İZMİR KÂTİP ÇELEBİ ÜNİVERSİTESİ
2010
FEN BİLİMLERİ ENSTİTÜSÜ

Federe Öğrenme Algoritmaları, Açık Kaynak Çerçeve ve Kütüphaneler

Yazılım Mühendisliği Ana Bilim Dalı

Yüksek Lisans

Ömer Faruk Göçgün

ORCID 0000-0003-1957-0794

Tez Danışmanı: Prof. Dr. Femin Yalçın Küçükbayrak

Tez Eş-Danışmanı: Doç. Dr. Aytuğ Onan

Temmuz 2023

İzmir Kâtip Çelebi Üniversitesi Fen Bilimleri Enstitüsü öğrencisi **Ömer Faruk Göçgün** tarafından hazırlanan **Federe Öğrenme Algoritmaları, Açık Kaynak Çerçeve ve Kütüphaneler** başlıklı bu çalışma tarafımızca okunmuş olup, yapılan savunma sınavı sonucunda kapsam ve nitelik açısından başarılı bulunarak jürimiz tarafından YÜKSEK LİSANS tezi olarak kabul edilmiştir.

ONAYLAYANLAR:

Tez Danışmanı: **Prof. Dr. Femin YALÇIN KÜÇÜKBAYRAK**
İzmir Kâtip Çelebi Üniversitesi

Tez Eş-danışmanı: **Doç. Dr. Aytuğ ONAN**
İzmir Kâtip Çelebi Üniversitesi

Jüri Üyeleri: **Prof. Dr. Femin YALÇIN KÜÇÜKBAYRAK**
İzmir Kâtip Çelebi Üniversitesi

Doç. Dr. Sıla Övgü KORKUT UYSAL
İzmir Kâtip Çelebi Üniversitesi

Dr. Öğr. Üyesi Emin BORANDAĞ
Celal Bayar Üniversitesi

Savunma Tarihi: 31.07.2023

Yazarlık Beyanı

Ben, **Ömer Faruk Göçgün**, başlığı **Federe Öğrenme Algoritmaları, Açık Kaynak Çerçeve ve Kütüphaneler** olan bu tezimin ve tezin içinde sunulan bilgilerin şahsıma ait olduğunu beyan ederim. Ayrıca:

- Bu çalışmanın bütünü veya esası bu üniversitede Yüksek Lisans derecesi elde etmek üzere çalıştığım süre içinde gerçekleştirilmiştir.
- Daha önce bu tezin herhangi bir kısmı başka bir derece veya yeterlik almak üzere bu üniversiteye veya başka bir kuruma sunulduysa bu açık biçimde ifade edilmiştir.
- Başkalarının yayımlanmış çalışmalarına başvurduğum durumlarda bu çalışmalara açık biçimde atıfta bulundum.
- Başkalarının çalışmalarından alıntıladığımda kaynağı her zaman belirttim. Tezin bu alıntılar dışında kalan kısmı tümüyle benim kendi çalışmamdır.
- Kayda değer yardım aldığım bütün kaynaklara teşekkür ettim.
- Tezde başkalarıyla birlikte gerçekleştirilen çalışmalar varsa onların katkısını ve kendi yaptıklarımı tam olarak açıkladım.

Tarih: 31.07.2023

_____

Federe Öğrenme Algoritmaları, Açık Kaynak Çerçeve ve Kütüphaneler

Öz

Günümüzde dağıtık sistemler ve büyük veri, merkeziyetçi makine öğrenmesi/derin öğrenme modellerinde yapılan çalışmalar zaman ve donanım maliyeti gibi engellere sebebiyet vermektedir. Bu sebeple dağıtık sistemlerde çalışan yazılımların veya Nesnelerin İnterneti (IoT) cihazlarından toplanan verilerin tek bir merkezde model eğitimi veya bu verilerden sonuç elde edilmesi aynı zamanda gizlilik gibi sorunlara da sebebiyet vermektedir. Bu sorunların üstesinden gelebilmek için Google, veri sızıntısını önlerken birden çok cihaza dağıtılan veri kümelerine dayalı makine öğrenmesi modelleri oluşturmak amacıyla Federe Öğrenme'yi öne sürmüştür.

Makine öğrenmesine yönelik nispeten yeni sayılabilecek olan Federe Öğrenme, giderek küreselleşen bu dünyada veri gizliliği ve güvenliği sebebiyle giderek daha da önemli olacaktır. Federe öğrenmede, küresel bir model oluşturmak için iş birliği yapan cihazlar ve/veya yazılımlar yinelemeli bir şekilde kendi verisini doğrudan paylaşmadan küresel modelin doğruluk oranını giderek arttırmaktadır. Bu durum kurumları veya firmaları büyük kaynaklar ayırarak küresel bir modeli eğitme maliyetinden kurtarmakla beraber eğitim sürecini de hızlandırmaktadır.

Gizliliği koruyan veri paylaşımı özellikle sağlık, finans ve iletişim gibi sektörlerde model eğitimi için federe öğrenmeyi öne çıkartmaktadır. Federe öğrenmenin son yıllarda araştırma odağı haline gelmiş olması sadece yeni bir kavram olmasından dolayı değildir. Gizliliği koruyan kanunlar, nüfus ve teknoloji kullanımında ki artış ile

ideal bir çözüm olarak gelecekte kullanımının oldukça yaygınlaşacağı düşünülmektedir.

Bu tezde, özellikle açık kaynak federe öğrenme kütüphaneleri üzerine incelemeler yapılmış ve değerlendirme sonuçları sunulmuştur.

Anahtar Sözcükler: Federe Öğrenme, Makine Öğrenmesi, Derin Öğrenme, Flower Federe Öğrenme Çerçevesi

Federated Learning Algorithms, Open Source Frameworks and Libraries

Abstract

In today's world, studies on distributed systems and big data in centralized machine learning/deep learning models are leading to obstacles such as time and hardware costs. As a result, software running on distributed systems or data collected from Internet of Things (IoT) devices face challenges when it comes to centralized model training or deriving results from this data, along with issues like privacy. To overcome these problems, Google has proposed Federated Learning, which involves creating machine learning models based on data sets distributed across multiple devices while preventing data leakage.

Federated Learning, a relatively new concept in the field of machine learning, will become increasingly important in this globalized world due to concerns about data privacy and security. In Federated Learning, collaborating devices and/or software iteratively contribute to building a global model without directly sharing their data, gradually improving the accuracy of the global model. This approach not only saves institutions or companies from the cost of training a global model by allocating significant resources, but also accelerates the training process.

Privacy-preserving data sharing particularly highlights Federated Learning for model training in sectors such as healthcare, finance, and communication. The growing focus on Federated Learning in recent years is not only due to it being a new concept. With

the rise of privacy-protecting regulations, population growth, and increased technology usage, it is believed that Federated Learning will become more widespread in the future as an ideal solution.

In this thesis, a specific focus is placed on examining and evaluating open-source Federated Learning libraries, and the results of these assessments are presented.

Keywords: Federated learning, machine learning, deep learning, Flower: Federated learning framework

Bu tezi, sevgili annem ve hocam Doç. Dr. Şeniz Erçağlar ile sevgili eşim Pelin Göçgün'e ithaf ediyorum.

Teşekkür

Bu çalışma süresince değerli bilgi ve tecrübelerinden yararlandığım danışman hocam Prof. Dr. Femin YALÇIN KÜÇÜKBAYRAK ile eş danışman hocam Doç. Dr. Aytuğ ONAN'a ve yüksek lisans yapmam konusunda beni teşvik eden hocam Dr. İsmail DÜŞMEZ'e tüm içtenliğimle teşekkür ederim.

İçindekiler

Yazarlık Beyanı iii

Öz iv

Abstract vi

Teşekkür ix

Şekiller Listesi xiii

Tablolar Listesi xiv

Kısaltmalar Listesi xv

Semboller Listesi xv

1. Giriş 1

2. Literatür Taraması 3

3. Federe Öğrenme 5

3.1 Federe Öğrenme Türleri 6

3.1.1 Yatay Federe Öğrenme 6

3.1.2 Dikey Federe Öğrenme 7

3.1.3 Federe Transfer Öğrenme 10

3.2 Federe Öğrenme Ayarları 11

3.2.1 Silolar Arası (Cross-Silo) Federe Öğrenme 11

3.2.2 Cihazlar Arası (Cross-Device) Federe Öğrenme 11

3.3 Federe Öğrenme Algoritmaları 12

3.3.1 FedAvg (Federe Ortalama) Algoritması 13
3.3.2 FedDANE Algoritması 13
3.3.3 FedNS (Federated Node Selection) Algoritması 14
3.3.4 FedProx Algoritması 15
3.3.5 FedFA Algoritması 16
3.3.6 FedBN Algoritması 17
3.3.7 FedMA Algoritması 17
3.3.8 FedOpt Algoritması 18
3.3.9 FedCurv Algoritması 18
3.4 Veri Gizliliği 18
3.5 Açık Kaynak Federe Öğrenme Kütüphaneleri 19
3.5.1 FedML Çerçevesi 19
3.5.2 OpenFL Çerçevesi 20
3.5.3 TensorFlow Federe Öğrenme Çerçevesi 20
3.5.4 Nvidia Clara Çerçevesi 21
3.5.5 PySyft ve PyGrid Kütüphaneleri 22
3.5.6 OpenMind Web ve Mobil Kütüphaneleri 22
3.5.7 WebFed Çerçevesi 22
3.5.8 Leaf: Federe öğrenme için açık kaynak veri kümeleri 24
3.5.9 Hugging Face Transformers Kütüphanesi 24
3.5.10 Flower Çerçevesi 24
3.5.10.1 Flower Başlangıç Paketleri 26
4. Materyal ve Metot 28

4.1 Materyal 28

4.1.1 CIFAR-10 Veri Kümesi ve Özellikleri 28

4.1.2 Deneysel Ortam 31

Tartışma ve Sonuç 32

Kaynaklar 35

Ekler 42

EK A Tezden Üretilmiş Yayınlar 43

Özgeçmiş 44

Şekiller Listesi

Şekil 3.1 Federe öğrenme yapısı 5

Şekil 3.2 Yatay federe öğrenme örneği 7

Şekil 3.3: Dikey Federe Öğrenme Örneği 8

Şekil 3.4: Dikey Federe Öğrenme süreci 9

Şekil 3.5 Transfer Federe Öğrenme örneği 10

Şekil 3.6: Transfer Federe Öğrenme örnek kümeleri grafiği 10

Şekil 3.7: Silolarar arası federe öğrenme örneği 11

Şekil 3.8: Cihazlar Arası Federe Öğrenme 12

Şekil 3.9: FedAvg federe öğrenme çerçevesi 13

Şekil 3.10: WebFed Mimarisinin Gösterimi 23

Şekil 3.11: Flower çekirdek mimarisi 25

Şekil 3.12 Sol: McMahan deneyi sonuçları, Sağ: Flower deneyi sonuçları..... 27

Şekil 4.1: CIFAR-10 veriseti örnek resimler 30

Şekil 5.1: 100 istemcinin 500 iletişim turu doğruluk oranları. 32

Şekil 5.2: 100 istemcinin 2000 iletişim turu doğruluk oranları. 33

Şekil 5.3: 500 istemcinin 4000 iletişim turu doğruluk oranları 33

Tablolar Listesi

Tablo 3.1: FashionM-NIST veri kümesiyle yapılan deneysel sonuçlar 14

Tablo 3.2: CIFAR-10 veri kümesiyle yapılan deneysel sonuçlar 15

Tablo 3.3: TinyImageNet veri kümesiyle yapılan deneysel sonuçlar 15

Tablo 3.4 : FedFA algoritmasının FedAvg ve FedProx deneysel sonuçları 16

Tablo 3.5: Yapılan deneysel sonuçlara göre FedBN'nin doğruluk oranları 17

Tablo 3.6:Yapılan deneysel sonuçlara göre FedMA'nin doğruluk oranları 18

Tablo 4.1: Yapılan deneysel çalışmada kullanılan parametreler……...29

Tablo 5.1: Yapılan deneysel sonuçların karışlaıtırılmalı doğruluk oranları……...33

Kısaltmalar Listesi

FBE	Fen Bilimleri Enstitüsü
İKÇÜ	İzmir Kâtip Çelebi Üniversitesi
ORCID	Open Researcher and Contributor ID
FÖ	Federe Öğrenme
MÖ	Makine Öğrenmesi
FL	Federated Learning (Federe Öğrenme)
ML	Machine Learning (Makine Öğrenmesi)
FedAvg	Federated Average
WebFed	Federated Learning Framework Based on Web Browser
FedNS	Federated Node Selection
FedFA	FedFA: Federated Learning with Feature Alignment for Heterogeneous Data
FedProx	FedProx, heterogeneity in federated networks
FedBN	Federated Learning Via Batch Normalization
FedMA	Federated Learning with Matched Averaging
IOT	Nesnelerin İnterneti (Internet of things)
FedCurv	Federated Curvature
FedOpt	Adaptive Federated Optimization
IID	Independent and identically distributed (Bağımsız ve özdeş dağılmış)

Bölüm 1

Giriş

Günümüzde dağıtık sistemler ve büyük veri, merkeziyetçi makine öğrenmesi/derin öğrenme modellerinde zaman ve donanım maliyeti gibi engellere sebebiyet vermektedir. Bu sebeple dağıtık sistemlerde çalışan yazılımların veya Nesnelerin İnterneti (IoT) cihazlarından toplanan verilerin tek bir merkezde model eğitimi veya bu verilerden sonuç elde edilmesi aynı zamanda gizlilik gibi sorunlara da sebebiyet vermektedir.

Geleneksel Makine Öğrenmesi (MÖ/ML) kullanımda, tüm veriler bir veri merkezinde toplanır. Toplanan verilerle model daha sonra güçlü sunucularda merkezi olarak eğitilir. Ancak, bu veri toplama süreci genellikle mahremiyete zarar verir. Birçok kullanıcı özel verilerini şirketlerle paylaşmak istemez. Bazı durumlarda makine öğrenmesini kullanmak zordur, ör. tıbbi veriler (sağlık), finansal veriler(bankacılık) ve konumsal veriler(konuma dayalı tavsiye mobil uygulamalarda). Gizlilik bir endişe kaynağı olmadığında bile, veri toplama zorunluluğunun makul olmadığı durumlar olabilmektedir. Örneğin, sürücüsüz arabalar çok fazla veri üretir ve hepsini bir sunucuya gönderebilir. Daha yaygın bir örnek ele alacak olursak, kullanıcı cep telefonlarıyla etkileşimde bulunarak büyük miktarda veri üretir. Örneğin, dil modelleri konuşma tanımayı ve metin girişini iyileştirebilir ve görüntü modelleri otomatik olarak iyi fotoğrafları seçebilir [1]. Bu veriler genellikle doğası gereği son derece özeldir ve bütünüyle paylaşılmamalıdır.

