T.C.
SAKARYA UYGULAMALI BİLİMLER ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

BEYİN MR GÖRÜNTÜLERİNDE GİZLİLİK TABANLI YAKLAŞIM : FEDERE ÖĞRENME

YÜKSEK LİSANS TEZİ

Şevket AY

Enstitü Anabilim Dalı	**:**	**BİLGİSAYAR MÜHENDİSLİĞİ**
Tez Danışmanı	**:**	**Dr. Öğr. Üyesi Ekin EKİNCİ**
Ortak Danışman	**:**	**Dr. Öğr. Üyesi Zeynep GARİP**

Haziran 2023

T.C.
SAKARYA UYGULAMALI BİLİMLER ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

BEYİN MR GÖRÜNTÜLERİNDE GİZLİLİK TABANLI YAKLAŞIM : FEDERE ÖĞRENME

YÜKSEK LİSANS TEZİ

Şevket AY

Enstitü Anabilim Dalı : BİLGİSAYAR MÜHENDİSLİĞİ

Bu tez 20/07/2023 tarihinde aşağıdaki jüri tarafından oybirliği ile kabul edilmiştir.

JÜRİ	BAŞARI DURUMU
Jüri Başkanı: Dr. Öğr. Üyesi Ekin EKİNCİ	BAŞARILI
Üye: Dr. Öğr. Üyesi Zeynep GARİP	BAŞARILI
Üye: Doç. Dr. Halit ÖZTEKİN	BAŞARILI
Üye: Dr. Öğr. Üyesi Kasım SERBEST	BAŞARILI
Üye: Dr. Öğr. Üyesi Fidan Kaya GÜLAĞIZ	BAŞARILI

BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim

Şevket AY

20/07/2023

TEŞEKKÜR

Bu tez çalışması, federe öğrenme ile veri gizliliğini koruyarak ortak bir model geliştirme amacıyla gerçekleştirilmiştir.

Yüksek lisans eğitimim boyunca değerli bilgi ve deneyimlerinden yararlandığım, her konuda bilgi ve desteğini almaktan çekinmediğim, araştırmanın planlanmasından yazılmasına kadar tüm aşamalarında yardımlarını esirgemeyen, teşvik eden, aynı titizlikte beni yönlendiren değerli danışman hocam Dr. Öğr. Üyesi Ekin EKİNCİ'ye ve ortak danışmanlığımı yürüten Dr. Öğr. Üyesi Zeynep GARİP'e teşekkürlerimi sunarım.

İÇİNDEKİLER

KISALTMALAR

AI : Artificial Intelligence

CNN : Convolutional Neural Network (Evrişimli Sinir Ağı)

DCNN : Deep Convolutional Neural Network (Derin Evrişimli Sinir Ağı)

FedAvg : Federe Ortalama

FedML : Federated Machine Learning

FedOpt : Federated Optimization

FedSDG : Federe Stokastik Gradyan Düşüşü

FATE : Federe AI Teknolojisi Etkinleştirici

FHE : Fully Homomorphic Encryption (Tamamen Homomorfik Şifreleme)

FT – FedAvg : Hata Toleranslı Federe Ortalama

gRPC : Google Remote Procedure Call

IID : Independent and Identically Distributed (Bağımsız ve Özdeş Dağıtılmış)

KNN : K-Nearest Neighbour (En Yakın Komşu)

MRI : Manyetik Rezonans Görüntüleri

MFL : Microsoft Federe Öğrenim

NB : Naive Bayes

NTRU : Nth Degree Truncated Polynomial Ring Units

PHE : Partially Homomorphic Encryption (Kısmen Homomorfik Şifreleme)

ReLU : Rectified Linear Unit (Doğrultulmuş Doğrusal Birim)

