده است، اجرا



(ب) بزرگ‌نمایی شده بخش اصلی



(آ) دید کلی از نتیجه

شکل ۵-۲۱: مقایسه منحنی‌های دقت در مجموعه داده FEMNISTclass با کاهش مشارکت کاربران و افزایش کل دورها.

شده‌اند. برای نمونه، در آزمایش شکل ۵-۲۰، هر دور به‌طور میانگین ۷۰ ثانیه زمان برده که برای ۶۰۰۰ دور، ۴/۸ روز به ازای هر منحنی زمان نیاز بوده است. همچنین، اجرای نتایج مشاهده شده در شکل ۵-۱۵ به‌طور میانگین در هر دور ۴۰ دقیقه زمان برده که برای ۴۵ دور، به ۱/۲۵ روز زمان نیاز داشته است.

این زمان‌ها مربوط به آزمایش‌هایی با تعداد کاربران کم بودند. در صورت افزایش تعداد کاربران، مرحله بررسی شباهت و جابه‌جایی مدل‌ها، به زمان زیادی نیاز خواهد داشت. به عنوان مثال، در اجرای مجموعه داده FEMNISTwriter با ۳۵۹۷ کاربر و مشارکت ۱۵ درصد از آن‌ها (۵۴۰ کاربر) در هر دور، بررسی شباهت و جابه‌جایی حدود ۲۰ دقیقه زمان می‌برد.

برای نمونه، در آزمایش مشاهده شده در شکل ۵-۱۶ با توجه به پارامترهای اجرا، از ۶۱۰ دور، ۸۱ دور به بررسی شباهت و جابه‌جایی اختصاص یافته که هر کدام ۲۰ دقیقه زمان برده و به ازای هر ۵۲۹ دور باقی‌مانده، ۵۰ ثانیه صرف شده است. در مجموع، اجرای هر منحنی به ۱/۴۳ روز زمان نیاز داشته است. اگر این آزمایش‌ها با میانگین پنج Seed انجام شوند، زمان اجرا به شکل قابل‌توجهی افزایش خواهد یافت. بنابراین، در برخی از نتایج، بررسی تمامی روش‌های شباهت به دلیل زمان‌بر بودن، انجام نشده‌اند.

برای استفاده بهینه از سخت‌افزار، به‌جای اجرای یک کد، دو کد به‌صورت همزمان اجرا شده‌اند تا نهایت استفاده از منابع سخت‌افزاری صورت گیرد. همان‌طور که در شکل ۵-۲۲ مشاهده می‌شود، در بیشتر اوقات حدود ۹۹ درصد منابع CUDA<sup>۱</sup> در حال استفاده بوده است.

این بخش به منظور آگاهی‌رسانی به محققانی است که قصد دارند در این زمینه فعالیت کنند. اما باید توجه داشت که در دنیای واقعی، به دلیل مجزا بودن کاربران، زمان اجرا به‌صورت متمرکز محاسبه نمی‌شود.

## ۵-۱۰ جمع‌بندی

در این فصل، دو مدل کلیدی شبکه عصبی، یعنی پرسپترون چندلایه (MLP) و شبکه عصبی پیچشی (CNN) که طراحی و به کار گرفته شدند، مورد بررسی قرار گرفتند. سپس، مجموعه داده‌های مختلف معرفی و نتایج مقایسه روش SimFedSwap با سایر روش‌های مرجع ارائه شد. در اغلب مجموعه داده‌ها، تفاوت زیادی بین روش‌ها

جدول ۵-۷: مشخصات سیستمی کلاستر اجرای کدها

مدل	قطعه
CPU	Intel(R) Core(TM) i7-9700K CPU @ 3.60GHz (8 CPUs)
RAM	32 GB
GPU	NVIDIA GeForce RTX 2080 Ti
Drive	Samsung SSD 860 EVO 250GB

<sup>۱</sup> Compute Unified Device Architecture

مشاهده نشد، اما در مجموعه داده FEMNISTwriter، روش‌های مبتنی بر شباهت عملکرد بهتری از روش‌های مرجع داشتند. این تحلیل‌ها نشان می‌دهد که وقتی شبکه به راحتی به دقت بالا می‌رسد و تعداد کاربران محدود است، اختلاف بین روش‌ها چندان قابل توجه نیست. با این حال، در مجموعه داده‌های پیچیده‌تر و با افزایش تعداد کاربران، این تفاوت‌ها آشکارتر می‌شوند. همچنین، کاهش تعداد کاربران در هر دور و افزایش تعداد کل دورها به همگرایی مدل منجر می‌شود، اما زمان بیشتری برای رسیدن به این همگرایی لازم است.

images/chap5/task\_manager\_inverted\_color.png

شکل ۵-۲۲: میزان استفاده از سخت‌افزار موجود، هنگام اجرای کد.