Federe öğrenme, cep telefonlarının tüm eğitim verilerini cihazın kendisinde tutarken ortak bir tahmin modelini iş birliği içinde öğrenmesini sağlar ve makine öğrenmesi yapma yeteneği ile verileri sunucuda depolama ihtiyacından ayırır [1]. Böylece veri paylaşımı yapılmadan sadece eğitilen model parametre sunucusuna gönderilerek aynı zamanda gizlilik de sağlanmış olur. Bu çalışmada seçilen federe öğrenme

algoritmaları, açık kaynak çerçeve ve kütüphanelerinin karşılaştırılması ortaya koyulmuştur.

Bu çalışmada, Federe Öğrenme (FÖ) olarak bilinen yeni bir yaklaşıma genel bir bakış sunulmaktadır. Federe öğrenme konusundaki mevcut çalışmaları inceliyor ve kapsamlı, güvenli bir birleşik öğrenme çerçevesi için tanımlar, sınıflandırmalar ve uygulamalar önerilmiştir. Bu çalışma da açıklanan, açık kaynak federe öğrenme çerçeveleri ve kütüphanelerin hangilerini seçeceğine karar vermek isteyen araştırmacıların ve sektör profesyonellerinin zaman kazanmasına ve daha verimli çalışmalarına yardımcı olacaktır.

Bölüm 2

Literatür Taraması

Federe Öğrenme terimi ilk olarak McMahan vd. tarafından 2016 yılında "Communication-Efficient Learning of Deep Networks from Decentralized Data" makalesinde tanımlanmıştır [2]. Akabinde, Google AI tarafından 2017 yılında "Federated Learning: Collaborative Machine Learning without Centralized Training Data" blog yayınında tanıtılmıştır [1]. Makalede, standart makine öğrenmesi yaklaşımlarının eğitim verilerini bir makinede veya bir veri merkezinde barındırmayı gerektirdiği ve hizmetlerini daha iyi hale getirmek adına bu verileri işlemek için en güvenli ve sağlam bulut altyapılarından birini oluşturduğu vurgulanmıştır.

Google'ın federe öğrenmeyi öne sürmekteki ana fikri, veri sızıntısını önlerken birden çok cihaza dağıtılan veri kümelerine dayalı makine öğrenmesi modelleri oluşturmaktır. Güncel çalışmalar, birleştirilmiş öğrenmede istatistiksel zorlukların üstesinden gelmeye ve güvenliği artırmaya odaklanmıştır [1].

2021'de Lian vd. tarafından, tarayıcının özelliklerinden yararlarak yerel diferansiyel gizlilik mekanizması aracılığıyla gizlilik korumasını geliştiren yeni bir web tabanlı federe öğrenme yöntemi olarak WebFed önerilmiştir [4].

2021 yılında Lo vd., federe öğrenme sistemi tasarlamanın makine öğrenmesi bilgisinden ayrı olarak yazılım sistemi tasarımı gerektirdiğini öne sürmüştür. Söz konusu çalışmada federe öğrenme sistemlerinin tasarımı için mimari modeller sunulmuştur.

Bazı çalışmalarda bağımsız olmayan ve özdeş dağılmış (non-IID) ile bağımsız ve özdeş dağılmış (IID) olarak ayrıştırılmış deneyler ve veri kümeleri kullanılmıştır. İstatistik ve olasılık teorisine göre, her bir rastgele değişken diğerleriyle aynı olasılık

dağılımına sahipse ve hepsi karşılıklı olarak bağımsızsa, bağımsız ve özdeş dağılmış rastgele değişkendir. Bu tür veri kümelerinde IID (independent and identically distributed) kısaltması başlığa eklenmektedir. Bu veri kümesinin kendisindeki bir özellik olmayıp, söz konusu çalışmalarda verinin bu şekilde dağıtıldığı varsayılır [6].

Yapay sinir ağlarında, eğitim esnasında her katman bir önceki katmanın çıktısıyla öğrenir, bu ise eğitim süresini uzatır. Yığın normalleştirme (Batch Normalization) ise her bir katman önceki katmanın öğrenmesini beklemek zorunda kalmaz ve aynı anda öğrenim sürecine devam eder [7].

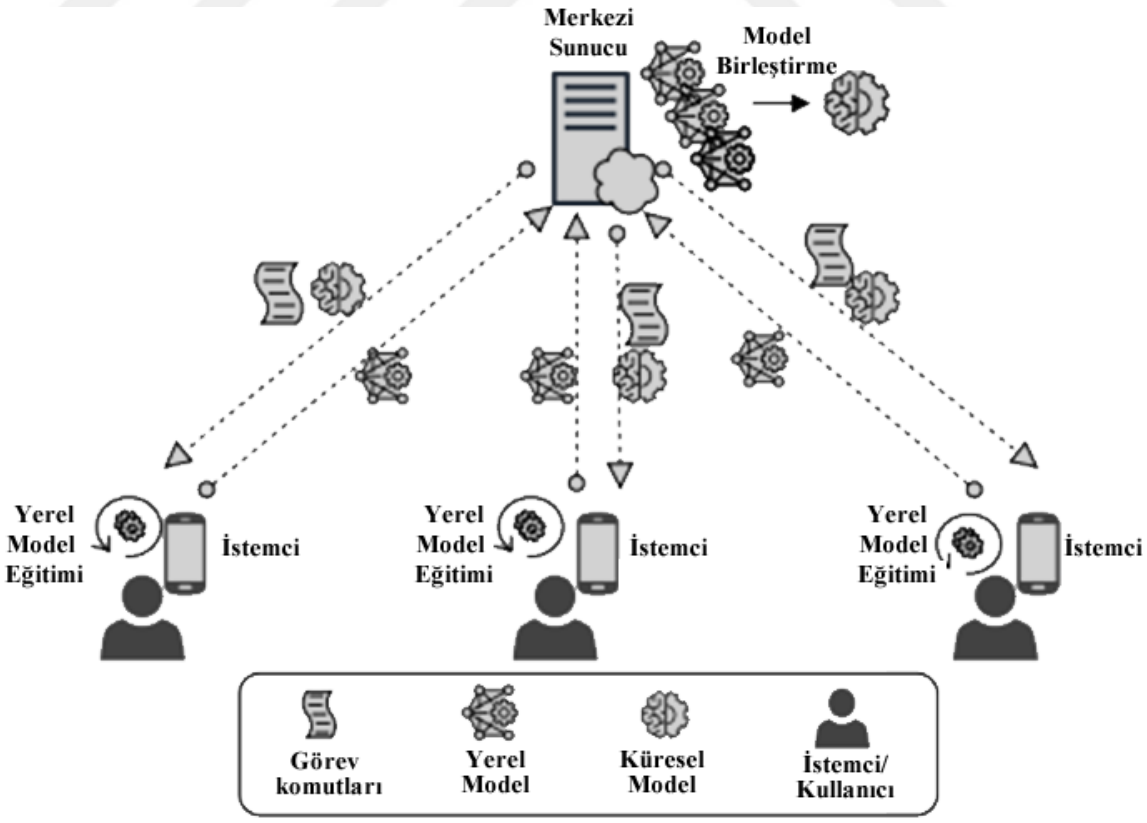
Eğitim için kullanılan veri kümesindeki her bir örnek için kayıp işlevlerinin ortalama değeri Stochastic Gradient Descent (SGD) olarak ifade edilir [8]. Derin öğrenmede SGD, her bir yineleme maliyetini azaltır.

Bölüm 3

Federe Öğrenme

Bu bölümde federe öğrenme, türleri, popüler algoritmalar, açık kaynak çerçeveler ve kütüphaneler tanıtılmıştır. Öncelikle federe öğrenme özellikleri ve türleri açıklanmıştır. Akabinde, federe öğrenme algoritmaları tanıtılmıştır. Son olarak, kullanılan açık kaynak çerçeveler ve kütüphaneler ile örnek bir çalışma ortaya konulmuştur.

Federe öğrenme, birbirinden farklı cihaz veya uygulamadan kullanıcı verilerinin paylaşılmadan sunucu veya sunucular üzerinden sadece model ağırlıklarını alarak algoritma eğiten bir makine öğrenmesi tekniğidir (Şekil 3.1).



Şekil 3.1 Federe öğrenme yapısı [35]

Federe öğrenme, veri kümelerinin çok sayıda veri üreticisine (yani cihazlar ve/veya sistemlere) dağıldığı gizliliği koruyan model geliştirmeye olanak tanır. Bu veri üreticileri, farklı gizlilik teknikleriyle öğrenme modellerini eğitir, model güncellemelerini küresel modelin eğitilmesi için merkezi sistemlerle paylaşır.

2015 yılında Shokri ve Shmatikov, katılımcı modellerinin bağımsız olarak eğitildiği ve yalnızca parametre güncellemelerinin alt kümelerini paylaştığı, iş birliğine dayalı bir derin öğrenme planı önermiştir [9]. 2017'de Google, kullanıcı verilerini kullanıcı cihazından almadan işbirlikçi olarak makine öğrenmesi modeli olan federe öğrenmeyi tanıtmıştır [1] ve makine öğrenmesi süreçlerini veri depolama ihtiyacından ayırmıştır. Google, kullanıma sunulduğundan bu yana akıllı metin seçme [10], kelime tahmini [11], emoji önerme [11], kelime dağarcığındaki kelime keşfi [11] de dahil olmak üzere birçok özelliği desteklemek için federe öğrenmenin gücünden faydalanmaktadır.

Federe öğrenme, Model Merkezli ve Veri Merkezli olmak üzere iki temel kategoriye ayrılabilir. Model merkezli federe öğrenmede, her istemci kendi verilerini üretir ve depolar, diğer istemcilerin verilerini okuyamaz. Veri merkezli federe öğrenmede ise, verilerin sahibi veya sahipleri bu verileri paylaşmadan üzerinde modeller oluşturulması için harici kuruluşlara erişim sağlayabilir. Örneğin, bankalar kredi kartı dolandırıcılıkları için kendi verilerini paylaşmadan birçok farklı cihazda elde edilen veriler doğrultusunda ortak bir model eğitmek için iş birliği yapabilir.

3.1 Federe Öğrenme Türleri

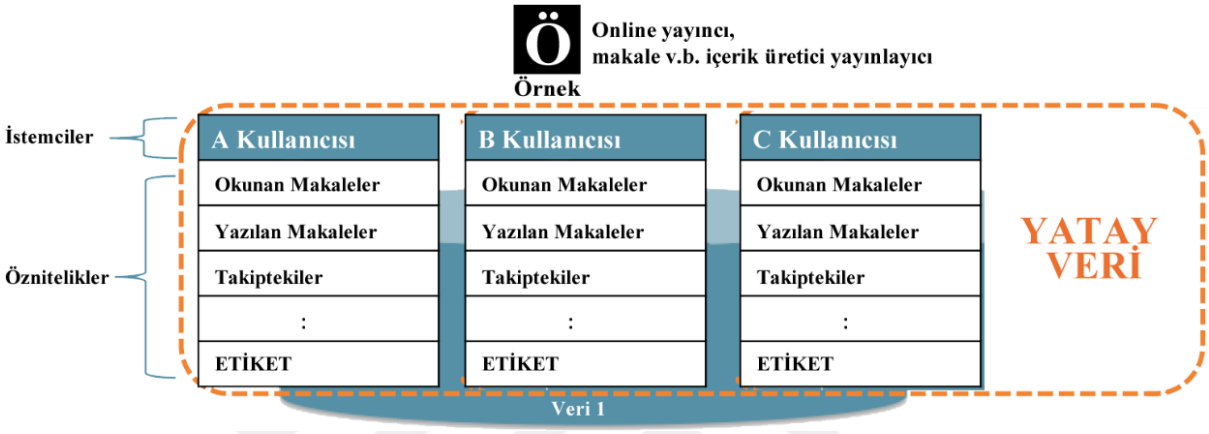
2019'da Yang vd., federe öğrenmeyi üç ayrı türde gerçekleştirimini önermiştir. Bunlar Yatay, Dikey ve Transfer öğrenme türleridir [3]. Tezin bu bölümünde federe öğrenme türleri detaylı bir şekilde açıklanacaktır.

3.1.1 Yatay Federe Öğrenme

Yatay Federe Öğrenme veya Örnek Tabanlı Federe Öğrenme, veri kümelerinin aynı özellik alanını paylaştığı ancak örneklerde farklı olduğu senaryolarda kullanılır (Şekil 3.2). Örneğin, aynı bölgedeki iki farklı hastanede göğüs hastaları farklı olabilir ve

hastaların kesişim kümesi çok küçüktür ancak işleri çok benzerdir. Bu sebeple özellik alanları aynıdır. Bu tür durumlarda, yatay federe öğrenme modeli oluşturulabilir [3].

2017'de Google, Android telefon modeli güncellemeleri için yatay bir birleşik öğrenme çözümü önerdmiştir [1]. Bu öneride, Android işletim sistemine sahip mobil cihaz (telefon, tablet v.b.) kullanan tek bir kullanıcı, model parametrelerini yerel olarak günceller ve parametreleri Android bulutuna yükler, böylece merkezi modeli diğer veri sahipleri ile birlikte eğitir.



Şekil 3.2 Yatay federe öğrenme örneği [12]

Böyle bir sistemin eğitim süreci genellikle şu dört adımdan oluşur [13]:

Adım 1: Katılımcılar eğitim gradyanlarını yerel olarak hesaplar, şifreleme, diferansiyel gizlilik veya gizli paylaşım teknikleriyle bir gradyan seçimini maskeler ve maskelenmiş olarak sonuçları sunucuya gönderir.

Adım 2: Sunucu, herhangi bir katılımcı hakkında bilgi öğrenmeden güvenli bir şekilde anonimleştirilmiş bilgileri bir araya getirir.

Adım 3 Sunucu, toplanan sonuçları katılımcılara geri gönderir.