RSA : Rivest-Shamir-Adleman

SGD	: Stokastik Gradient Descent (Stokastik Gradyan İnişi)
SVM	: Support Vector Machines (Destek Vektör Makineleri)
TFF	: TensorFlow Federe
TLS	: Transport Layer Security (Aktarım Katmanı Güvenliği)
XgBoost	: Extreme Gradient Boosting (Aşırı Gradyan Arttırma)
QFedAvg	: Kuantum Federe Ortalama

SİMGELER

b	: Dağılımın yayılma kontrolü
C	: Şifreli metin
D	: Veri kümesi
δ	: M algoritmasının diferansiyel gizliliği sağlamakta başarısız olma ölçüsü
E	: Şifreleme işlevi
ε	: Verilere eklenen gürültü parametresi
e	: e sayısı
e_i	: Her model ağırlığına eklenen ek gürültü
F	: Filtrenin boyutu
g	: Temsili sinir ağı
h	: Gizli temsil
k	: Değişken
K	: Değişken
L1	: Cezalandırma terimi
L2	: Cezalandırma terimi
λ	: Düzenlileştirme gücü
m	: Metin
μ	: Dağılımın merkezi
n	: Değişken
O	: Çıkış hacminin boyutu
p	: Dolgu miktarı
p_i	: i. istemcinin verilerinin olasılık dağılımı
q	: Referans dağılımı
q_i	: Veri dağılımı ile bir referans dağılımı arasındaki Kullback-Leibler (KL) sapmasına dayalı olarak i. istemciye atanan ağırlık
S	: Adım

s(i)	: Düğüm
t	: t yinelemesindeki global model
w	: Ağdaki ağırlık parametresi
w_g	: Global model ağırlıkları
w_i	: İstemci i'ye atanan ağırlık
w_i^t	: t yinelemesindeki i aygıtındaki yerel model
$w_{i,s(i)}^t$	: s(i) düğümünün modeli
W	: Giriş hacminin boyutu
w_l	: Yerel model ağırlıkları
w^t	: t yinelemesindeki global model
x	: Değişken
$x_{(i-1)S+p,(j-1)S+q,k}$	: k'ıncı özellik haritasındaki (i-1)S+p,(j-1) konumundaki giriş değeri
$\gamma^{(j)}$	:Öğrenilebilir ölçek
$\beta^{(j)}$	:Kaydırma parametresi
η	: Öğrenme oranı
$\eta\nabla f_i(w_i^t)$	: Yerel gradyan
λ	: Yakın düzenleme sabiti
$\widehat{x_{i,J}}$	: i'nci istemcinin j'inci özelliğinin normalleştirilmiş girişi
ϵ	: Verilere eklenen gürültü parametresi
δ	: M algoritmasının diferansiyel gizliliği sağlamakta başarısız olma ölçüsü
N	: Toplam istemci sayısı
y	: Değişken
$y_{i,j,k}$	: k. özellik haritasının çıktı değeri
y_{hat}	: Nihai çıktı
z'	: Model ağırlıkları ve özellik değerlerinin şifrelenmiş nokta çarpımı