## فصل ششم

### نتیجه‌گیری و پیشنهادها

#### ۱-۶ نتیجه‌گیری

در دنیای امروزی با رشد سریع فناوری و افزایش تعداد دستگاه‌های متصل به اینترنت، نیاز به برقراری ارتباطات مؤثر و حفاظت از حریم شخصی کاربران بیش از پیش اهمیت یافته است. این مسئله منجر به توسعه روش‌های توزیع‌شده‌ای مانند یادگیری فدرال شده است. در یادگیری فدرال، داده‌ها به‌جای ارسال به یک سرور مرکزی برای پردازش، در محل خود دستگاه‌ها نگهداری می‌شوند و مدل‌ها به‌صورت محلی آموزش داده می‌شوند. سپس این مدل‌ها با یکدیگر ترکیب می‌شوند تا یک مدل سراسری به‌دست آید. این روش علاوه بر کاهش نیاز به انتقال داده‌ها، حریم شخصی کاربران را نیز بهتر حفظ می‌کند.

با این حال، یادگیری فدرال با چالش‌های متعددی مواجه است. یکی از این چالش‌ها، ناهمگنی آماری داده‌ها است. به این معناست که داده‌های موجود در دستگاه‌های مختلف می‌توانند بسیار متفاوت و گوناگون باشند. این ناهمگنی باعث می‌شود که مدل‌های محلی نتوانند به‌طور کامل ویژگی‌های تمامی داده‌ها را یاد بگیرند و در نتیجه، مدل سراسری نیز به‌خوبی همگرا نشود. در این شرایط، رسیدن به یک مدل سراسری که عملکرد قابل قبولی داشته باشد، ممکن است با مشکل مواجه شود.

برای مقابله با این چالش، یکی از راهکارهای پیشنهادی، جابه‌جایی وزن‌های شبکه عصبی بین کاربران

نهایی در طول فرآیند یادگیری است. این روش باعث می‌شود که مدل‌های محلی با داده‌های متنوع‌تری روبه‌رو شوند و در نتیجه، مدل سراسری بتواند به یک همگرایی بهتر دست یابد. به عبارت دیگر، با جابه‌جایی وزن‌ها، مدل‌ها به داده‌های بیشتری دسترسی پیدا می‌کنند که این امر به بهبود کیفیت یادگیری و همگرایی مدل کمک می‌کند.

در روش‌های متداول، جابه‌جایی وزن‌ها به‌طور تصادفی انجام می‌شود، اما در این پژوهش پیشنهاد گردید که به‌جای استفاده از روش تصادفی، این جابه‌جایی به‌صورت هوشمندانه و بر اساس معیارهای شباهت انجام شود، به این صورت که مدل‌هایی که کمترین میزان شباهت را با یکدیگر دارند، جابه‌جا شوند. این رویکرد باعث می‌شود که مدل‌ها با داده‌هایی که به‌طور کامل با آن‌ها آشنا نیستند، روبه‌رو شوند و این امر می‌تواند به بهبود همگرایی مدل سراسری منجر شود. به عبارت دیگر، این نوع جابه‌جایی هوشمندانه می‌تواند مدل سراسری را سریع‌تر و بهتر به همگرایی نهایی هدایت کند.

یکی دیگر از جنبه‌های این پژوهش، بررسی تأثیر جابه‌جایی وزن‌ها بر حفظ حریم شخصی کاربران بود. روش‌های مرسوم جابه‌جایی، این فرآیند را به‌طور مستقیم بین کاربران نهایی انجام می‌دهند. این روش اگرچه می‌تواند به کاهش سربار شبکه کمک کند، اما ممکن است به تشدید مشکلات حریم شخصی منجر شود. در این پژوهش، پیشنهاد شد که سرور مرکزی به عنوان واسطه در فرآیند جابه‌جایی عمل کند. با این کار، حریم شخصی کاربران بهتر حفظ می‌شود و پیاده‌سازی روش‌هایی مانند حفظ حریم خصوصی تفاضلی نیز ساده‌تر می‌شود.

باید توجه داشت که هرچند این روش بهبود قابل‌توجهی در حفظ حریم شخصی ایجاد می‌کند، اما نسبت به روش‌های جابه‌جایی وزن‌ها بدون دخالت سرور، اندکی به سربار شبکه می‌افزاید. با این حال، این سربار در مقایسه با روش‌های سنتی مانند میانگین‌گیری فدرال همچنان بسیار کمتر خواهد بود. در واقع، مزایایی که این روش در بهبود همگرایی و حفاظت از حریم شخصی به ارمغان می‌آورد، به‌خوبی افزایش جزئی سربار شبکه را نسبت به روش‌های جابه‌جایی بدون دخالت سرور جبران نموده و آن را به یک گزینه مطلوب و کارآمد تبدیل خواهد کرد. در نهایت، این پژوهش نشان داد که جابه‌جایی مدل‌های شبکه عصبی به‌صورت هوشمندانه و بر اساس معیارهای شباهت، می‌تواند فرآیند همگرایی مدل سراسری را تسریع کند. این امر به‌ویژه در شرایطی که تعداد کاربران زیاد باشد، اثرات مثبت خود را بهتر نشان می‌دهد. به بیان دیگر، هرچه تعداد کاربران بیشتر باشد، مزایای استفاده از این روش بیشتر خواهد بود و بهبود قابل‌توجهی در فرآیند همگرایی مشاهده می‌شود.