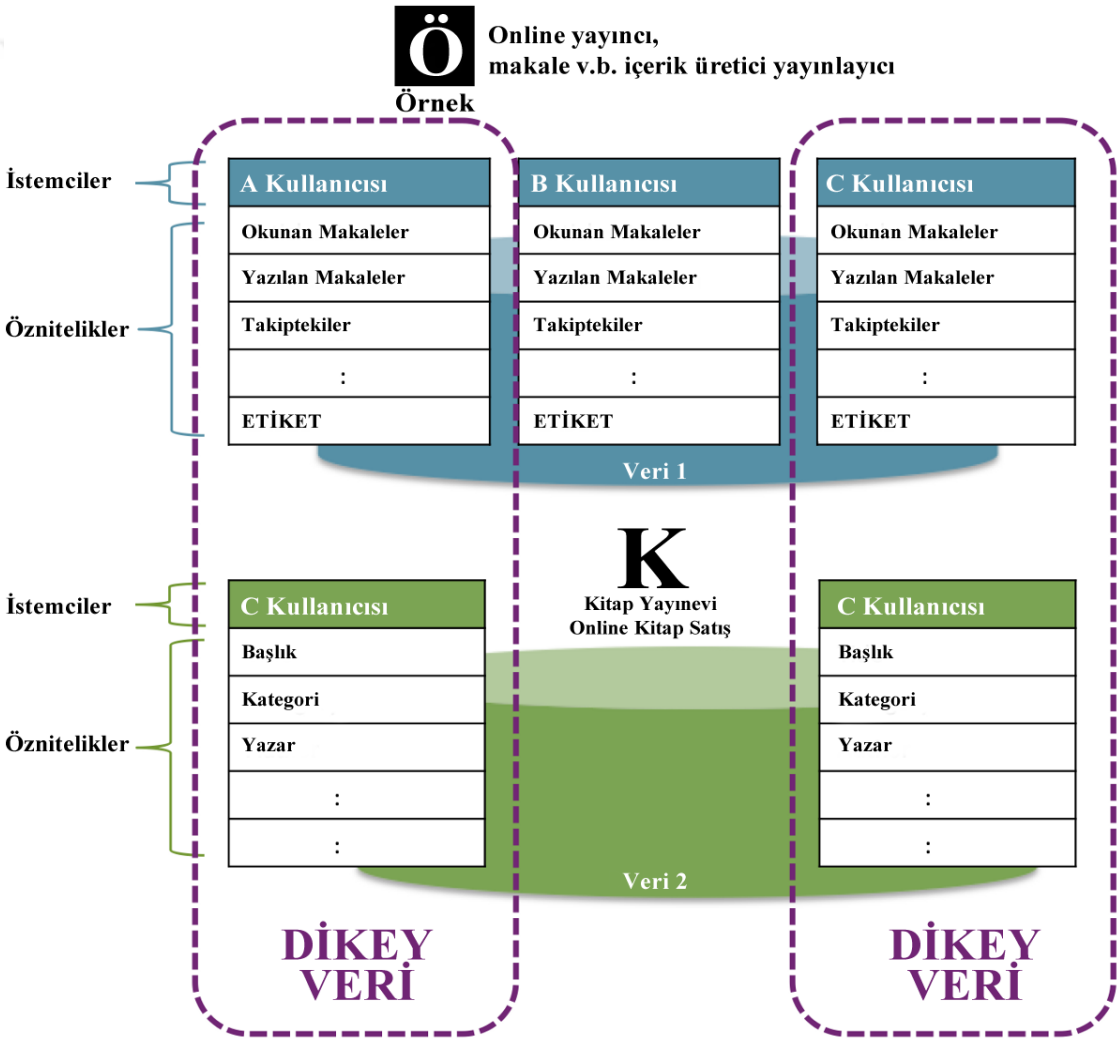
Adım 4: Katılımcılar, şifresi çözülmüş gradyanlarla ilgili modellerini günceller.

3.1.2 Dikey Federe Öğrenme

Dikey Federe Öğrenme veya Özellik Tabanlı Federe Öğrenme, iki veri kümesinin aynı kullanıcıları paylaştığı ancak özellik alanının farklı olduğu senaryolarda kullanılır (Şekil 3.3).

Örneğin, biri banka diğeri e-ticaret şirketi olan iki farklı şirket ele alındığında, kullanıcı kümelerinin büyük oranda kesişmesi muhtemeldir ve bu nedenle kullanıcı alanlarının kesişim kümesi büyüktür. Ancak banka, kullanıcının gelir ve harcamaları ile kredilendirme bilgisini tutmakta iken e-ticaret firması, kullanıcının arama ve satın alma geçmişini tuttuğu için özellik alanları çok farklıdır. Her iki tarafın da kullanıcı ve ürün bilgilerine dayalı, müşteri gelir durumuna göre ürün satın alma işlemlerine bir tahmin modeli oluşturmak için dikey federe öğrenme yöntemi kullanılır [3].

Dikey olarak birleşik öğrenme, örnek senaryodaki gibi farklı özellikleri toplama ve her iki taraftan da verilerle gizliliği koruyan bir şekilde modeli hesaplama sürecidir [3].



Şekil 3.3: Dikey federe öğrenme örneği [12]

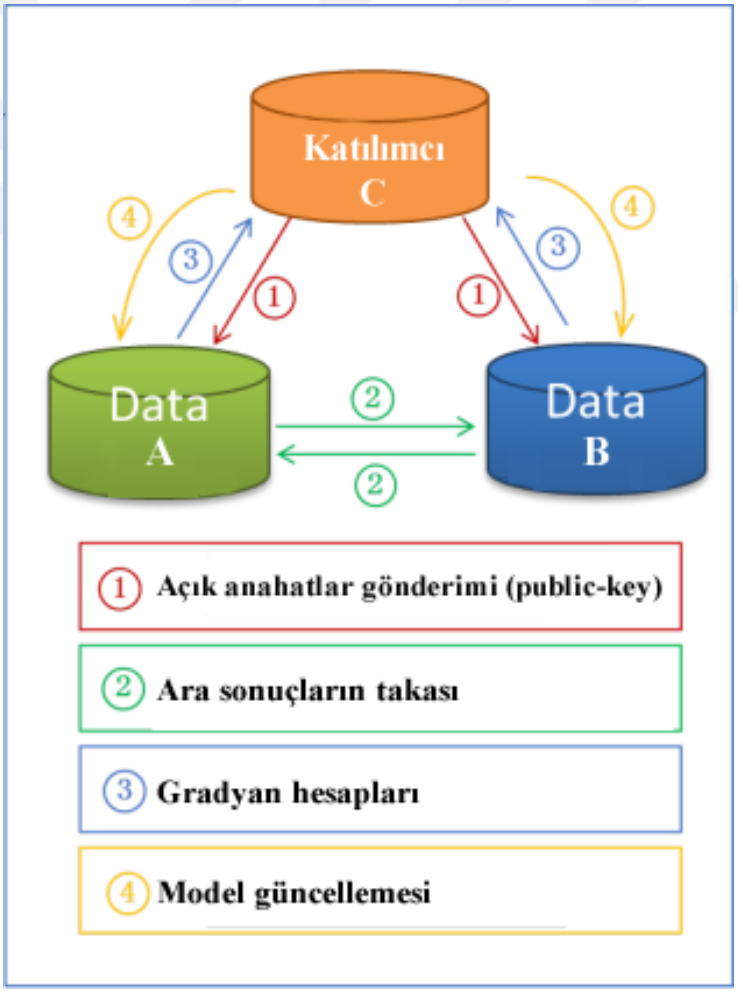
Böyle bir sistemin eğitim süreci genellikle şu dört adımdan oluşur [13] :

Adım 1: Ortak çalışan C, şifreleme eşlerini oluşturur, genel anahtarı A ve B'ye (Şekil 4) gönderir.

Adım 2: A ve B, 9radian ve kayıp hesaplamaları için ara sonuçları şifreler ve değiştirir.

Adım 3: A ve B sırasıyla şifreli gradyanları hesaplar ve ek maske ekler. Daha sonra A ve B şifrelenmiş değerleri C'ye gönderir.

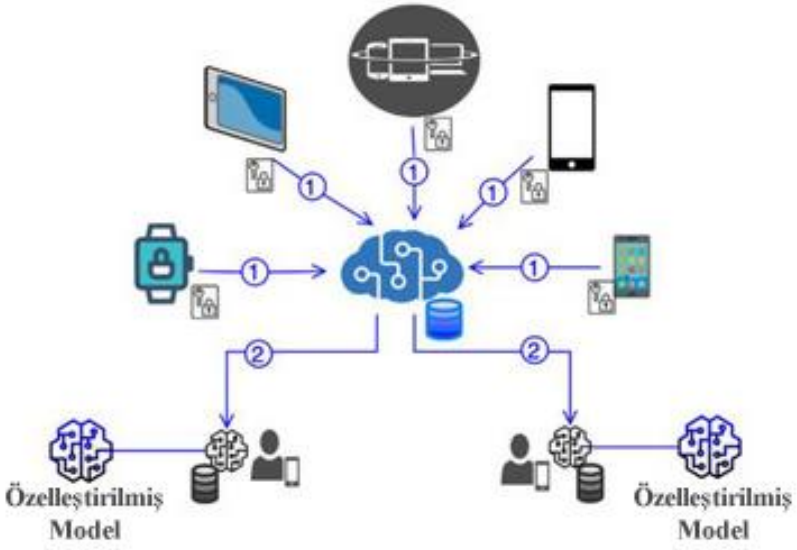
Adım 4: C şifresini çözer ve şifresi çözülmüş gradyanları A ve B'ye geri gönderir. A ve B gradyanların maskesini kaldırır, model parametrelerini buna göre günceller.



Şekil 3.4: Dikey federe öğrenme süreci [13]

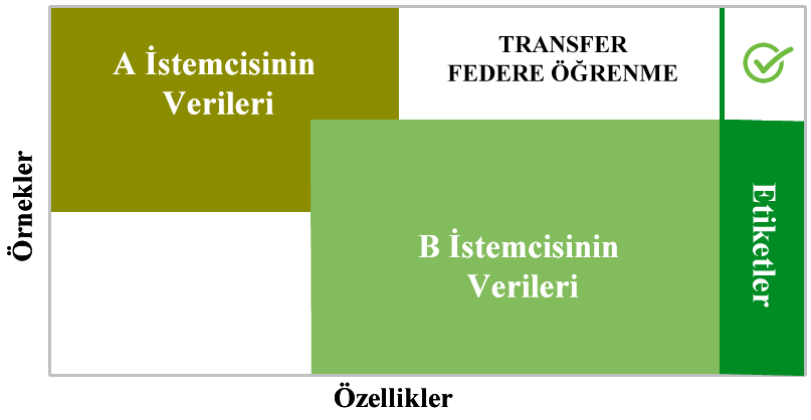
3.1.3 Federe Transfer Öğrenme

Federe Transfer Öğrenme, iki veri kümesinin yalnızca örneklerde değil, aynı zamanda özellik alanında da farklılık gösterdiği senaryolar için kullanılır. Örneğin, iki farklı ülkede bir banka ve bir e-ticaret şirketi ele alındığında, bu iki şirketin kullanıcılarının kesişim kümesi oldukça küçük olacaktır. Yani, her iki tarafın özellik alanının yalnızca küçük bir kısmı çakışmaktadır. Bu durumda federe transfer öğrenme kullanılmalıdır (Şekil 3.5).



Şekil 3.5 Transfer federe öğrenme örneği [14]

Federe transfer öğrenme, sınırlı ortak örnek kümeleri kullanılarak iki özellik uzayı arasındaki ortak bir temsil öğrenilmekte ve daha sonra yalnızca tek taraflı özelliklere sahip örnekler için tahminler elde etmek için uygulanmaktadır (Şekil 3.6) [3].



Şekil 3.6: Transfer Federe Öğrenme örnek kümeleri grafiği [15]

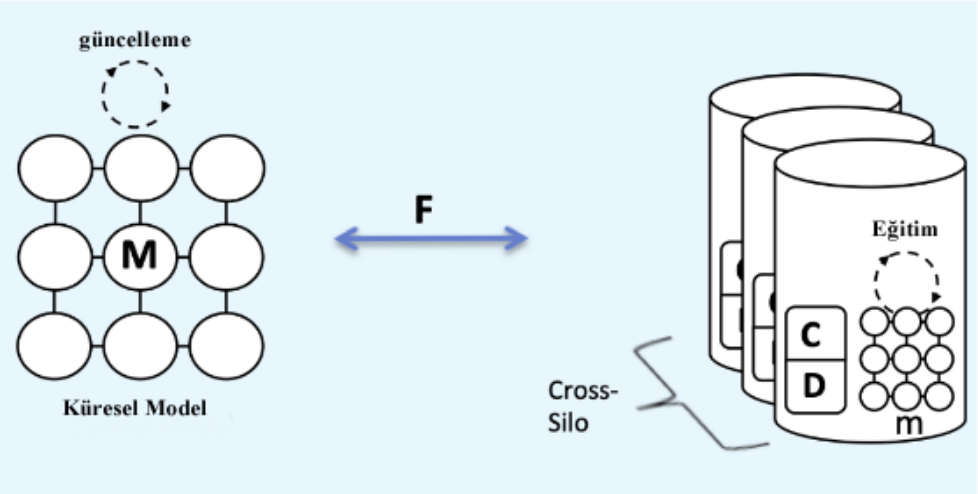
3.2 Federe Öğrenme Ayarları

Federe öğrenme cihazlar arası ve silolar arası olmak üzere iki önemli ayar yöntemiyle ayırt edilir [16]. Tezin bu bölümünde bu ayar yöntemleri açıklanacaktır.

3.2.1 Silolar Arası (Cross-Silo) Federe Öğrenme

Silolar Arası Federe Öğrenme, katılan istemcilerin az, iletişimin güvenli ve işlemin yoğun olduğu durumlarda kullanılabilecek federe öğrenme yöntemidir [16]. Bir organizasyon yasal kısıtlamalardan dolayı kendi verilerini merkezileştiremediğinde veya benzer hedefleri olan kuruluşlar modellerini iş birliği içinde geliştirmek istediğinde kullanılabilir. Örneğin, farklı bankalar dolandırıcılık tespiti için sınıflandırma veya anormallik tespit modellerini iş birliği içinde eğitebilir [16].

Şekil 3.7'de silolar arası federe öğrenme örneği gösterilmiştir. Şekildeki her bir silindir, bir organizasyona ait veritabanını temsil etmektedir. D harfi özel data, C harfi hesaplama, m harfi yerel model, M harfi küresel model ve F harfi ise federe öğrenme sürecini temsil etmektedir.

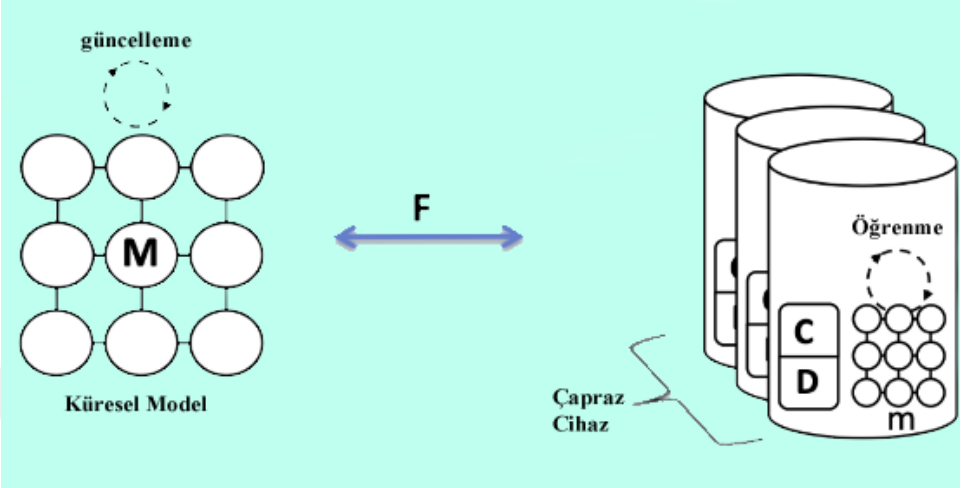


Şekil 3.7: Silolarar arası federe öğrenme örneği [9]

3.2.2 Cihazlar Arası (Cross-Device) Federe Öğrenme

Cihazlar Arası Federe Öğrenme, milyarlarca cihazdan meydana gelen bir istemci kümesi ile çalışıldığı durumlar için kullanılan federe öğrenme yöntemidir [17]. Şekil

3.8’de cihazlar arası federe öğrenme örneği gösterilmiştir. Şekildeki her bir silindir, bir organizasyona ait telefon, IoT cihaz veya yazılımı temsil etmektedir. D harfi özel data, C harfi hesaplama, m harfi yerel model, M harfi küresel model ve F harfi ise federe öğrenme sürecini temsil etmektedir.