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

BEYİN MR GÖRÜNTÜLERİNDE GİZLİLİK TABANLI YAKLAŞIM : FEDERE ÖĞRENME

ÖZET

Güçlü derin öğrenme modelleri, çok fazla veriye ve veri çeşitliliğine ihtiyaç duymaktadır. Tıp alanı modellerin kullanımı açısından en popüler olan alanlardan bir tanesidir. Tıp alanında çok fazla veri vardır ancak gizlilik nedeniyle tıbbi verilere erişim oldukça sınırlıdır. Bu durum sağlık sektöründeki AI çözümlerinin gelişimini yavaşlatmakta ve olumsuz anlamda etkilemektedir. Gizlilik sorununu ele alan bir yaklaşım federe öğrenmedir. Federe öğrenme, kurumların verilerini dışarıya çıkarmadan diğer kurumlar ile işbirlikçi model gelişimini sağlayarak veri gizliliğini korumaktadır böylece derin öğrenme modellerinin geliştirilmesine katkıda bulunabilmektedir. Bu çalışmada, Federe öğrenme ile tıp alanındaki mahremiyet sorununa odaklanılmıştır. Beyin tümörü görüntülerini sınıflandırmak amacıyla görüntü sınıflandırma problemlerinde en çok tercih edilen derin öğrenme ağı olan Evrişimli Sinir Ağı'na (CNN) dayalı bir model federe öğrenme ile entegre edilmiştir. Yerel CNN modelleri, simule edilen üç farklı kuruluşun yerel cihazlarının Manyetik Rezonans Görüntüleri (MRI) kullanılarak başlatılmıştır. Sunucuda bulunan global CNN modeli yerel CNN modellerinin ağırlıkları ile beslenmektedir. Yerel model ağırlıkları sunucuda optimize edilen yerel model ağırlığı ile güncellenmiştir. Federe Ortalama (FedAvg), FedAvg ile kuantum Federe ortalama (QFedAvg), Hata Toleranslı FedAvg ve diferansiyel gizlilik ile birlikte FedAvg olmak üzere dört farklı Federe öğrenme stratejisi kullanılmıştır. FedAvg olarak adlandırılan ilk stratejide üç farklı istemciden gelen model ağırlıkları sunucuda model ağırlıklarının ortalaması alınarak optimize edilmektedir. İkinci strateji olan QFedAvg yönteminde ise standart FedAvg yöntemine bir q parametresi eklenmektedir. Bu parametre ile istemciler arası olabilecek farklılıklar toplanan model ağırlıklarının optimize edilmesinde dikkate alınmaktadır. Üçüncü strateji ise Hata Toleranslı FedAvg stratejisidir. Hata toleranslı yaklaşım eğitim sürecinde herhangi bir sebepten dolayı bir istemcide oluşabilecek hata veya kopmaların tolere edilmesini içermektedir. Son stratejide ise sunucuda standart FedAvg model ağırlıklarını optimize ederken kullanılmıştır fakat ilk yöntemden farklı olarak sunucuya model ağırlığı gönderilmeden önce bir istemcinin model ağırlıklarına diferansiyel gizlilik ile gürültü eklenmiştir. Dördüncü stratejinin amacı her ne kadar sadece model ağırlıklarının iletilmesi olsa da olası herhangi bir saldırıda model ağırlıklarının elde edilmesini engellemektir. Her bir strateji 10 turda eğitilmiş ve en başarılı stratejinin FedAvg olduğu görülmüştür. FedAvg'e göre son turda eğitilen modelin ilk turda eğitilen modele göre %41 daha başarılı olduğu kanıtlanmıştır. Önerilen tüm stratejiler kullanılarak yerel verilerin istemcilerde kalması sağlanarak istemciler arası ortak bir model geliştirilmiştir. Gizliliğin korunmadığı ve verilerin bir sunucuda toplanarak eğitim yapıldığı klasik derin öğrenme modellerine kıyasla hem gizlilik korunmuş hem de istemcilerin daha az yerel veri ile yüksek model başarımı elde etmesi sağlanmıştır.