بنابراین، در مواردی که تعداد کاربران زیاد است و داده‌ها بسیار متفاوت و پراکنده هستند، استفاده از روش جابه‌جایی وزن‌ها بر پایه شباهت می‌تواند بهبود قابل‌ملاحظه‌ای در کارایی و سرعت همگرایی مدل سراسری به‌همراه داشته باشد. این روش نه تنها از نظر فنی مؤثر است، بلکه از دیدگاه حفاظت از حریم شخصی نیز بسیار

مناسب به نظر می‌رسد.

در نتیجه، این پژوهش اهمیت و تأثیر جابه‌جایی هوشمندانه وزن‌های شبکه‌های عصبی در یادگیری فدرال را برجسته کرد و راهکاری جدید برای بهبود فرآیند همگرایی و حفظ حریم شخصی کاربران ارائه داد. این رویکرد می‌تواند به عنوان یک راه حل مؤثر در مواجهه با چالش‌های موجود در یادگیری فدرال مورد استفاده قرار گیرد.

## ۲-۶ پیشنهادها

با توجه به رویکردهای بررسی شده در زمینه حل چالش‌های یادگیری فدرال، پیشنهادهای زیر برای ادامه این پژوهش مطرح می‌گردند:

### ۱. بررسی معیارهای مختلف و انتخاب کمینه شباهت

همان‌گونه که در روند اجرای روش پیشنهادی بیان شد، در آغاز یک معیار شباهت در سرور مشخص می‌شود و تمام مقایسه‌های شبکه‌های عصبی بر اساس این معیار انجام می‌گیرد. با این حال، می‌توان چند معیار را برای ارزیابی شبکه‌های عصبی مد نظر قرار داد. به عنوان نمونه، ماتریس بالا مثلثی برای همه معیارها محاسبه می‌شود و در مرحله انتخاب دو مدل برای جابه‌جایی، مقدار کمینه از میان تمام معیارها برگزیده می‌شود.

### ۲. میانگین وزن دار بین لایه‌های مختلف

هنگام مقایسه دو مدل شبکه عصبی، این مقایسه به صورت لایه به لایه انجام شده و در نهایت میانگین تمامی لایه‌ها محاسبه می‌شود. سوالی که مطرح می‌شود این است که آیا تأثیر تمام لایه‌ها به یک اندازه است که از میانگین‌گیری با وزن یکسان استفاده می‌شود؟ در این جا، می‌توان اهمیت هر لایه در شبکه عصبی را تعیین کرد و سپس میانگین وزن دار را برای مقایسه به کار برد.

### ۳. بررسی شباهت بین لایه‌های مختلف دو شبکه عصبی، نه فقط لایه‌های متناظر

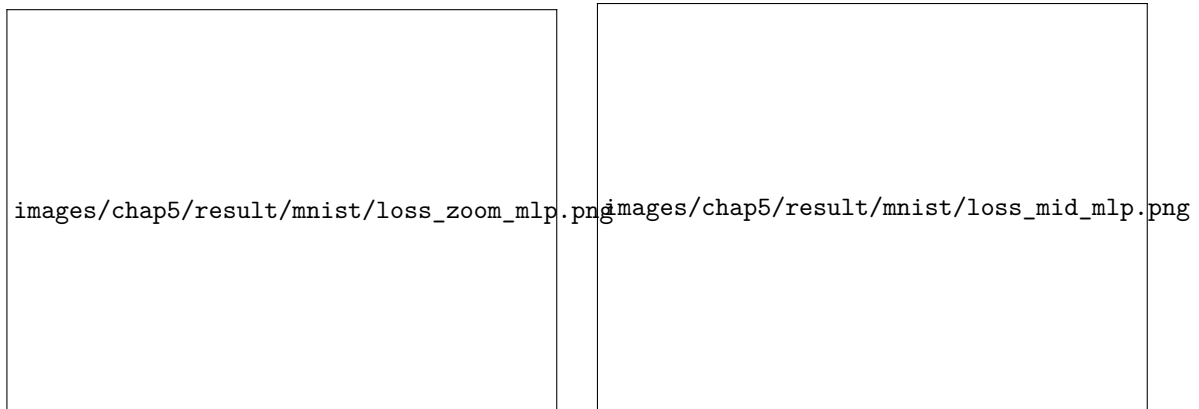
در روش موجود، مقایسه دو شبکه عصبی تنها با بررسی لایه‌های متناظر و محاسبه معیار شباهت انجام می‌گردد. با این حال، از آن جا که در شبکه‌های عصبی ممکن است بین لایه‌های مختلف نیز ارتباط وجود داشته باشد، امکان مقایسه لایه‌های غیرمتناظر نیز وجود دارد. از طریق ارزیابی این لایه‌ها، می‌توان شاخصی مناسب برای تعیین شباهت کلی شبکه‌ها، استخراج کرد.