Şekil 3.8: Cihazlar arası federe öğrenme örneği [12]

3.3 Federe Öğrenme Algoritmaları

FedAvg [18], Google tarafından federe öğrenme sorunlarını çözmek için formüle edilen ilk federe öğrenme algoritmasıdır [1]. O zamandan beri, ortaya çıkan FedDANE [19], FedNS [20], FedProx [21], FedFa [22], FedBN [23], FedMa [24], FedOpt [25] ve FedCurv [26] bunlardan birkaçı olmakla birlikte, en sık kullanılan algoritma FedAvg’dir.

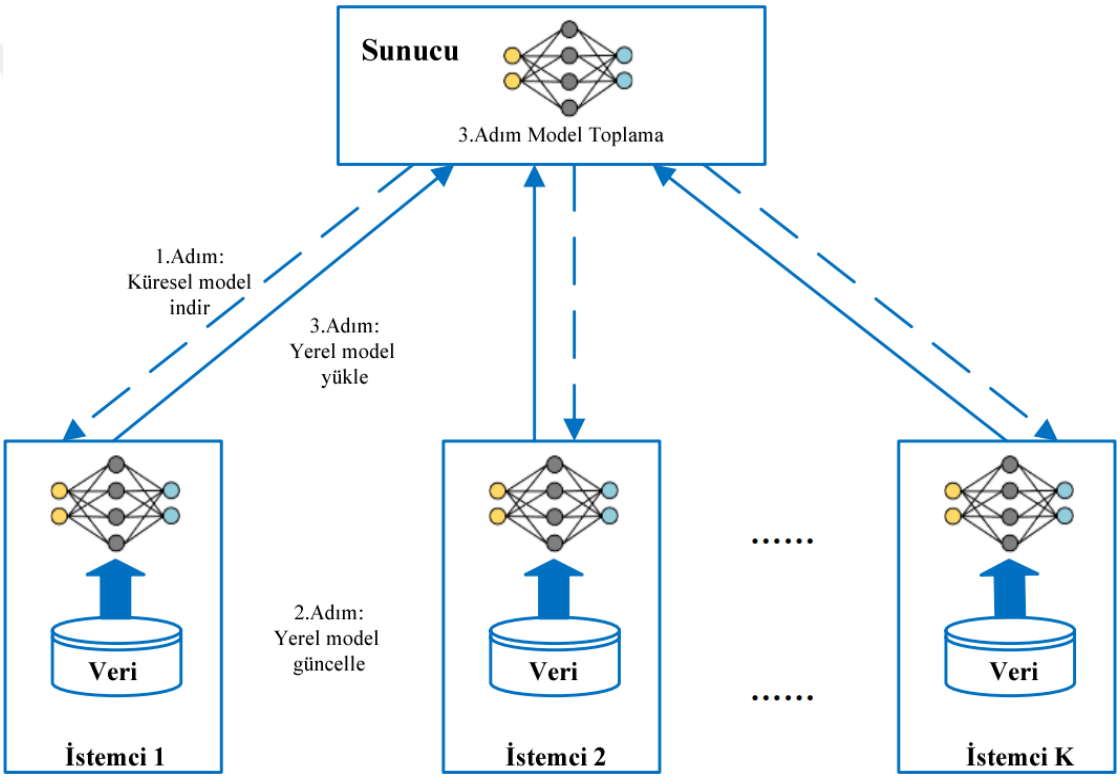
Federe Öğrenme, temelde dört adımı içermektedir [27]:

1. İstemci Seçimi: Ağdaki istemciler rastgele veya istemci seçimiyle ilgili algoritmalar tarafından seçilir.
2. Parametre Yayını: Eğitilen küresel model ve parametreler seçili istemcilere gönderilir ve istemcilerdeki model, sunucunun parametreler yayınına göre güncellenir.
3. Yerel Eğitim: İstemciler paralel olarak güncellenen modele göre, yerel verileriyle yeniden eğitilir.
4. Model Toplama: İstemciler yerel model parametrelerini sunucuya geri gönderir, sunucu ve model parametreleri küresel modele doğru toplanır.

Yukarıdaki adımlar, *n* kez yinelemeli bir şekilde veya istenilen şekilde tekrarlanır [27].

3.3.1 FedAvg (Federe Ortalama) Algoritması

FedAvg, çok sayıda istemciyle dağıtılmış eğitim için iletişim açısından verimli bir algoritmadır. FedAvg algoritmasında istemciler, gizlilik koruması için verilerini yerel olarak tutar; istemciler arasında iletişim kurmak için merkezi bir parametre sunucusu kullanılır. Bu merkezi sunucu, parametreleri her istemciye dağıtır ve istemcilerden güncellenen parametreleri toplar [18]. McMahan vd. 2017 yılındaki çalışmaları ile parametrelerin optimizasyonun ne kadar önemli olduğunu göstermiştir [1].



Şekil 3.9: FedAvg federe öğrenme çerçevesi [18]

3.3.2 FedDANE Algoritması

Federe öğrenme, toplu olarak dağıtılmış uzak cihazlar üzerinden istatistiksel modelleri ortaklaşa öğrenmeyi amaçlar. FedDANE, federe öğrenmenin pratik kısıtlamalarını ele almak için klasik dağıtık optimizasyon için bir yöntem olan DANE'den uyarlanmış bir algoritmadır [19].

DANE algoritması temelinde, her istemci kendi alt problemini çözdükten sonra, merkezi sunucu bu güncellemeleri toplar ve bunları işleme alır. FedDANE ise, rastgele örneklenmiş cihazlardan bir gradyan alt kümesi kullanarak tam gradyanlara ulaşmayı hedefler [19].

3.3.3 FedNS (Federated Node Selection) Algoritması

FedNS, sunucunun federe öğrenme ayarında global model toplaması için Federe Düğüm Seçimi (FedNS) yaklaşımı üzerine kuruludur. FedNS, istemci modellerini düğüm/çekirdek düzeyinde filtreler ve yeniden ağırlıklandırır. Böylece istemcilerin en iyi bileşenlerini birleştirerek potansiyel olarak daha iyi bir küresel modele yol açar. Zhuo ve Li [20] tarafından 2021 yılında yapılan deneysel çalışmada, işbirlikçi görüntü sınıflandırması kullanarak, FedNS algoritmasının FedAvg üzerinden sürekli olarak iyileştirilmiş performans elde edebileceğini gösterilmiştir [20].

Yapılan deneysel çalışmada, her istemcide, her sınıf için örnek sayısına dayalı ağırlıklar kullanılarak FedAvg algoritmasında potansiyel iyileştirmeyi amaçlayan FC (Fully Connected) katmanı oluşturan eklenti kullanılmıştır. FC katmanı yapay sinir ağının olduğu kısımdır. Bu katman yapay sinir ağlarıyla öğrenmenin gerçekleştiği katmandır.

Söz konusu çalışmada sonuçlar karşılaştırıldığında mikro saniyelik iyileşmeler olduğu gösterilmiş ve farklı veri kümesiyle yapılan deney sonuçları Tablo 3.1-3.3'te sunulmuştur [20].

Fashion MNIST	**iid**		**non-iid**		
	FedAvg	**FedNS**	**FedAvg**	**FedAvg + lastFC**	**FedNS**
accuracy	83.53	**83.79**	82.96	83.5	**83.56**
macro precision	0.834	**0.838**	0.83	0.837	**0.838**
macro F-Score	0.931	**0.936**	0.926	**0.933**	0.929

Tablo 3.1: FashionM-NIST veri kümesiyle yapılan deneysel sonuçlar [20]

CIFAR10	iid		non-iid		
	FedAvg	**FedNS**	**FedAvg**	**FedAvg + lastFC**	**FedNS**
accuracy	66.09	**67.07**	65.86	65.57	**67.55**
macro precision	0.658	**0.668**	0.660	0.658	**0.671**
macro F-Score	0.740	**0.745**	0.737	0.737	**0.745**

Tablo 3.2: CIFAR-10 veri kümesiyle yapılan deneysel sonuçlar[20]

tiny ImageNet	iid		non-iid		
	FedAvg	**FedNS**	**FedAvg**	**FedAvg + lastFC**	**FedNS**
accuracy	18.99	**19.11**	18.64	18.89	**18.91**
top-5 accuracy	41.14	**41.39**	40.80	40.86	**41.05**

Tablo 3.3: TinyImageNet veri kümesiyle yapılan deneysel sonuçlar [20]

3.3.4 FedProx Algoritması

FedProx, federe öğrenme için mevcut önde gelen yöntem olan FedAvg'nin genelleştirilmesi ve yeniden parametrelendirilmesi olarak görülebilir. FedProx, her turda (Şekil 3.9) bir cihaz alt kümesinin seçilmesi, yerel güncellemelerin yapılması ve daha sonra bu güncellemelerin küresel bir güncelleme oluşturmak için ortalamasının alınması bakımından FedAvg algoritmasına benzermektedir. Ancak, 2020 yılında Li vd. [19], önemli deneysel iyileştirmelerle sonuçlanan ve ayrıca yöntem için yakınsama garantileri sağlanmasına olanak tanıyan basit ama kritik değişiklikleri yaparak FedAvg'den daha istirarlı yakınsama oranına ulaştıklarını çalışmaları FedProx ile duyurmuştur [21]. Sahu vd. [21]'nin farklı veri kümeleriyle yapmış olduğu deneysel çalışmalarda, FedProx algoritmasının FedAvg'ye kıyasla doğruluk oranında %22 artış olduğu gösterilmiştir.

3.3.5 FedFA Algoritması

FedFA, çift momentum gradyan yöntemi ve uygun ağırlıklandırma stratejileri olarak iki bölüm halinde sunulmaktadır [47]. Çift momentum gradyan yöntemindeki temel fikir, gradyanın üstel ağırlıklı ortalamasını hesaplayıp, bu gradyanı ağırlıkları güncellemek için kullanılmasıdır [47]. Genel olarak, bir federe ağdaki istemci sayısı yüzlerce hatta milyonlarca olabilir. Federe öğrenmenin genel amacı, küresel modeli istemcideki verilere uygulamaktır. Ortalamada doğruluk oranı yüksek olsa da ağdaki her istemcinin doğruluk oranı her zaman çok iyi olmayabilir. Federe ağdaki istemciler arasında model performansının daha adil bir şekilde dağıtılmasını teşvik etmek için uygun ağırlıklandırma stratejisi FedFA algoritması ile öne sürülmüştür [47] .

Huang vd. [47] tarafından yapılan deneysel çalışma sonucunda, farklı tekniklerle FedAvg, FedProx ve FedFA algoritmalarıyla yapılan karşılaştırmaların birinde FedAvg'den düşük ancak FedProx'dan daha iyi bir sonuç elde edilmiş olsa da genel ortalama dikkate alındığında %20'ye varan iyileşmeler elde edilmiştir (Tablo 3.4).

Veri kümesi	Metot	Ortalama	En kötü 20%	En iyi 20%
Syndthetic_iid	**FedAvg**	89.11%	78.17%	100.00%
	FedProx	71.92%	55.46%	86.09%
	FedFa	**85.70%**	**71.46%**	**100.00%**
Syndthetic_0_0	**FedAvg**	54.67%	6.81%	98.18%
	FedProx	75.02%	37.21%	100.00%
	FedFa	**78.25%**	**43.41%**	**100.00%**
Syndthetic_0.5_0.5	**FedAvg**	40.24%	0.00%	98.18%
	FedProx	67.84%	33.60%	100.00%
	FedFa	**73.30%**	**41.27%**	**100.00%**
Syndthetic_1_1	**FedAvg**	54.78%	1.38%	98.85%
	FedProx	64.75%	9.72%	100.00%
	FedFa	**76.88%**	**37.03%**	**100.00%**
Femnist	**FedAvg**	70.96%	34.77%	100.00%
	FedProx	72.20%	40.50%	100.00%
	FedFa	**77.96%**	**48.99%**	**100.00%**
Sent140	**FedAvg**	68.51%	40.58%	93.22%
	FedProx	64.71%	34.02%	92.65%
	FedFa	**68.83%**	**42.66%**	**95.71%**

Tablo 3.4 : FedFA algoritmasının FedAvg ve FedProx ile yapılan deneysel sonuçları [47]

3.3.6 FedBN Algoritması

FedBN olarak adlandırılan öğrenme algoritması non-IID özellikli veri kümelerine yönelik bir önermedir [23].

FedAvg'ye benzer şekilde, FedBN yerel güncellemeler gerçekleştirir ve yerel modellerin ortalamasını alır. Ancak FedBN, yerel modellerin yığın normalleştirme katmanlarına sahip olduğunu varsayar ve parametrelerini ortalama alma adımından hariç tutar [23].

Li vd.[19]'nin yapmış olduğu çalışmalarında FedAvg, FedProx ve FedBN algoritmaları karşılaştırılmıştır. Çalışma neticesinde farklı veri kümelerinde %15'i bulan iyileşmeler elde edilmiştir (Tablo 3.5).

Metot	**SVHN**	**USPS**	**Synth**	**MNIST-M**	**MNIST**
Single	65.25 (1.07)	95.16 (0.12)	80.31 (0.38)	77.77 (0.47)	94.38 (0.07
FedAvg	62.86 (1.49)	95.56 (0.27)	82.27 (0.44)	76.85 (0.54)	95.87 (0.20)
FedProx	63.08 (1.62)	95.58 (0.31)	82.34 (0.37)	76.64 (0.55)	95.75 (0.21)
FedBN	**71.04 (0.31)**	**96.97 (0.32)**	**83.19 (0.42)**	**78.33 (0.66)**	**96.57 (0.13)**

Tablo 3.5: Yapılan deneysel sonuçlara göre FedBN'nin doğruluk oranları [23]

3.3.7 FedMA Algoritması

FedMA algoritması, evrişimli sinir ağları (CNN'ler) ve uzun kısa süreli bellekler (LSTM'ler) gibi modern sinir ağı mimarilerinin birleştirilmiş öğrenimi için tasarlanmıştır. FedMA, benzer özellik çıkarma imzalarıyla gizli öğeleri (evrişim katmanları için kanallar, LSTM için gizli durumlar, tamamen bağlı katmanlar için nöronlar) eşleştirerek ve ortalamasını alarak, paylaşılan küresel modeli katman bazında oluşturur [24].