Anahtar Kelimeler: Derin Öğrenme, Federe Öğrenme, Gizlilik, Sınıflandırma

PRIVACY-BASED APPROACH TO BRAIN MRI: FEDERATED LEARNING

ABSTRACT

Powerful deep learning models need a lot of data and data diversity. The medical field is one of the most popular fields in terms of the use of models. There is a lot of data in the medical field, but access to medical data is very limited due to confidentiality. This slows down and negatively affects the development of AI solutions in the healthcare industry. One approach to addressing the privacy issue is federated learning. Federated learning protects data confidentiality by providing collaborative model development with other institutions without exposing the data of institutions, so it can contribute to the development of deep learning models. This study focuses on the problem of privacy in the medical field with Federated learning. In order to classify brain tumor images, a model based on Convolutional Neural Network (CNN), which is the most preferred deep learning network in image classification problems, is integrated with federated learning. Local CNN models were initiated using Magnetic Resonance Imaging (MRI) of local devices from three different organizations simulated. The global CNN model on the server is fed with the weights of the local CNN models. The local model weights have been updated with the local model weight optimized on the server. Four different Federated learning strategies were used: Federated Average (FedAvg), FedAvg with quantum Federated average (QFedAvg), FedAvg with Fault Tolerance, and FedAvg with differential privacy. In the first strategy, called FedAvg, the model weights from three different clients are optimized by averaging the model weights on the server. In the second strategy, the QFedAvg method, a q parameter is added to the standard FedAvg method. With this parameter, differences between clients are taken into account in optimizing the collected model weights. The third strategy is the Fault Tolerant FedAvg strategy. This approach includes tolerating errors or breaks that may occur in a client for any reason during the training process. In the last strategy, the standard FedAvg model weights are used on the server, but unlike the first method, differential privacy and noise are added to the model weights of a client before the model weight is sent to the server. The goal of the fourth strategy is to prevent model weights from being acquired in any possible attack, although only the model weights are transmitted. Each strategy was trained for 10 rounds and it was seen that the most successful strategy was FedAvg. According to FedAvg, the model trained in the last round proved to be 41% more successful than the model trained in the first round. A common client-to-client model has been developed by keeping the local data on the clients by using all the suggested strategies. Compared to classical deep learning models, where confidentiality is not protected and data is collected and trained on a server, both confidentiality is preserved and clients are provided with high model performance with less local data.

Keywords: Classification, Deep Learning, Federated Learning, Privacy

BÖLÜM 1. GİRİŞ

Beyin tümörleri beyin dokusunda meydana gelen iyi huylu ve kötü huylu anormal büyümelerdir. Beyindeki konumlarına, davranışlarına ve hücre tiplerine göre birkaç çeşit beyin tumörü vardır. Gliomalar, beyindeki nöronları destekleyen ve koruyan hücreler olan glial hücrelerden gelişen tümörlerdir. Meningiomalar, beyni ve omuriliği kaplayan koruyucu tabakalar olan meninkslerde gelişen tümörlerdir. Pituitary tümörleri, beynin tabanında yer alan ve vücudun hormonal sistemini düzenleyen hipofiz bezinde gelişen tümörlerdir (Sultan ve diğ., 2019).

Son yıllarda, CNN gibi derin öğrenme modellerinin kullanımı, beyin tümörlerinin sınıflandırılması da dahil olmak üzere tıbbi görüntüleme analizi alanında büyük umut vaat etmektedir. Bununla birlikte, merkezi derin öğrenme yöntemlerinin kullanımı, mahremiyet endişeleri ve hassas tıbbi verilerin kurumlar arasında paylaşılması gereği nedeniyle genellikle sınırlıdır. Tıbbi veriler AI (AI) gelişimi için önemli verilerdir fakat aynı zamanda elde edilmesi zor olan verilerdir. Bunun sebebi olarak hasta bilgilerinin etik nedenlerle üçüncü kurum veya kişilerle paylaşılamamasıdır. Tıbbi veri gizliliği AI için kritik öneme sahiptir. Bunlardan ilki hasta mahremiyetinin korunmasının gerekliliğidir. Tıpkı geleneksel sağlık hizmetlerinde olduğu gibi, AI uygulamalarında da hastaların mahremiyetini korumak için gizlilik esastır. AI modellerinde kullanılan hasta verileri, tıbbi geçmiş, genetik veriler ve gizli tutulması gereken diğer kişisel ayrıntılar gibi hassas bilgileri içerebilmektedir. İkinci kritik sebep, etik standartların sürdürülmesidir. AI uygulamalarının geliştirilmesini ve etik kullanımını sağlamak için gizliliğin korunması esastır. AI modelleri, hasta özerkliğine saygı gösterecek ve mahremiyet hakkı da dahil olmak üzere haklarını koruyacak şekilde geliştirilmeli ve kullanılmalıdır. Üçüncü kritik nokta ise önyargı ve ayrımcılığın önlenmesidir. Gizli tıbbi veriler kullanılarak geliştirilen AI modelleri, önyargı ve ayrımcılığı önleyecek şekilde kullanılmalıdır. Gizlilik, bireylere veya gruplara karşı ayrımcılık yapmak