## پیوست اول

### بررسی نمودارهای خطا

آ-۱ مقایسه روش SimFedSwap با روش‌های پایه

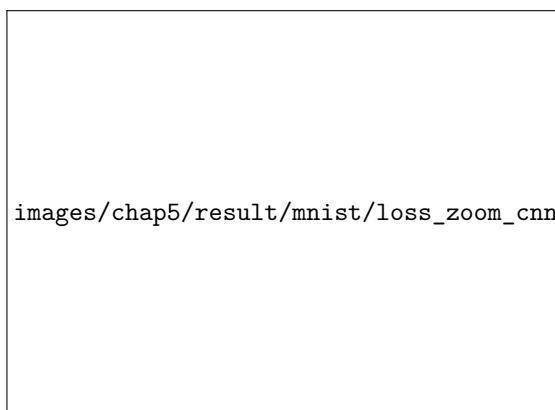
آ-۱-۱ مجموعه داده MNIST



(ب) بزرگ‌نمایی شده بخش اصلی

(آ) دید کلی از نتیجه

شکل آ-۱: مقایسه منحنی‌های خطا در مجموعه داده MNIST با استفاده از مدل MLP.



(ب) بزرگ‌نمایی شده بخش اصلی



(آ) دید کلی از نتیجه

شکل آ-۲: مقایسه منحنی‌های خطا در مجموعه داده MNIST با استفاده از مدل CNN.

## آ-۱-۲ مجموعه داده CIFAR-10



(ب) بزرگ‌نمایی شده بخش اصلی



(آ) دید کلی از نتیجه

شکل آ-۳: مقایسه منحنی‌های خطا در مجموعه داده CIFAR-10 با توزیع داده یکنواخت.



images/chap5/result/cifar10/loss\_zoom\_normal.png

(ب) بزرگ‌نمایی شده بخش اصلی



images/chap5/result/cifar10/loss\_base\_normal.png

(آ) دید کلی از نتیجه

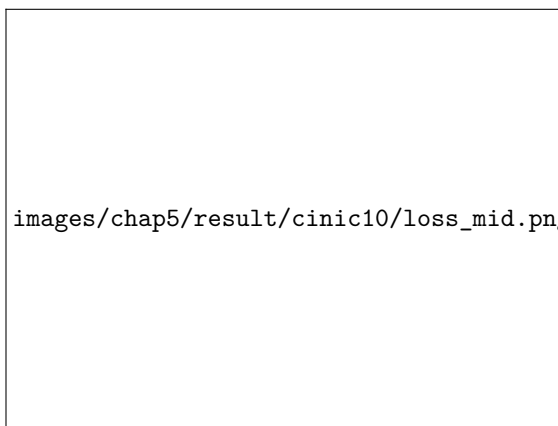
شکل آ-۴: مقایسه منحنی‌های خطا در مجموعه داده CIFAR-10 با توزیع داده نرمال.

### آ-۱-۳ مجموعه داده CINIC-10



images/chap5/result/cinic10/loss\_zoom.png

(ب) بزرگ‌نمایی شده بخش اصلی



images/chap5/result/cinic10/loss\_mid.png

(آ) دید کلی از نتیجه

شکل آ-۵: مقایسه منحنی‌های خطا در مجموعه داده CINIC-10.

### آ-۱-۴ مجموعه داده FEMNIST

آ-۱-۴-۱ مقایسه نتایج در رویکرد کلاس‌بندی (FEMNISTclass)



images/chap5/result/FEMNISTclass/loss\_zoom.png

(ب) بزرگ‌نمایی شده بخش اصلی

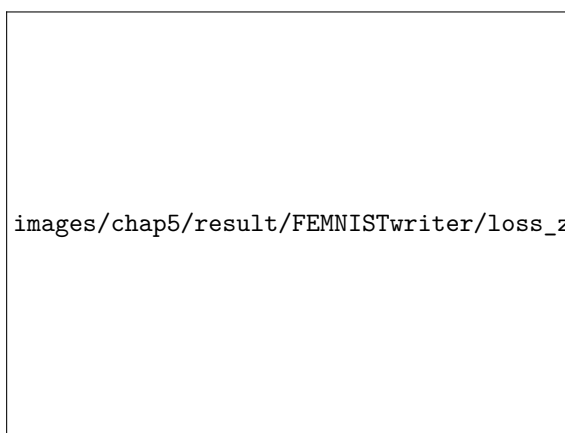


images/chap5/result/FEMNISTclass/loss\_base.png

(آ) دید کلی از نتیجه

شکل آ-۶: مقایسه منحنی‌های خطا در مجموعه داده FEMNISTclass.