Söz konusu çalışmada farklı veri kümeleriyle yapılan çalışmada %8'i bulan iyileştirmeler sunulmuştur (Tablo 3.6).

Metot	FedAvg	FedProx	Ensemble	FedMA
Final Accuracy(%)	86.29	85.32	75.29	**87.53**
Best local epoch(E)	20	20	N/A	150
Model growth rate	1x	1x	16x	1.11x

Tablo 3.6:Yapılan deneysel sonuçlara göre FedMA'nin doğruluk oranları [24]

3.3.8 FedOpt Algoritması

Reddi vd. [25]'ne göre, birleşik olmayan ortamlarda uyarlamalı optimizasyon yöntemleri mücadelede kayda değer bir başarı elde etmiştir. Bu sebeple klasik yöntemlerde başarılı olmuş çeşitli optimizasyon algoritmalarının federe yöntemlerini önermiştir. Yapılan çalışmada, uyarlanabilir optimize edicilerin kullanımının federe öğrenme performansını önemli ölçüde iyileştirebileceğini öne sürülmüştür [25].

3.3.9 FedCurv Algoritması

Shoham vd. yapmış oldukları çalışmada, FedAvg ile non-IID veriler üzerinde eğitim, yerel olarak eğitilen modellerin çok az veriye sahip oldukları veya hiç veriye sahip olmadıkları sınıfları "göz ardı etmelerine" yol açtığını öne sürerek, bu sorunun üstesinden gelmek için FedCurv algoritmasının kullanılmasını önermişlerdir [26].

3.4 Veri Gizliliği

Federe öğrenme, kalıcı verilerle ilgili veri merkezi eğitimine kıyasla belirgin gizlilik avantajlarına sahiptir. Anonimleştirilerek sunucuya gönderilen güncellemelerden verilerin tekrardan yapılandırılabileceği düşünülmektedir. Esasen, federe öğrenme için iletilen bilgi, belirli bir modeli geliştirmek için gereken minimum güncelleme olmalıdır. Ham eğitim verisinden bilgi içermeyen parametreler, güncellemelerle küresel model eğitilir [1] .

Federe öğrenmede klasik makine öğrenmesi yöntemlerinden farklı olarak, verilerin yerel cihazı terk etmemesi büyük bir eksikliği giderileceği düşünülmektedir [15].

Dolaylı bilgi sızıntısı, birleştirilmiş öğrenmenin öncü çalışmaları, Stochastic Gradient Descent (SGD) gibi bir optimizasyon algoritmasından parametre güncellemeleri gibi ara sonuçları ortaya çıkarır, ancak hiçbir güvenlik garantisi sağlanmaz ve bu gradyanların sızması aslında önemli veri bilgilerini sızdırabilir [13].

Diferansiyel gizlilik, bu tür gizlik sorunlarına karşı standart bir yaklaşımdır. Diferansiyel gizlilik, bir saldırganın rastgele bir algoritmanın çıktısı yoluyla çıkarabileceği bilgilere karşı veriye gürültü ekleyerek tek bir bireyin algoritmanın çıktısı üzerindeki etkisine bir sınır sağlar.

Homomorfik şifreleme, makine öğrenmesi sırasında şifreleme mekanizması altında parametre değişimi yoluyla kullanıcı verilerinin gizliliğini korumak için benimsenmiştir [3]. Yereldeki veriler sunucuya şifreli olarak iletilir ve geri alınır. Bu verinin yeniden yapılandırılma ihtimalini minimize etmektedir. Bu işlemlerin en iyi, Cheon-Kim-Kim-Song (CKKS) ve Brakerski / Fan-Vercauteren (BFV) şemaları ile kullanıldığı öne sürülmektedir [15].

Federe öğrenmede, yerel ve küresel olarak iki farklı yöntemden biri kullanılabilir [28]. Yerel diferansiyel gizlilik kullanımında, diferansiyel gizlilik için gereken gürültü eklemesi, her istemci tarafından yerel olarak gerçekleştirilir. Diğer yandan, küresel diferansiyel gizlilikte ise, gizlilik için gerekli gürültü eklemesi sunucuda yapılır. İstemciler, sunucuya güvenmek zorundadırlar. Öte yandan, küresel diferansiyel gizliliği kullanırken, verilerini ileten kişilerin gizliliklerini korumak adına gerekli gürültüyü eklemek için veri kümesi sunucusuna güvenmeleri gerekir [28].

3.5 Açık Kaynak Federe Öğrenme Kütüphaneleri

Federe öğrenmenin 2017'de Google tarafından tanıtılmasıyla birlikte pek çok açık kaynak kodlu, kütüphaneler ve çerçeveler geliştirilmiştir. Bunlardan birkaçı, kullanım farklılıklarını dikkate alarak bu bölümde tanıtılacaktır.

3.5.1 FedML Çerçevesi

FedML araştırma odaklı açık kaynak bir kütüphanedir [29] . Ayrıca federe öğrenme dağıtımını basitleştirmek için kullanıcı dostu bir MLOps platformu sağlar. FedML,

IoT ve mobil cihazlar için model eğitiminde, dağıtılmış bilgi işlem ve tek makine simülasyonunda, çok çeşitli sektörlerde (sağlık, finans, sigorta, akıllı şehirler, IoT vb.) ve uygulamalarda (bilgisayar vizyonu, doğal dil işleme, veri madenciliği ve zaman serisi tahmini) dikey çözümleri de destekler. FedML'ye ellinin üzerinde yayın içeriğinde atıfta bulunulmuştur [30]. FedML ekosisteminde, çeşitli uygulama alanları için federe öğrenme araştırmasını ve ürünleştirmeyi kolaylaştıran çözümler mevcuttur. FedML Core Framework'ün temel desteğiyle, doğal dil işleme için FedNLP [31] , bilgisayar görüşü için FedCV [32] , grafik sinir ağları için FedGraphNN [33] ve nesnelerin interneti için FedIoT [34] çerçevelerini destekler.

3.5.2 OpenFL Çerçevesi

OpenFL, başlangıçta Intel® Labs ve Pensinvalya Üniversitesi tarafından geliştirilen [35], topluluk destekli açık kaynaklı bir projedir [36].

OpenFL çerçevesinde de, istemciler ile parametre sunucusu arasındaki iletişim Grpc [37] ve TLS [38] protkolleri üzerinden güvenli bir şekilde sağlanmaktadır. OpenFL, bir Python API'si ve bir komut satırı arayüzü ile birlikte gelir. Hem TensorFlow hem de PyTorch ile çalışır ve diğer ML ve derin öğrenme çerçevelerine kolayca genişletilebilir [35].

Başlangıçta tıbbi görüntülemede kullanılmak üzere geliştirilmiş olan OpenFL, kullanım durumu, endüstri ve makine öğrenmesi çerçevesi için de kullanılacak şekilde tasarlanmıştır. OpenFL çerçevesi, FedAvg, FedProx, FedOpt ve FedCurv algoritmalarını desteklemektedir [35] .

3.5.3 TensorFlow Federe Öğrenme Çerçevesi

TensorFlow Federe Öğrenme Çerçevesi (TensorFlow Federated Framework (TFF)) dağıtık veriler üzerinde makine öğrenmesi ve hesaplamalar için açık kaynaklı bir çerçevedir. TFF, eğitim verilerini yerel olarak tutan birçok katılımcı istemci arasında paylaşılan bir küresel modelin eğitildiği bir makine öğrenmesi yaklaşımı olan Federated Öğrenme (FL) ile açık araştırma ve deneyleri kolaylaştırmak için geliştirilmiştir [39].

TFF, federe öğrenmeyle çözülebilecek gerçek dünya sorunlarını temsil eden pek çok veri kümesi barındırmakla beraber [40], çok çeşitli örnek kodlamalarda makalelerle beraber bulunmaktadır [41].

TFF'nin arayüzleri Federe Öğrenme API ve Federe Çekirdek API olarak iki ana katmanda düzenlenmiştir:

- Federe Öğrenme API, geliştiricilerin, dahil edilen federe öğrenme ve değerlendirme uygulamalarını mevcut TensorFlow modellerine uygulamalarına olanak tanıyan bir dizi arabirim sunar. Üç ana bileşenden oluşmaktadır.

Modeller:	Mevcut modellerin TFF ile sarılmasını sağlayan sınıflar ve fonksiyonlar.
Birleşik Hesap Oluşturucular:	Birleşik hesaplamalar oluşturmak için işlevler.
Veri kümeleri	Simülasyon senaryoları için kullanılacak hazır veri kümeleri

- Federe Çekirdek API sistemin merkezinde, TensorFlow'u dağıtılmış iletişim operatörleri ile güçlü bir şekilde tanımlanmış işlevsel programlama ortamında birleştirerek yeni birleşik algoritmaları kısa ve öz bir şekilde ifade etmek için bir dizi alt düzey arabirim bulunur. Bu katman aynı zamanda federe öğrenmenin üzerine kurulan temel görevi görür.

Federe Öğrenme API makine öğrenmesi geliştiricilerinin federe öğrenmeden TensorFlow modellerine uygulamalarına ve federe öğrenme araştırmacılarının yeni algoritmalar tanıtmasına [42] yardımcı olurken, Federe Çekirdek API ise sistem araştırmacıları içindir [43] .

3.5.4 Nvidia Clara Çerçevesi

NVIDIA Clara, sağlık hizmeti kullanım durumları için tasarlanmış bir uygulama çerçevesidir. Geliştiriciler, veri bilimciler ve araştırmacılar için gerçek zamanlı, güvenli ve ölçeklenebilir birleşik öğrenme çözümleri oluşturmak için GPU kütüphaneleri, SDK'lar ve referans uygulamaları içerir [44].

NVIDIA Clara'da tıbbi cihazlar için Clara Holoscan MGX, ilaç geliştirme için Clara Discovery, akıllı hastaneler için Clara Guardian, tıbbi görüntüler için Clara Imaging ve genomik veriler için Clara Parabricks çözümleri mevcuttur [44].

3.5.5 PySyft ve PyGrid Kütüphaneleri

OpenMined topluluğu tarafından geliştirilen PySyft, araştırma amaçları için birleşik öğrenme sağlayan ve FL, diferansiyel gizlilik ve şifreli hesaplamalar kullanan açık kaynaklı Python 3 tabanlı bir kütüphanedir [45].

PyGrid, web, mobil, uç cihazlarda ve farklı türdeki terminallerde birleşik öğrenmeyi uygular [46]. PyGrid ayrıca hem model merkezli hem de veri merkezli federe öğrenmeyi çalıştırmak için kullanılan programlama arayüzüdür. PyGrid Admin kullanılarak kontrol edilebilir [47].

3.5.6 OpenMind Web ve Mobil Kütüphaneleri

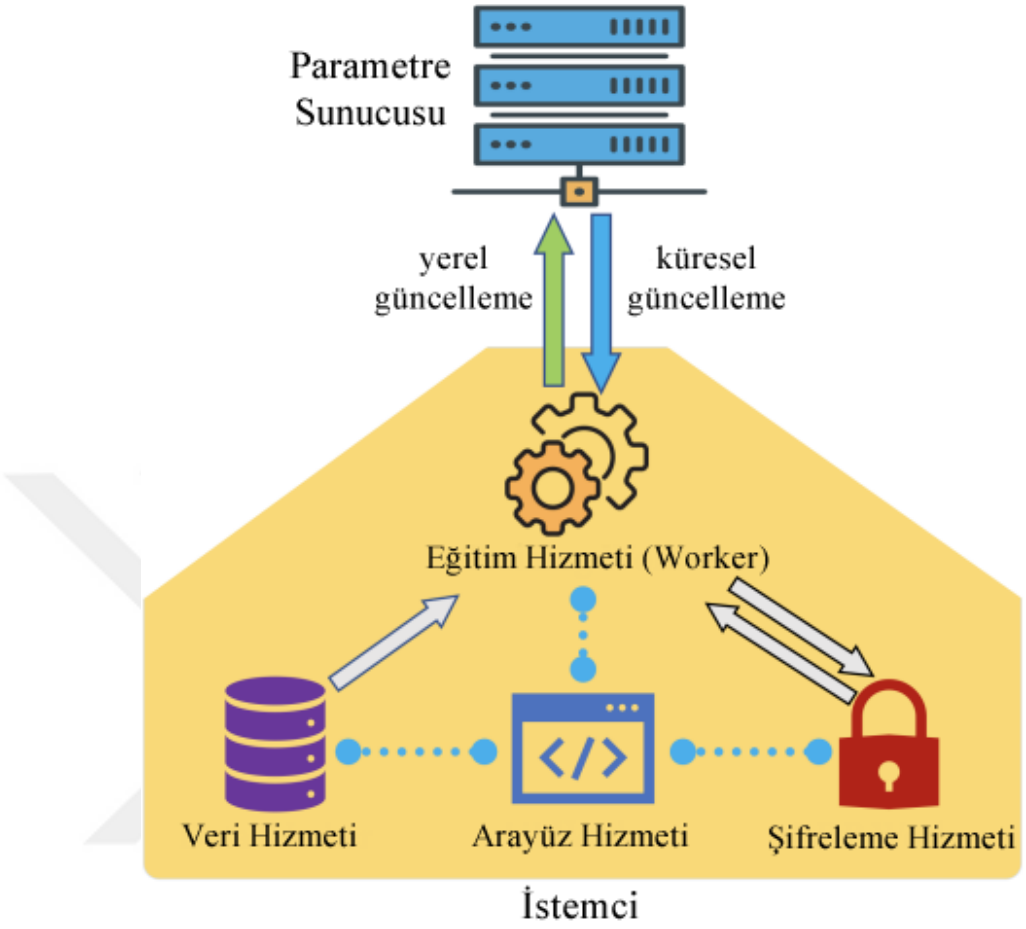
OpenMined.org, ücretsiz ve açık kaynaklı bir yazılım geliştirme topluluğudur. Topluluk 2020 yılında model merkezli federe öğrenme için her biri PyGrid tarafından merkezi olarak koordine edilen dört yeni kütüphane geliştirdiğini duyurmuştur [48].

- Syft.js, web, iOS, Android ve sunucular/IoT'yi kapsayan federe öğrenme için kullanılan açık kaynak ekosisteminin "web" kısmıdır [48].
- KotlinSyft, Android cihazlarda federe öğrenme için kullanılan bir kütüphanedir [48].
- SwiftSyft, iOS cihazlarda birleşik öğrenim için kullanılan bir kütüphanedir [48].
- Threepio, PyTorch, TensorFlow ve TensorFlow.js arasındaki komutları bir çerçeveden diğerine çevirmek için kullanılan bir kütüphanedir [48].