amacıyla tıbbi verilerin kötüye kullanılmasının önlenmesine katkıda bulunabilir. Dördüncü kritik sebep ise yasal gerekliliklere uygunluktur.

AI uygulamalarında tıbbi verilerin işlenmesi için birçok yasal gereklilik vardır. Gerekliliklere uymak ve yasal sonuçlardan kaçınmak için gizlilik esastır. Özetle, tıbbi veri gizliliği, AI için oldukça önemlidir. Sağlık hizmeti sağlayıcıları ve AI geliştiricileri, hasta mahremiyetini korumak, etik standartları sürdürmek, önyargı ve ayrımcılığı önlemek ve yasal gerekliliklere uymak için hasta verilerinin gizliliğini sağlamak için gerekli olan tüm önlemleri almalıdır. Şimdiye kadar, makine öğrenmesi ve derin öğrenmede beyin tümörü sınıflandırma çalışmalarında gizliliğin göz ardı edildiği görülmüştür (Bartoletti ve diğ., 2019; Hao ve diğ., 2019).

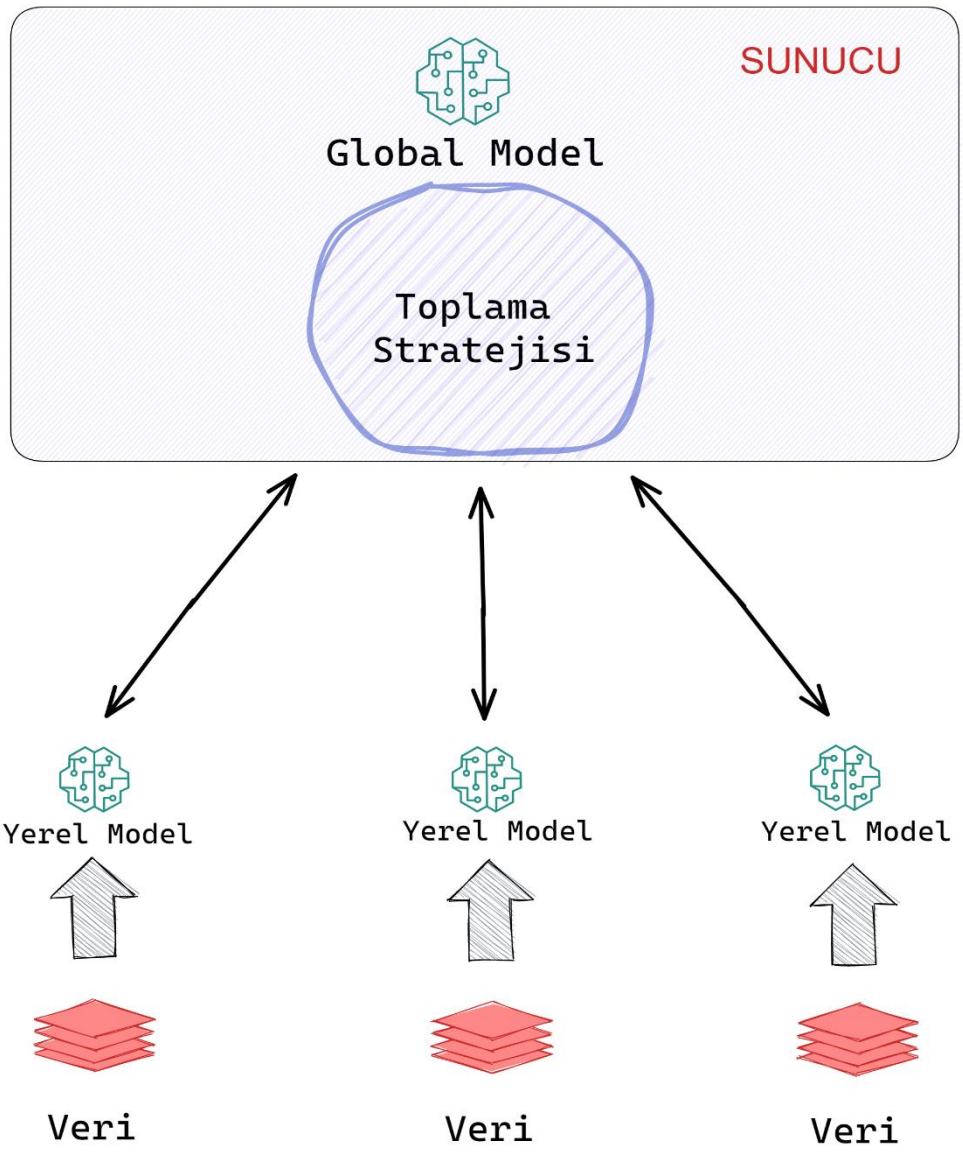
Gizliliği göz ardı eden çalışmalar verileri geleneksel bir şekilde işleyerek modellerini geliştirmiştir. Dmytro Filatov ve diğ. ResNet50, EfficientNetB1, EfficientNetB7, EfficientNetV2B1 modellerini çalışmalarında uygulamış ve EfficientNetB1 modelini önermiş ve %87,67 performans elde etmişlerdir. Gomez ve diğ. çalışmalarında InceptionV3 modelini önermektedir ve modelin performansı %97,12'dir. Saikat ve diğ. çalışmalarında %92 performans ile VGG16 modelini önermektedir. Sadoon ve diğ. çalışmalarında CNN modeli ile %96 performans