#### آ-۱-۴-۲ مقایسه نتایج در رویکرد نویسندگان (FEMNISTwriter)



images/chap5/result/FEMNISTwriter/loss\_zoom.png

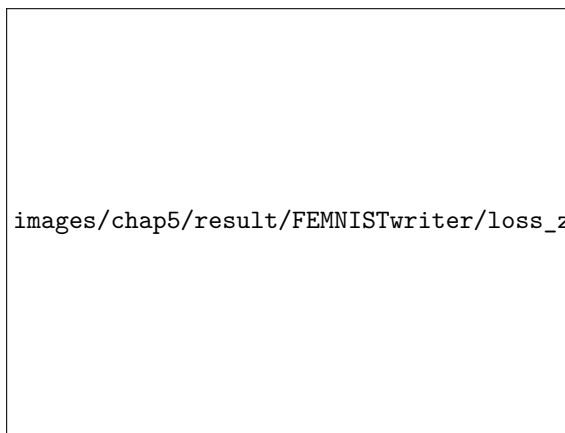
(ب) بزرگ‌نمایی شده بخش اصلی



images/chap5/result/FEMNISTwriter/loss\_base\_one.png

(آ) دید کلی از نتیجه

شکل آ-۷: مقایسه منحنی‌های خطا در یک اجرا بر روی مجموعه داده FEMNISTwriter.



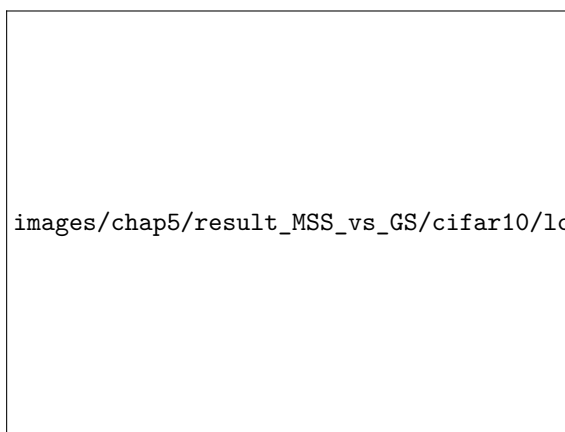
(ب) بزرگ‌نمایی شده بخش اصلی



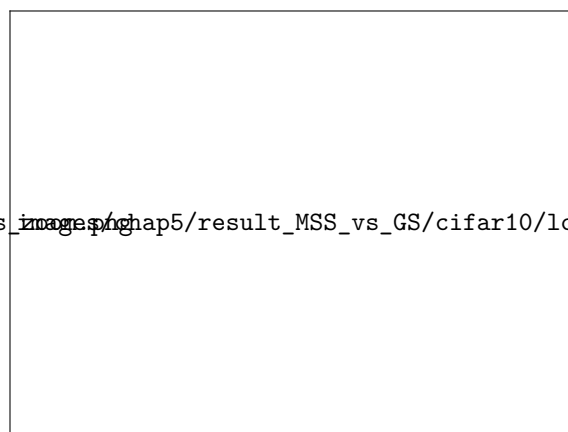
(آ) دید کلی از نتیجه

شکل آ-۸: مقایسه منحنی‌های خطا در میانگین پنج اجرا بر روی مجموعه داده FEMNISTwriter.

## آ-۲ مقایسه جابه‌جایی حریصانه با جابه‌جایی حداقل شباهت در روش SimFedSwap



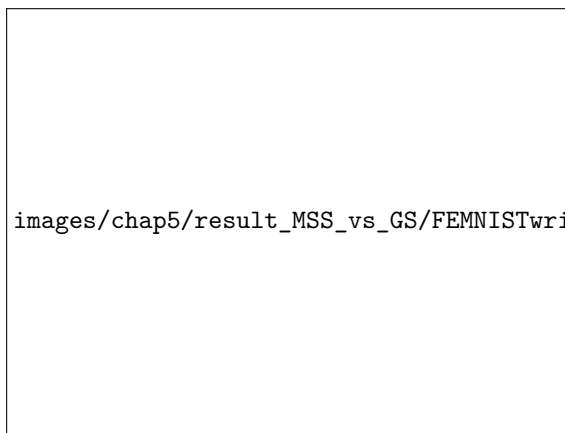
(ب) بزرگ‌نمایی شده بخش اصلی



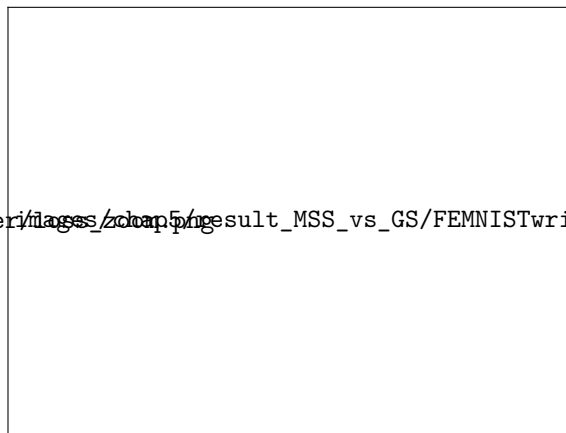
(آ) دید کلی از نتیجه

شکل آ-۹: مقایسه منحنی‌های خطا بین MSS و GS، در مجموعه داده CIFAR-10 با توزیع داده یکنواخت.