3.5.7 WebFed Çerçevesi

WebFed, mimarisinin Şekil 3.10'da gösterildiği şekilde tarayıcının özelliklerinden (örneğin, Çapraz platform, JavaScript Programlama Özellikleri) yararlanan ve yerel

diferansiyel gizlilik mekanizması aracılığıyla gizlilik korumasını geliştiren tarayıcı tabanlı federe öğrenme çerçevesidir [49].



Şekil 3.10: WebFed mimarisi [49]

WebFed Mimarisi süreçleri şu şekilde açıklanabilir.

• Arayüz Hizmeti, ilgili web sayfası açıldığında başlatılır. İstemciler, arayüz aracılığıyla sunucuya bağlanma, eğitim görevlerini seçme vb. gibi bazı işlemleri gerçekleştirebilir.

• Veri Hizmeti, eğitime esas olarak yerel veri kümelerini yöneterek katılır. Eğitim verilerini önceden yükleyerek ve önceden işleyerek tüm eğitim sürecini optimize edebilir.

• LDP Hizmeti, küresel model birleştirilmesinden önce yerel eğitim sonuçlarına yapay gürültü eklemekten sorumludur.

• Eğitim Hizmeti, belirli eğitim görevlerini yerine getirmekten ve sunucuyla etkileşim kurmaktan sorumludur. İlk olarak, sunucu tarafından yayınlanan küresel modeli indirecek ve ardından tutulan yerel modeli güncelleyecektir. Veri hizmeti tarafından sağlanan eğitim verilerini kabul eder ve ardından tarayıcı içi eğitim sürecini yürütür. Yerel eğitimden sonra, yapay gürültü ile model ağırlıklarını elde etmek ve bunları global toplama için parametre sunucusuna yüklemek için LDP Hizmeti ile etkileşime girecektir [49].

3.5.8 Leaf: Federe öğrenme için açık kaynak veri kümeleri

LEAF, federe öğrenme, çok görevli öğrenme, meta-öğrenme ve cihaz üzerinde öğrenme gibi uygulamalarla birleştirilmiş ortamlarda öğrenmeye yönelik bir kıyaslama çerçevesidir. Resim sınıflandırma için Celeba ve FEMNIST, Sonraki karakter tahminlemesi için Shakespeare, Duygu analizi için Twitterdan alınan Sentiment140, Sınıflandırma için Synthetic ve dil modelleme için Reddit veri kümeleri federe öğrenme için kullanıma hazır hale getirilmiştir [50].

3.5.9 Hugging Face Transformers Kütüphanesi

Hugging Face Transformers, son teknoloji önceden eğitilmiş modelleri kolayca indirmek ve eğitmek için API'ler ve araçlar sağlar. Önceden eğitilmiş modelleri kullanmak, bir modeli sıfırdan eğitmek için gereken zamandan ve kaynaklardan tasarruf etmek isteyen özellikle girişimciler ve kendi modellerinin sonuçlarını farklı modellerle karşılaştırmak isteyen araştırmacılar için ideal bir kaynaktır. Doğrudan Python projenize eğitilmiş modeli ekleyebilir ve çalışmalarınız da kullanılabilmektedir [51].

3.5.10 Flower Çerçevesi

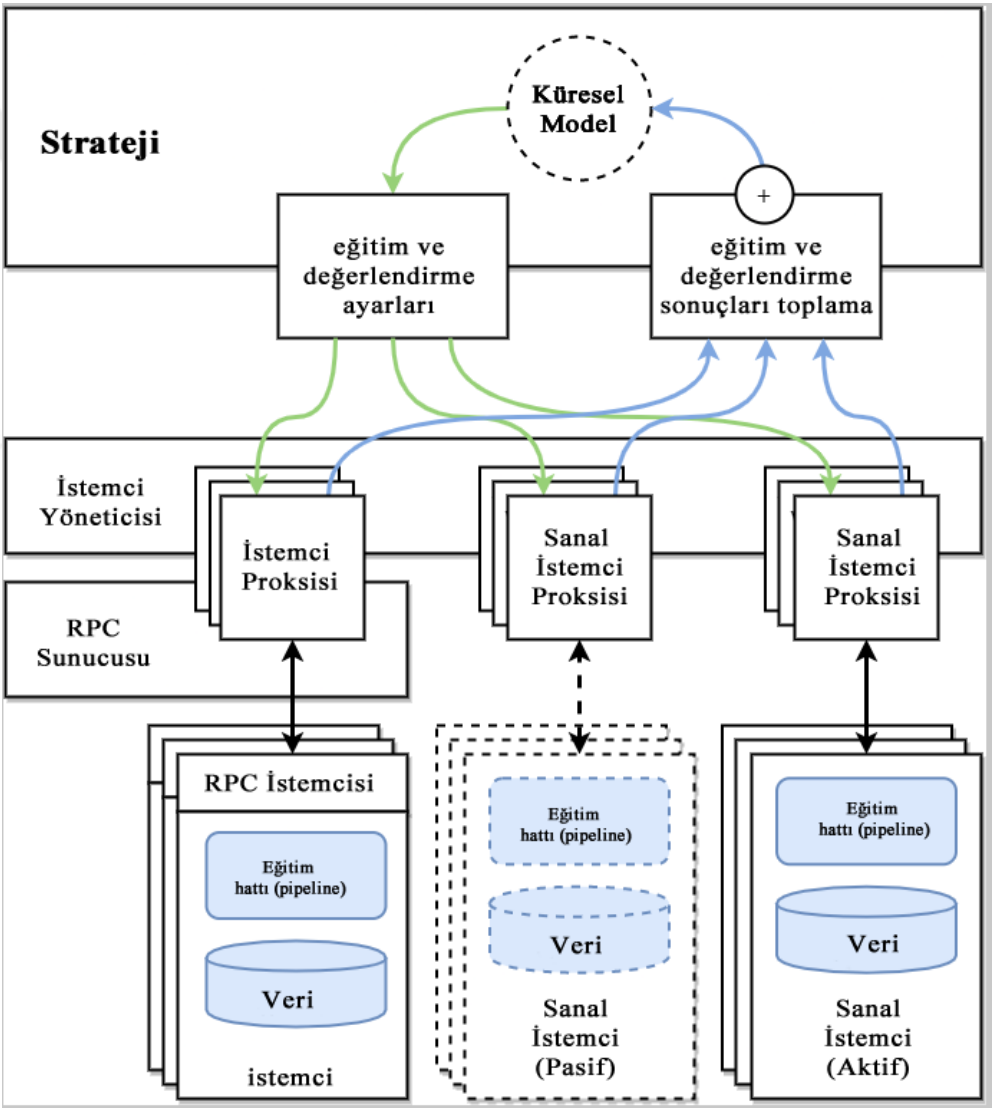
Açık kaynak kodlu federe öğrenme çerçevesi olan Flower, mevcut makine öğrenmesi iş yüklerini federe öğrenme ortamına taşımak isteyen araştırmacıları ve geliştiricileri hedeflemiştir. Flower'ın hedeflerinden biri bunu basitleştirmektir [52].

PyTorch, TensorFlow, HuggingFace Transformers, JAX (merkezi JAX tabanlı makine öğrenmesi), Pandas, fastai, PyTorch Lightning, Apache MXNet (açık kaynak derin

öğrenme çerçevesi) ve scikit-learn (makine öğrenmesi kütüphanesi) kütüphanelerine yönelik hızlı başlangıç açık kaynak kodları ve makaleleri ile araştırmacılar veya girişimciler için kolaylık sağlamaktadır.

Beutel vd. [52] tarafından yapılan deneysel çalışmalarda 15 milyon istemci ile gerçekleştirilen sonuçlar sunulmuştur.

Uç istemciler (edge client) gerçek istemcilerle RPC üzerinden iletişim kurar. Sanal istemci motoru ise sadece eğitim ve değerlendirme için kullanılır, aktif değilken sıfıra yakın kaynak tüketir [52]. Mimari Şekil 3.11'de gösterilmiştir.



Şekil 3.11: Flower çekirdek mimarisi [52]

Flower çerçevesi, ölçeklenebilirlik, heterojenlik ve gizlilik şeklinde üç ana duruma odaklanarak geliştirilmiştir, .

Ölçeklenebilirlik: Flower'ın tek makineli simülasyonlarda mevcut kaynakları verimli bir şekilde kullanılabildiği ve her eğitimde binlerce örnekleme ve milyonlarca istemciyle deneyler yürütebildiği çerçevenin tanıtıldığı çalışmada gösterilmiştir [52].

Heterojenlik: Flower'ın cihazlar arası senaryoda, heterojen cihazlara dağıtılabileceğini ve sistem istatistiklerini ölçmek için nasıl kullanılabileceği referans çalışmada sunulmuştur [47].

Gizlilik: Son olarak, güvenli toplamanın beklenen teorik bilgilerle nasıl eşleştiğine ilişkin gerçekleştirimin nasıl olduğunu gösterilmiştir [52] .

3.5.10.1 Flower Başlangıç Paketleri

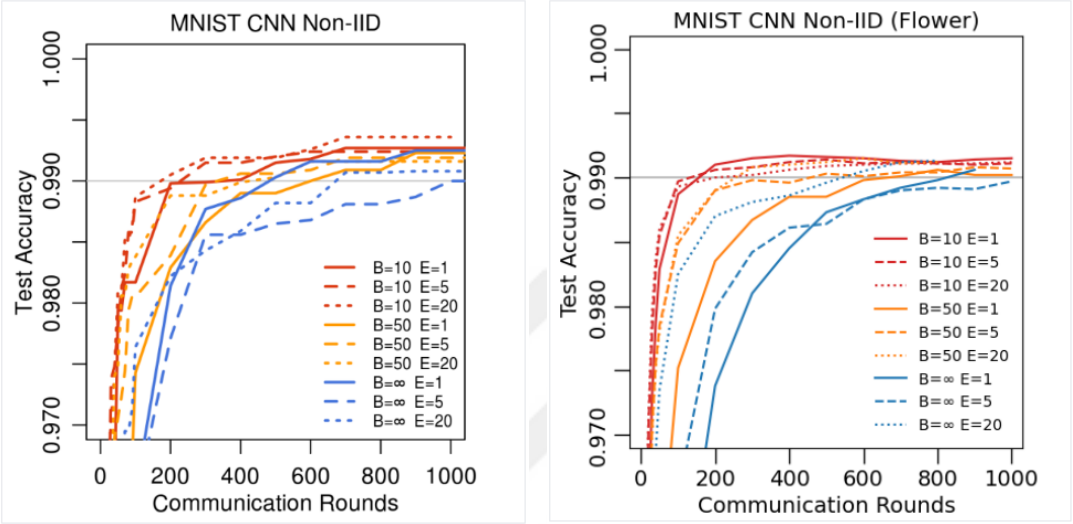
Bazı iyi bilinen federe öğrenme yayınlarından deneyleri yeniden üreten bir Flower açık kaynak kod koleksiyonudur. Federe öğrenmeyi keşfetmeye başlamak için iyi bir seçenek olması itibariyle, yaygın olan FedAvg ile FedProx algoritması karşılaştırmalı deneysel sonuçlarının paylaşılması çerçeve hakkında daha iyi bir fikir sahibi olmamamızı sağlayacaktır.

Başlangıç Paketi: CNN kullanarak MNIST veri kümesi üzerinde FedAvg:

CNN kullanılarak MNIST veri kümesi üzerinde FedAvg çalışması [53], 2017 McMahan vd.'nin MNIST deneyinin flower çerçevesi ile karşılaştırmalı bir uygulamasıdır [1]. Gerçekleştirilen deneysel çalışmanın kaynak kodları flower çerçevesinin github adresinde flwr_baselines/publications/FedAvg_mnist altında bulunmaktadır [53].

FedAvg nasıl uygulandığıyla ilgili makalede, McMahan vd.'nin yayınının birebir kopyalamaya çalışarak iki bölümleme yöntemi yeniden deneye tabi tutulduğu görülmüştür [54]. Söz konusu deneyde ilk olarak, verilerin istemciler arasında rastgele dağıtıldığı, böylece her birinin her sınıf için kabaca aynı sayıda örneği tuttuğu IID bölümlemesi vardır. Bu senaryonun amacı gerçek hayattaki bir durumu taklit etmekten çok bir referans noktası vermektir [54]. İkinci olarak, IID olmayan bölümleme vardır. Hiçbir istemcinin verilerinde dört'ten fazla farklı etiket olmayacak şekilde veri dağıtılmıştır. Bu, her kullanıcının farklı bir veri dağılımına sahip olabileceği gerçek dünya durumuna daha yakındır [54].

Flower Çerçevesi ile yapılan çalışma sonuçlarıyla, McMahan vd.'nin çalışmasının sonuçları karşılaştırmak için, deneyi aynı parametrelerle 100 istemci ve 1000 tekrar boyunca yapılmıştır. Şekil 3.12'de görüldüğü gibi, McMahan vd.'nin çalışmasında yer alan çizimle karşılaştırıldığında, 0.99'un üzerinde doğruluk ve yaklaşık olarak aynı sayıda turda (yaklaşık 200 tur) gerçekleştirilmiştir.



Şekil 3.12 Sol: McMahan deneyi sonuçları, Sağ: Flower deneyi sonuçları [53]

Başlangıç Paketi: CNN kullanarak MNIST'te FedProx:

CNN kullanrak'te FedProx çalışması, 2018 Tian Li vd.'nin MNIST deneyinin flower çerçevesi ile karşılaştırmalı bir uygulamasıdır [1] . Çalışmanın kaynak kodları flower çerçevesinin github adresinde, flwr_baselines/publications/ fedprox_mnist altında yer almaktadır [55].

2018 yılında, Tiai Li vd., birleşik bir ortamda derin öğrenme modellerini eğitmek için "FedProx" adlı yeni bir optimizasyon yöntemini tanıttı. FedProx'un arkasındaki temel fikir, standart birleştirilmiş optimizasyon hedefine, istemciler arasında veri dağıtımının etkisini azaltmaya yardımcı olan ve genel model performansını iyileştirmektir [47].

Bölüm 4

Materyal ve Metot

Bu tezde, kullanım kolaylığı, dokümantasyon ile örnek kodların tutarlılığı, ve açık kaynak topluluğu üzerinden sorulan sorulara kısa sürede cevap alınabilmesi sebebiyle, Cambridge Üniversitesi'nden Prof. Nicholas Lane ile misafir araştırmacılar Daniel J. Beutel ve Taner Topal tarafından kurulan Açık kaynak Flower Çerçevesi seçilmiştir.