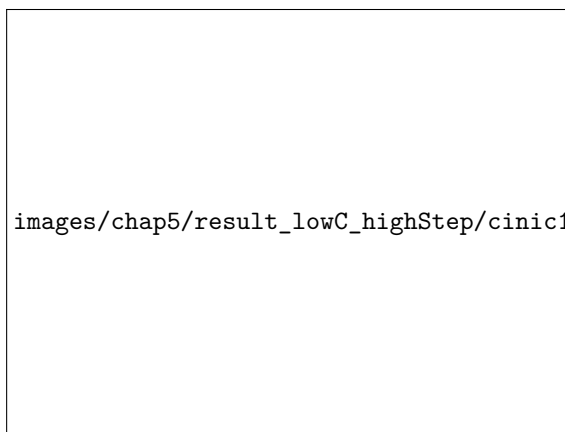
(ب) بزرگ‌نمایی شده بخش اصلی



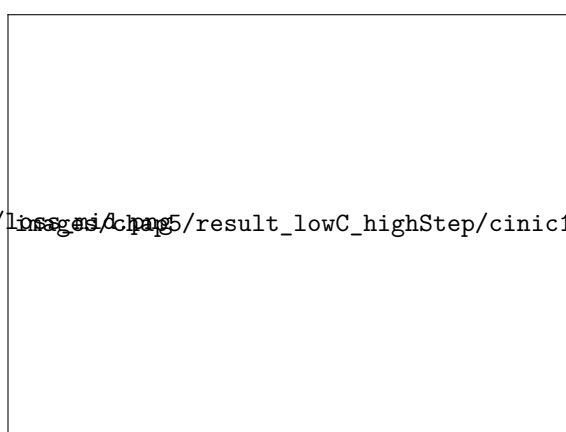
(آ) دید کلی از نتیجه

شکل آ-۱۰: مقایسه منحنی‌های خطا بین MSS و GS، در مجموعه داده FEMNISTwriter.

### آ-۳ تحلیل کاهش تعداد کاربران در هر دور و افزایش تعداد کل دورها در روش SimFedSwap



(ب) بزرگ‌نمایی شده بخش اصلی

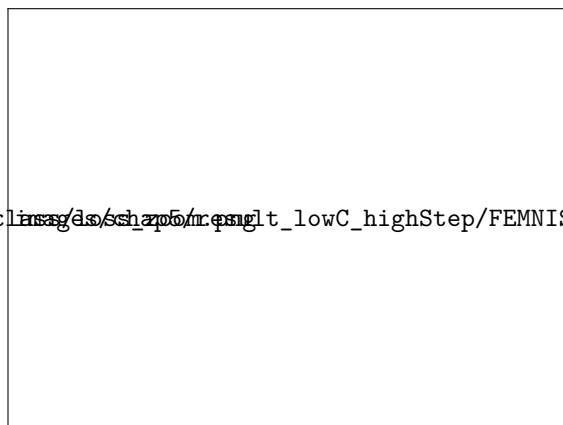


(آ) دید کلی از نتیجه

شکل آ-۱۱: مقایسه منحنی‌های خطا در مجموعه داده CINIC-10 با کاهش مشارکت کاربران و افزایش کل دورها.



(ب) بزرگ‌نمایی شده بخش اصلی



(آ) دید کلی از نتیجه

شکل آ-۱۲: مقایسه منحنی‌های خطا در مجموعه داده FEMNISTclass با کاهش مشارکت کاربران و افزایش کل دورها.

## مراجع

- [1] Elbir, Ahmet M, Coleri, Sinem, Papazafeiropoulos, Anastasios K, Kourtessis, Pandelis, and Chatzinotas, Symeon. A family of hybrid federated and centralized learning architectures in machine learning. *IEEE Transactions on Cognitive Communications and Networking*, 2022.
- [2] Zhou, Zhi, Chen, Xu, Li, En, Zeng, Liekang, Luo, Ke, and Zhang, Junshan. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE*, 107(8):1738–1762, 2019.
- [3] Ma, Xiaodong, Zhu, Jia, Lin, Zhihao, Chen, Shanxuan, and Qin, Yangjie. A state-of-the-art survey on solving non-iid data in federated learning. *Future Generation Computer Systems*, 135:244–258, 2022.
- [4] Smith, Virginia, Chiang, Chao-Kai, Sanjabi, Maziar, and Talwalkar, Ameet S. Federated multi-task learning. *Advances in neural information processing systems*, 30, 2017.
- [5] McMahan, Brendan, Ramage Daniel. Federated learning: Collaborative machine learning without centralized training data. <https://www.omron.com/global/en/technology/information/dcx>, 6 Apr 2017. [Accessed: 18 Apr 2024].
- [6] Li, Tian, Sahu, Anit Kumar, Talwalkar, Ameet, and Smith, Virginia. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020.
- [7] Talaei, Mahtab. Algorithm development and performance analysis for adaptive differential privacy in federated learning, 21 Aug 2022.
- [8] Rieke, Nicola. What is federated learning? <https://blogs.nvidia.com/blog/what-is-federated-learning/>, 13 Oct 2019. [Accessed: 10 Apr 2024].
- [9] Chiu, Te-Chuan, Shih, Yuan-Yao, Pang, Ai-Chun, Wang, Chieh-Sheng, Weng, Wei, and Chou, Chun-Ting. Semisupervised distributed learning with non-iid data for aiot service platform. *IEEE Internet of Things Journal*, 7(10):9266–9277, 2020.
- [10] Gavrilova, Yulia. Artificial intelligence vs. machine learning vs. deep learning: Essentials. <https://serokell.io/blog/ai-ml-dl-difference>, 8th Apr 2020. [Accessed: 5 Apr 2024].
- [11] Goehner, AIT. Deep learning, welcome to the future! <https://www.ait.de/en/deep-learning/>. [Accessed: 12 May 2024].

- [12] LeCun, Yann, Bottou, Léon, Bengio, Yoshua, and Haffner, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [13] McMahan, Brendan, Moore, Eider, Ramage, Daniel, Hampson, Seth, and y Arcas, Blaise Aguera. Communication-efficient learning of deep networks from decentralized data. in *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.
- [14] Hellström, Henrik, da Silva Jr au2, José Mairton B., Amiri, Mohammad Mohammadi, Chen, Mingzhe, Fodor, Viktoria, Poor, H. Vincent, and Fischione, Carlo. Wireless for machine learning, 2022.
- [15] Wang, Hongyi, Sievert, Scott, Liu, Shengchao, Charles, Zachary, Papailiopoulos, Dimitris, and Wright, Stephen. Atomo: Communication-efficient learning via atomic sparsification. *Advances in neural information processing systems*, 31, 2018.
- [16] Konečný, Jakub, McMahan, H Brendan, Yu, Felix X, Richtárik, Peter, Suresh, Ananda Theertha, and Bacon, Dave. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.
- [17] Fang, Chen, Guo, Yuanbo, Hu, Yongjin, Ma, Bowen, Feng, Li, and Yin, Anqi. Privacy-preserving and communication-efficient federated learning in internet of things. *Computers & Security*, 103:102199, 2021.
- [18] Konečný, Jakub, McMahan, Brendan, and Ramage, Daniel. Federated optimization: Distributed optimization beyond the datacenter. *arXiv preprint arXiv:1511.03575*, 2015.
- [19] Hasan, Jahid. Security and privacy issues of federated learning. *arXiv preprint arXiv:2307.12181*, 2023.
- [20] Yin, Xuefei, Zhu, Yanming, and Hu, Jiankun. A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions. *ACM Computing Surveys (CSUR)*, 54(6):1–36, 2021.
- [21] Ioffe, Sergey and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. in *International conference on machine learning*, pp. 448–456. pmlr, 2015.
- [22] Li, Tian, Sahu, Anit Kumar, Zaheer, Manzil, Sanjabi, Maziar, Talwalkar, Ameet, and Smith, Virginia. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.
- [23] Zhao, Yue, Li, Meng, Lai, Liangzhen, Suda, Naveen, Civin, Damon, and Chandra, Vikas. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.
- [24] Collins, Liam, Hassani, Hamed, Mokhtari, Aryan, and Shakkottai, Sanjay. Exploiting shared representations for personalized federated learning. in *International conference on machine learning*, pp. 2089–2099. PMLR, 2021.
- [25] Jeong, Eunjeong, Oh, Seungeun, Kim, Hyesung, Park, Jihong, Bennis, Mehdi, and Kim, Seong-Lyun. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*, 2018.
- [26] Taïk, Afaf, Moudoud, Hajar, and Cherkaoui, Soumaya. Data-quality based scheduling for federated edge learning. in *2021 IEEE 46th Conference on Local Computer Networks (LCN)*, pp. 17–23. IEEE, 2021.
- [27] Zeng, Yan, Wang, Xin, Yuan, Junfeng, Zhang, Jilin, and Wan, Jian. Local epochs inefficiency caused by device heterogeneity in federated learning. *Wireless Communications & Mobile Computing*, 2022.
- [28] Sannara, EK, Portet, François, Lalanda, Philippe, and German, VEGA. A federated learning aggregation algorithm for pervasive computing: Evaluation and comparison. in *2021 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pp. 1–10. IEEE, 2021.
- [29] Qin, Yang and Kondo, Masaaki. Mlmg: Multi-local and multi-global model aggregation for federated learning. in *2021 IEEE international conference on pervasive computing and communications workshops and other affiliated events (PerCom Workshops)*, pp. 565–571. IEEE, 2021.