Bu bölümde, tez için yapılan deneysel çalışmada uygulanan yöntem ve araştırma teknikleri (veri toplama araçları ve analiz yöntemleri dahil) açıklanmıştır. Öncelikle, bu çalışmada kullanılan CIFAR-10 [56] veri kümesi ile özellikleri açıklanmıştır. İkinci olarak, deneysel ortamın hazırlanması ve ayarlanması açıklanmıştır. Daha sonra, kullanılan geliştirme ortamı ve kütüphaneler sunulmuştur. Son olarak, çalışmada kullanılan sınıflandırma algoritması ve parametrelerin anlamları açıklanarak deneysel sonuçlarıyla birlikte ortaya konmuştur.

4.1 Materyal

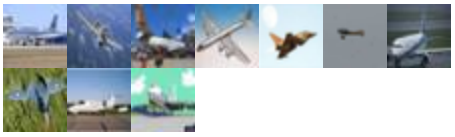
Bu tezde, Meta AI tarafından açık kaynak olarak geliştirilen bir derin öğrenme kütüphanesi olan PyTorch kullanılmıştır. Bu sebeple CIFAR-10 veri kümesi doğrudan PyTorch veri kümesi aracılığıyla ResNet18 ağı üzerinden alınmıştır. ResNet Ağı, ağların eğitimini kolaylaştırmak için, Kaiming He vd. tarafınan "Deep Residual Learning for Image Recognition" makalesinde tanıtılan belirli bir sinir ağı türüdür [57].

4.1.1 CIFAR-10 Veri Kümesi ve Özellikleri

Krizhevsky vd. [56] tarafından oluşturulan CIFAR-10 veri kümesi, sınıf başına 6000 resim olmak üzere, 10 sınıfta toplam 60000 adet 32x32 ebatında renkli resimden

oluşmaktadır. Krizhevsky'nin websitesi [56] üzerinden alınan veri kümesi için rastgele çekilen 10 sınıfa ait örnek resimler aşağıda gösterilmiştir:

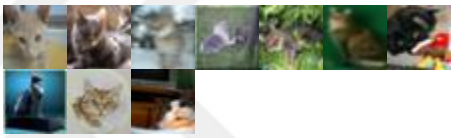
Uçak



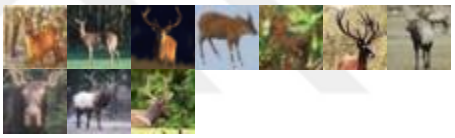
Otomobil

Kuş

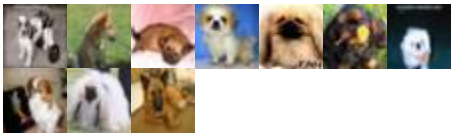
Kedi



Geyik

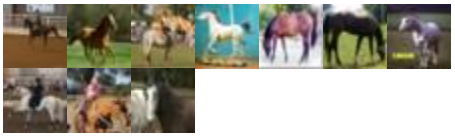


Köpek

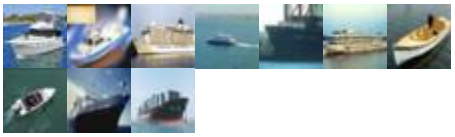


Kurbağa

At



Gemi



Kamyon 

Şekil 4.13: CIFAR-10 veri kümesi örnek resimler [56]

Sınıflar tamamen birbirinden bağımsızdır. Otomobiller ve kamyonlar arasında çakışma yoktur. "Otomobil" sınıfı, sedanları, SUV'ları ve bu tür araçları içerir. "Kamyon" yalnızca büyük kamyonları ve tırları içerir. Hiçbiri kamyonetleri içermemektedir. Amaç daha önce görülmemiş görüntüleri tanımak ve bunları yukarıda belirtilen sınıftan birine atamaktır.

4.1.2 Deneysel Ortam

Açık kaynak çerçeve olarak Flower çerçevesi, simülasyon özelliğiyle seçilmiştir. Uygulamaların gerçekleştirilmesinde, Python programlama dili tercih edilmiş ve kodlama ortamı olarak Visual Studio Code kullanılmıştır.

Python, Guido van Rossum tarafından ilk sürümü 1991'de ortaya konan genel amaçlı bir programlama dilidir. Diğer dillere göreli öğrenim kolaylığı ve geniş kütüphane desteğiyle oldukça yaygın kullanıcı kitlesine ulaşmıştır [58]. PYPL programlama dilleri popülerlik indeksine göre Aralık 2019-Aralık 2020 zaman aralığında Python dilinin birinci sırada olduğu gözlemlenmiştir [59].

PyTorch kullanarak, ResNet18 ağında önceden transfer eğitimi tamamlanmış CIFAR-10 veri kümesi üzerinde konvolüsyonel sinir ağının eğitimi gerçekleştirilmiştir. Torch üzerinden veri kümesi indirilerek normalizasyon işlemleri yapılmıştır. Akabinde konvansyonel sinir ağı üzerinde model eğitimi gerçekleştirimi yapılmıştır.

Flower çerçevesinde desteklenen federe öğrenme stratejilerinden FedAvg seçilmiş ve aşağıdaki parametrelere göre çalıştırılmıştır.

Veri noktası sayısı (batch_size)	20
Iterasyon sayısı (Epoch)	5
İstemci (Total Client)	500
Tur (Round)	4000

Tablo 4.1: Yapılan deneysel çalışmada kullanılan parametreler

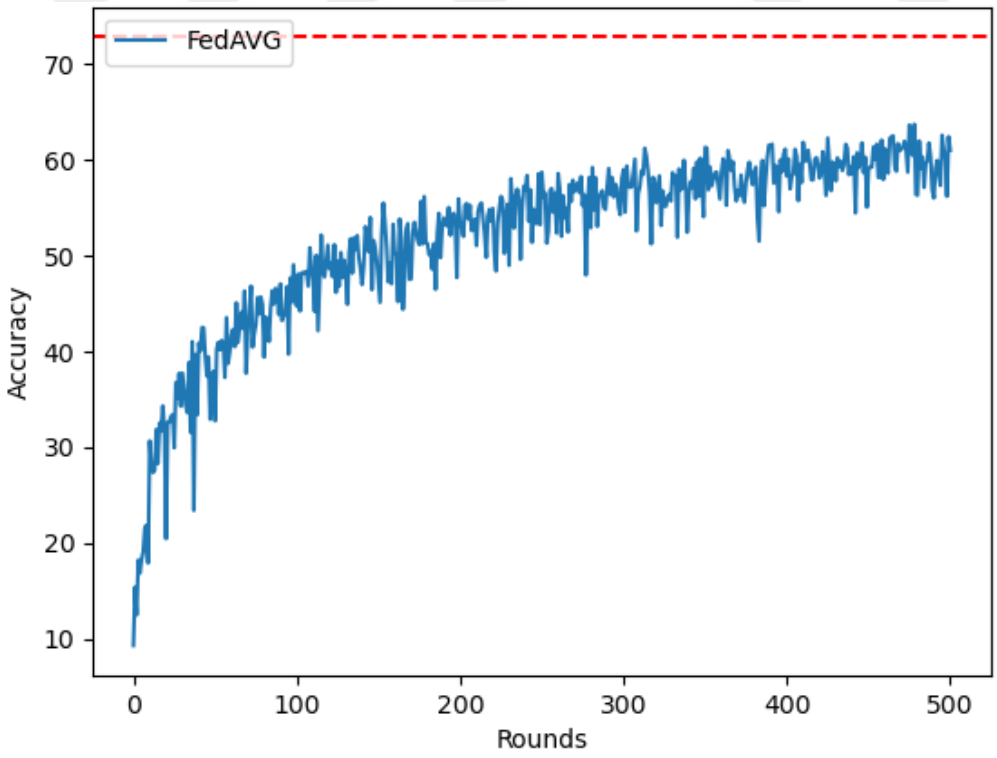
RAY, AI ve Python uygulamalarını ölçeklendirmesinde kullanılan açık kaynaklı simülasyon çerçevesidir [60]. RAY çerçevesi kullanan Flower üzerinde simülasyon ve eğitim gerçekleştirimi yapılmıştır.

Bölüm 5

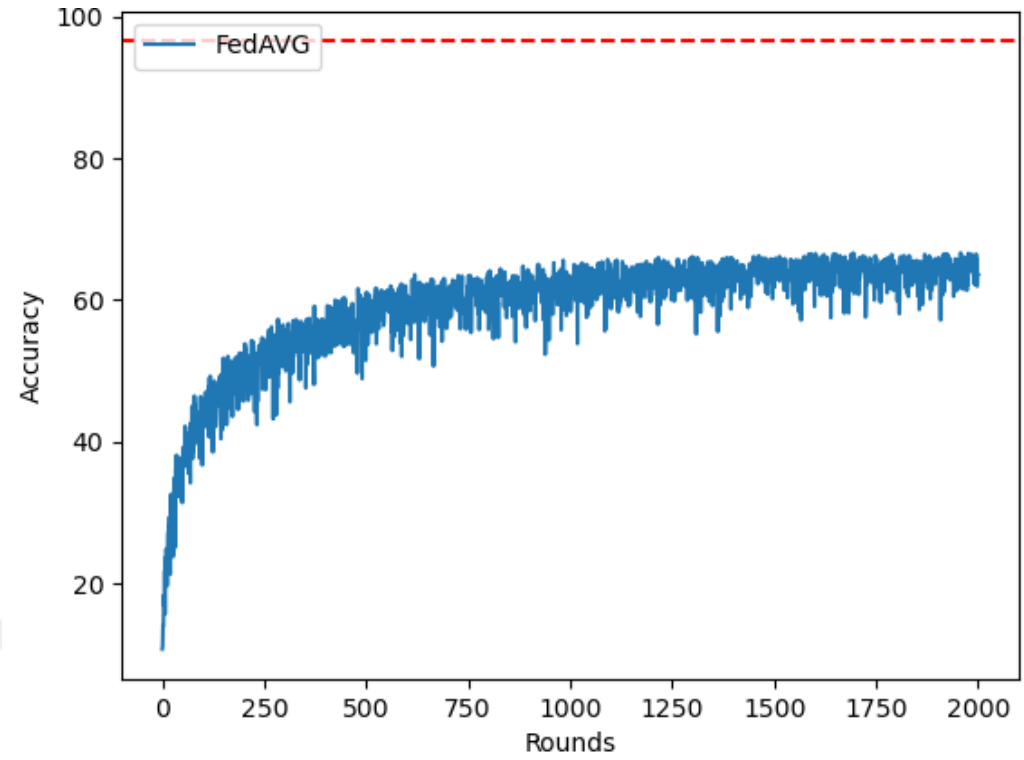
Tartışma ve Sonuç

CIFAR-10 seti üzerinde, FedAvg algoritması ile 10 sınıfı ayrılmış 32x32 ebatında resim, veri nokta sayısı (batch size) 50 olacak şekilde 100 istemciye dağıtılarak ve 500 eğitim turu gerçekleştiren McMahan vd. %96.5 deneysel sonuç elde etmiştir. McMahan vd. Aynı deneyde eğitim turunu 2000'e çıkartarak veri nokta sayısı ise 100'e çıkartıldığında doğruluk oranını %85'e düştüğünü belirtmiştir [1] .

Tezde yapılan çalışma sonucunda 100 istemci ve 500 iletişim turu sonucunda %62.61 sonuç elde edilmişken, tur 2000'e veri nokta sayısı 100'e çıkartıldığında ise doğruluk oranının %63.63 olduğu görülmüştür.

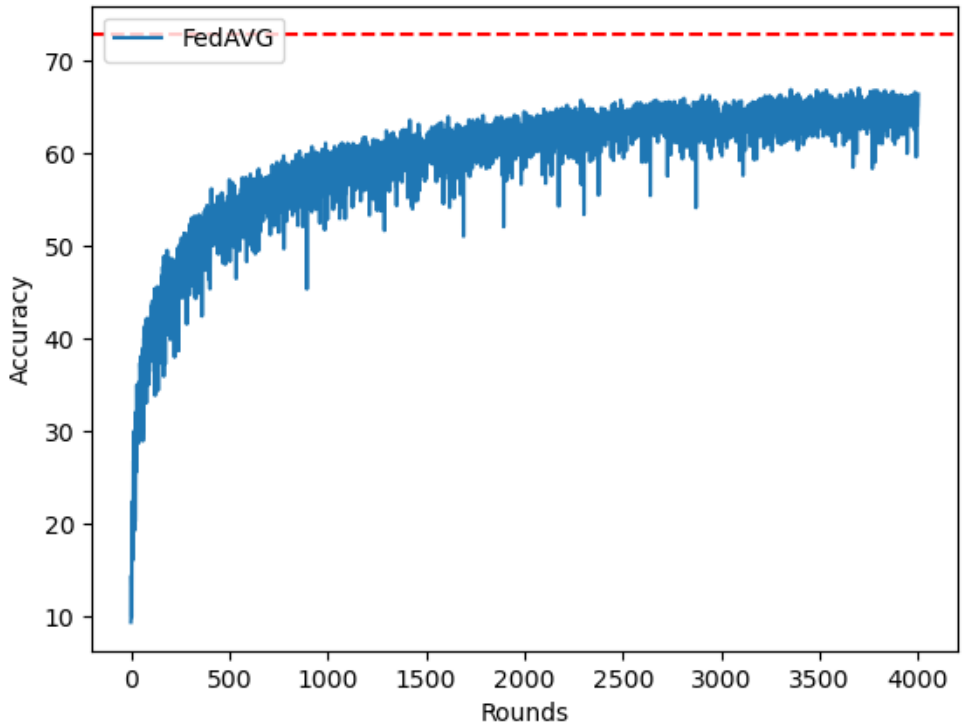


Şekil 5.14: 100 istemcinin 500 iletişim turu doğruluk oranları



Şekil 5.15: 100 istemcinin 2000 iletişim turu doğruluk oranları

Ayrıca, veri nokta sayısı (batch size) 20, 500 istemci, eğitim turu 4000 olmak üzere her seferinde rastgele 10 istemciden veri alınarak eğitim gerçekleştirilmiştir. Eğitim tamamlandığında ise doğruluk oranlarını %66.39'a ulaştığı Şekil 5.3'te gösterilmiştir.



Şekil 5.3: 500 istemcinin 4000 iletişim turu doğruluk oranları

FedProx için Flower yazarlarından Charles Beauville tarafından, MNIST veri kümesiyle yapılan "FL Starter Pack: FedAvg on MNIST using a CNN" başlıklı çalışmada [61] flower'ın asıl implementasyonla aynı sonuçlara (Şekil 3.12) ulaştığı gösterilmiştir.

Metot	İstemci Sayısı	Veri Nokta Sayısı	Eğitim Turu	Doğrulu Oranı
McMahan FedAVG	100	50	500	**%96,5**
McMahan FedAVG	500	20	4000	**%66,39**
Flower FedAVG	100	50	500	%62,61
Flower FedAVG	500	50	4000	%63,63

Tablo 5.1: Yapılan deneysel sonuçların karışlaştırılmalı doğruluk oranları

Sonuç itibariyle, Flower'ın kullanım kolaylığı yanı sıra gerek bu tezde yapılan deneyler gerekse farklı araştırmacılar tarafından yapılan ve açık kaynak kodları github adreslerinde [62] paylaşılan diğer araştırma sonuçları, gerçekleştirimleriyle benzer sonuçlara ulaşması itibariyle federe öğrenme gerçekleştirimi yapmak isteyen startup'lar ve federe öğrenme hakkında araştırma yapan araştırmacılar için ideal bir seçim olacağı düşünülmektedir.