- [30] Ma, Qianpiao, Xu, Yang, Xu, Hongli, Jiang, Zhida, Huang, Liusheng, and Huang, He. Fedrsa: A semi-asynchronous federated learning mechanism in heterogeneous edge computing. *IEEE Journal on Selected Areas in Communications*, 39(12):3654–3672, 2021.
- [31] Li, Li, Duan, Moming, Liu, Duo, Zhang, Yu, Ren, Ao, Chen, Xianzhang, Tan, Yujuan, and Wang, Chengliang. Fedrsae: A novel self-adaptive federated learning framework in heterogeneous systems. in *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–10. IEEE, 2021.
- [32] Reddi, Sashank, Charles, Zachary, Zaheer, Manzil, Garrett, Zachary, Rush, Keith, Konečný, Jakub, Kumar, Sanjiv, and McMahan, H Brendan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.
- [33] Li, Xiaoli, Liu, Nan, Chen, Chuan, Zheng, Zibin, Li, Huizhong, and Yan, Qiang. Communication-efficient collaborative learning of geo-distributed jointcloud from heterogeneous datasets. in *2020 IEEE international conference on joint cloud computing*, pp. 22–29. IEEE, 2020.
- [34] Ghosh, Avishek, Hong, Justin, Yin, Dong, and Ramchandran, Kannan. Robust federated learning in a heterogeneous environment. *arXiv preprint arXiv:1906.06629*, 2019.
- [35] Itahara, Sohei, Nishio, Takayuki, Koda, Yusuke, Morikura, Masahiro, and Yamamoto, Koji. Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data. *IEEE Transactions on Mobile Computing*, 22(1):191–205, 2021.
- [36] Chai, Zheng, Ali, Ahsan, Zawad, Syed, Truex, Stacey, Anwar, Ali, Baracaldo, Nathalie, Zhou, Yi, Ludwig, Heiko, Yan, Feng, and Cheng, Yue. Tifl: A tier-based federated learning system. in *Proceedings of the 29th international symposium on high-performance parallel and distributed computing*, pp. 125–136, 2020.
- [37] Jiang, Yihan, Konečný, Jakub, Rush, Keith, and Kannan, Sreeram. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.
- [38] Zhang, Xinwei, Hong, Mingyi, Dhople, Sairaj, Yin, Wotao, and Liu, Yang. Fedpd: A federated learning framework with adaptivity to non-iid data. *IEEE Transactions on Signal Processing*, 69:6055–6070, 2021.
- [39] Corinzia, Luca, Beuret, Ami, and Buhmann, Joachim M. Variational federated multi-task learning. *arXiv preprint arXiv:1906.06268*, 2019.
- [40] Shoham, Neta, Avidor, Tomer, Keren, Aviv, Israel, Nadav, Benditkis, Daniel, Mor-Yosef, Liron, and Zeitak, Itai. Overcoming forgetting in federated learning on non-iid data. *arXiv preprint arXiv:1910.07796*, 2019.
- [41] Kornblith, Simon, Norouzi, Mohammad, Lee, Honglak, and Hinton, Geoffrey. Similarity of neural network representations revisited. in *International conference on machine learning*, pp. 3519–3529. PMLR, 2019.
- [42] Chen, An Mei, Lu, Haw-minn, and Hecht-Nielsen, Robert. On the geometry of feedforward neural network error surfaces. *Neural computation*, 5(6):910–927, 1993.
- [43] Orhan, A Emin and Pitkow, Xaq. Skip connections eliminate singularities. *arXiv preprint arXiv:1701.09175*, 2017.
- [44] LeCun, Yann, Kanter, Ido, and Solla, Sara. Second order properties of error surfaces: Learning time and generalization. *Advances in neural information processing systems*, 3, 1990.
- [45] Gretton, Arthur, Bousquet, Olivier, Smola, Alex, and Schölkopf, Bernhard. Measuring statistical dependence with hilbert-schmidt norms. in *International conference on algorithmic learning theory*, pp. 63–77. Springer, 2005.
- [46] Cortes, Corinna, Mohri, Mehryar, and Rostamizadeh, Afshin. Algorithms for learning kernels based on centered alignment. *The Journal of Machine Learning Research*, 13:795–828, 2012.
- [47] Cristianini, Nello, Shawe-Taylor, John, Elisseeff, Andre, and Kandola, Jaz. On kernel-target alignment. *Advances in neural information processing systems*, 14, 2001.

- [48] Cui, Tianyu, Kumar, Yogesh, Marttinen, Pekka, and Kaski, Samuel. Deconfounded representation similarity for comparison of neural networks. *Advances in Neural Information Processing Systems*, 35:19138–19151, 2022.
- [49] Holzer, Patrick, Jacob, Tania, and Kavane, Shubham. Dynamically weighted federated k-means. *arXiv preprint arXiv:2310.14858*, 2023.
- [50] Krizhevsky, Alex, Hinton, Geoffrey, et al. Learning multiple layers of features from tiny images. 2009.
- [51] Carr, Evan Marie. Cifar10 with fast.ai. <https://www.evanmarie.com/cifar10-with-fast-ai/>, 16 Nov 2022. [Accessed: 4 May 2024].
- [52] Darlow, Luke N, Crowley, Elliot J, Antoniou, Antreas, and Storkey, Amos J. Cinic-10 is not imagenet or cifar-10. *arXiv preprint arXiv:1810.03505*, 2018.
- [53] Caldas, Sebastian, Duddu, Sai Meher Karthik, Wu, Peter, Li, Tian, Konečný, Jakub, McMahan, H Brendan, Smith, Virginia, and Talwalkar, Ameet. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.