Kaynaklar

[1] Federated Learning: Collaborative Machine Learning without Centralized Training Data [İnternet]. İzmir; 2017 [erişim tarihi 08.03.2022]. https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

[2] McMahan B, Moore E, Ramage D. vd. Communication-Efficient Learning of Deep Networks from Decentralized Data, the 20th International Conference on Artificial Intelligence and Statistics PMLR 54:1273-1282, 2017 Nisan 20-22 : https://proceedings.mlr.press/v54/mcmahan17a

[3] Yang, Q., Liu, Y., Chen, T., & Tong, Y. . Federated Machine Learning. ACM Transactions on Intelligent Systems and Technology (TIST), 2019, https://doi.org/10.1145/3298981

[4] Lian, Z., Yang, Q., Zeng, Q., & Su C., WebFed: Cross-platform Federated Learning Framework Based on Web Browser with Local Differential Privacy [İnternet]. İzmir, 2021 [erişim tarihi 01.02.2022]. https://arxiv.org/abs/2110.11646

[5] Lo, S.K., Lu, Q., Zhu, L., Paik, H., Xu, X., ve Wang, C. Architectural Patterns for the Design of Federated Learning Systems, Journal of Systems and Software 2022, https://doi.org/10.1016/j.jss.2022.111357

[6] Prof. Aaron Clauset , Inference, Models and Simulation for Complex Systems, Lecture 0, [İnternet]. İzmir, 2011 [erişim tarihi: 17.06.2022] https://aaronclauset.github.io/courses/7000/csci7000-001_2011_L0.pdf

[7] Brownlee J., A Gentle Introduction to Batch Normalization for Deep Neural Networks, [İnternet], İzmir, 2019, [erişim tarihi: 19.06.2022] https://machinelearningmastery.com/batch-normalization-for-training-of-deep-neural-networks/

[8] Kiknadze, M. & Gürhanlı, A. (2020). Yapay Sinir Ağı Kullanarak Meme Kanseri Hastalığının Tahmini . AURUM Journal of Engineering Systems and Architecture, 4 (2) , 255-272 . Retrieved from https://dergipark.org.tr/tr/pub/ajesa/issue/59133/675694

[9] Shokri, R. ve Shmatikov, V.. Privacy-Preserving Deep Learning. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15). Association for Computing Machinery, New York, NY, USA, 1310–1321. https://doi.org/10.1145/2810103.2813687

[10] Predicting Text Selections with Federated Learning [İnternet]. İzmir; 2021 [erişim tarihi 11.03.2022]. http://ai.googleblog.com/2021/11/predicting-text-selections-with.html

[11] Federated Learning with Formal Differential Privacy Guarantees [İnternet]. İzmir; 2022 [erişim tarihi 01.03.2022]. https://ai.googleblog.com/2022/02/federated-learning-with-formal.html

[12] Understanding Types of Federated Learning [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://blog.openmined.org/federated-learning-types

[13] Qiang Yang, Yang Liu, Tianjian Chen ve Yongxin Tong, Federated Machine Learning: Concept and Applications,2019, https://arxiv.org/pdf/1902.04885.pdf

[14] Federe Transfer öğrenme grafiği, [İnternet]. İzmir; 2020 [erişim tarihi 11.06.2022]. Şekil 5.2: https://www.mdpi.com/2076-3417/12/18/8980

[15] Süzen, A. A. & Kayaalp, K. (2019). Büyük verileride gizlilik tabanlı yaklaşım: Federe Öğrenme. International Journal of 3D Printing Technologies and Digital Industry. https://dergipark.org.tr/tr/pub/ij3dptdi/issue/51591/664020

[16] Chengliang Zhang, Suyi Li, Junzhe Xia, Wei Wang, Feng Yan vd., BatchCrypt: Efficient Homomorphic Encryption for Cross-Silo Federated Learning, 2020 https://usenix.org/system/files/atc20-zhang-chengliang.pdf

[17] Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., Bennis, M. vd. Advances and Open Problems in Federated Learning. 2021, *Foundations and Trends® in Machine Learning: Vol. 14: No. 1–2, pp 1-210. doi:10.1561/2200000083*

[18] T. Sun, D. Li and B. Wang, Decentralized Federated Averaging,IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022, doi: 10.1109/TPAMI.2022.3196503.

[19] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar and V. Smithy, "FedDANE: A Federated Newton-Type Method," 2019 53rd Asilomar Conference on Signals, Systems, and Computers, 2019, pp. 1227-1231, doi: 10.1109/IEEECONF44664.2019.9049023

[20] Y. Zhuo and B. Li, "Fedns: Improving Federated Learning for Collaborative Image Classification on Mobile Clients," 2021 IEEE International Conference on Multimedia and Expo (ICME), 2021, pp. 1-6, doi: 10.1109/ICME51207.2021.9428075

[21] Sahu, A., Li, T., Sanjabi, M., Zaheer, M., Talwalkar, A.S., & Smith, V. Federated Optimization in Heterogeneous Networks. Proceedings of Machine Learning and Systems 2 (MLSys 2020), 2020, doi.: 10.48550/arXiv.1812.06127

[22] Huang, W., Li, T., Wang, D., Du, S., ve Zhang, J. (). Fairness and Accuracy in Federated Learning [İnternet]. İzmir; 2020, [erişim tarihi 14.04.2022]. Doi: 10.48550/arXiv.2012.10069

[23] FedBN: Federated Learning on Non-IID Features via Local Batch Normalization, Li, X., Jiang, M., Zhang, X., Kamp, M., & Dou, Q. (2021), ArXiv, abs/2102.07623. Kaynak Kod: https://github.com/med-air/FedBN

[24] Federated Learning with Matched Averaging Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, Yasaman Khazaeni, 2019 ICLR 2020 Conference https://openreview.net/forum?id=BkluqlSFDS

[25] Reddi, S.J., Charles, Z.B., Zaheer, M., Garrett, Z., Rush, K., Konecný, J., Kumar, S., & McMahan, H.B. (2020). Adaptive Federated Optimization. ArXiv, abs/2003.00295

[26] Shoham, N., Avidor, T., Keren, A., Israel, N., Benditkis, D., Mor-Yosef, L., & Zeitak, I. (2019). Overcoming Forgetting in Federated Learning on Non-IID Data. ArXiv, abs/1910.07796

[27] Priyanka Mary Mammen , 2021, Federated Learning: Opportunities and Challenges: https://arxiv.org/abs/2101.05428

[28] Naseri, M., Hayes, J., & De Cristofaro, E. (2022). Local and Central Differential Privacy for Robustness and Privacy in Federated Learning. Proceedings 2022 Network and Distributed System Security Symposium. Doi: 10.48550/arXiv.2009.03561

[29] He, C., Li, S., So, J., Zhang, M., Wang, vd. FedML: A Research Library and Benchmark for Federated Machine Learning. 2020. doi: 10.48550/arXiv.2007.13518

[30] Publications, [İnternet]. İzmir; 2022 [erişim tarihi 20.04.2022]. https://fedml.ai/research-papers

[31] Lin, B., He, C., Zeng, Z., Wang, H., Huang, Y. vd. FedNLP: Benchmarking Federated Learning Methods for Natural Language Processing , 2022, Tasks. NAACL-HLT. Doi: 10.48550/arXiv.2104.08815

[32] Chaoyang He, Alay Dilipbhai Shah, Zhenheng Tang vd., FedCV: A Federated Learning Framework for Diverse Computer Vision Tasks, 2021, https://arxiv.org/pdf/2111.11066.pdf

[33] Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Carl Yang vd.,2021, FedGraphNN: A Federated Learning Benchmark System for Graph Neural Networks https://arxiv.org/pdf/2104.07145.pdf

[34] Tuo Zhang, Chaoyang He, Tianhao Ma, Lei Gao vd., 2021, Federated Learning for Internet of Things: A Federated Learning Framework for On-device Anomaly Data Detection https://arxiv.org/pdf/2106.07976.pdf

[35] G Anthony Reina, Alexey Gruzdev, Patrick Foley,vd., 2021, OpenFL: An open-source framework for Federated Learning https://arxiv.org/abs/2105.06413,

[36] Open Federated Learning (OpenFL) - An Open-Source Framework For Federated Learning [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://github.com/intel/openfl/blob/develop/README.md

[37] A high performance, open source universal RPC framework, [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. Grpc Protokolü: https://grpc.io/

[38] Transport Layer Security [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. TLS Protokolü: https://en.wikipedia.org/wiki/Transport Layer Security

[39] TensorFlow Federated: Merkezi Olmayan Verilerde Makine Öğrenimi [İnternet]. İzmir; 2020 [erişim tarihi 20.04.2022]. https://www.tensorflow.org/federated

[40] Federe Öğrenim Araştırması için TFF Kullanımı [İnternet]. İzmir; 2020 [erişim tarihi 22.04.2022]. https://www.tensorflow.org/federated/tff_for_research

[41] TensorFlow Federated Tutorials [İnternet]. İzmir; 2020 [erişim tarihi 22.04.2022]. https://github.com/tensorflow/federated/blob/main/docs/tutorials/tutorials_overview.md

[42] Composing Learning Algorithms [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://github.com/tensorflow/federated/blob/main/docs/tutorials/composing_learning_algorithms.ipynb

[43] Building Your Own Federated Learning Algorithm [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://github.com/tensorflow/federated/blob/main/docs/tutorials/building_your_own_federated_learning_algorithm.ipynb

[44] NVIDIA Clara [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://www.nvidia.com/en-us/clara

[45] OpenMined Syft [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://github.com/OpenMined/PySyft

[46] OpenMined PyGrid [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://github.com/OpenMined/PyGrid

[47] OpenMined PyGrid Admin [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://github.com/OpenMined/pygrid-admin

[48] Federated Learning on web and mobile devices [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://blog.openmined.org/announcing-new-libraries-for-fl-on-web-and-mobile

[49] Zhuotao Lian, Qinglin Yang, Qingkui Zeng, Chunhua Su vd., 2021 WebFed: Cross-platform Federated Learning Framework Based on Web Browser with Local Differential Privacy , https://arxiv.org/abs/2110.11646

[50] LEAF: A Benchmark for Federated Settings, Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Workshop on Federated Learning for Data Privacy and Confidentiality (2019). Arxiv, abs/1812.01097

[51] Hugging Face Topluluğu, Transformers kütüphanesi [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. Hugging Face Transformans erişim adresi: https://huggingface.co/docs/transformers/index

[52] Beutel, Daniel vd. "Flower: A Friendly Federated Learning Research Framework,2020, ArXiv abs/2007.14390

[53] FL Başlangıç Paketi: CNN kullanarak MNIST üzerinde FedAvg çalışmasının kaynak kodları, [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://github.com/adap/flower/tree/main/baselines/flwr_baselines/publications/FedAvg_mnist

[54] FL Başlangıç Paketi: FedAvg [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://flower.dev/blog/2023-01-12-fl-starter-pack-FedAvg-mnist-cnn/

[55] FL Başlangıç Paketi: CNN kullanarak MNIST üzerinde FedProx çalışma kodu, [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022].

https://github.com/adap/flower/tree/main/baselines/flwr_baselines/publications/fedprox_mnist

[56] Krizhevsky A. Learning multiple layers of features from tiny images, [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://www.cs.toronto.edu/~kriz/cifar.html

[57] Kaiming He, Xiangyu Zhang, Shaoqing Ren ve Jian Sun, "Deep Residual Learning for Image Recognition." 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015): 770-778. https://arxiv.org/abs/1512.03385

[58] Malkoç B.(2012), "Temel Bilimler ve Mühendislik Eğitiminde Programlama Dili Olarak Python", XIV. Akademik Bilişim Konferansı Bildirileri, 201.

[59] PYPL, programlama dilleri popülerlik indeksi [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://pypl.github.io/PYPL.html

[60] RAY, AI ve Python uygulamalarını ölçeklendirmede kullanılan açık kaynaklı çerçeve,[İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://www.ray.io/

[61] Charles Beauville,2023, FL Starter Pack: FedAvg on MNIST using a CNN, [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://flower.dev/blog/2023-01-12-fl-starter-pack-FedAvg-mnist-cnn/

[62] Flower, akadamik yayınlar karşılaştırmalı deneylerin kaynak kodları url adresi: [İnternet]. İzmir; 2020 [erişim tarihi 11.04.2022]. https://github.com/adap/flower/tree/main/baselines

Ekler

Ek A

Tezden Üretilmiş Yayınlar

Konferans Bildirileri

1. Federe Öğrenme Algoritmaları ve Açık Kaynak Çerçeveler, 5th International Conference on Applied Engineering and Natural Sciences, July 10-12, 2023 : Konya, Turkey

Özgeçmiş

Adı Soyadı: Ömer Faruk Göçgün

Eğitim:

2020– Devam	İzmir Kâtip Çelebi Üniversitesi, Yazılım Müh. Y.L. Öğrencisi
2019–2021	Anadolu Üniversitesi, Yönetim Bilişim Sistemleri
2017–2019	Dokuz Eylül Üniversitesi, İMYO, Programcılık
2010–2015	İzmir Ekonomi Üniversitesi, İşletme

İş Deneyimi:

2022 – Halen	Plan A Technologies, Yazılım Mühendisi
2020 – 2022	Sertom Yazılım Ltd., Ekip Lideri
2019 – 2020	Sertom Yazılım Ltd., Yazılım Geliştirme Uzmanı
2019 – 2019	Innosa Bilişim A.Ş., ARGE Yazılım Geliştirme Uzmanı
2016 – 2019	SmmmEgitimi.com, Yazılım Geliştirme Uzmanı
2009 – 2017	Metatron İnşaat Ltd. Şti., Genel Müdür
2004 – 2007	Serbest, Web Yazılım Geliştirme Uzmanı

Yayınlar (varsa):

1. Göçgün, Ö. F. & Onan, A. (2021). Amazon Ürün Değerlendirmeleri Üzerinde Derin Öğrenme/Makine Öğrenmesi Tabanlı Duygu Analizi Yapılması . Avrupa Bilim ve Teknoloji Dergisi, Ejosat Özel Sayı 2021 (ARACONF) , 445-448 . DOI: 10.31590/ejosat.902674

2. Öztürk N., Özdağlar D. , Göçgün Ö.F., Karadağ D. (2019) Çevre izin ve lisans süreç yazılımı. Syf 567-575, 13.Ulusal 1. Uluslararası Çevre Müh. Kongresi, Gebze-Kocaeli.