göstermiştir. Rasheed ve diğ. geliştirdikleri CNN modeli ile %98'lik bir model performansı yakalamış ve bu modelin performansını VGG16, VGG19, ResNet50 ve InceptionV3 algoritmalarıyla karşılaştırmışlardır. Sarkar ve diğ. yaptıkları çalışmada CNN+BayesNet, AlexNet CNN + NB modellerini kullanmışlardır. El-Wahab ve diğ. BTC-fCNN ismini verdikleri model ile %98.63'lük bir performans sergilemişlerdir. Saeed ve diğ. özelleştirilmiş bir CNN modeli ile %93.60'lık bir başarım elde etmişlerdir. Ankit ve diğ. yaptıkları çalışmada SVM, KNN, Random Forest ve XGBoost modellerini sınıflandırma için kullanmışlardır ve %90 performans ile XGBoost en başarılı model olmuştur. Malla ve diğ. DCNN modeli ile %98.93 oranında performans elde etmişlerdir. Tiwari ve diğ. CNN tabanlı geliştirdikleri model ile %99 performans gösterdiklerini belirtmişlerdir. Mahesh ve diğ. kontur çıkarma odaklı geliştirdikleri EfficientNet-B0 modeli ile %96.86 F1 skor elde etmişlerdir. Mandle ve diğ. VGG19 modeli ile %99.83 oranında bir model performansı elde etmişlerdir. Raza ve diğ. DeepTumorNet adını verdikleri model ile %99.66 F1 skor elde etmişlerdir. Şimdiye kadar bahsedilen tüm çalışmalar, hem veri depolamayı hem de eğitim modellerini yöneten merkezi bir sunucunun veya bir bilgi

işlem cihazının bulunduğu geleneksel makine öğrenimi kullanılarak yapılmıştır. Bu nedenle eğitim öncesi tüm verilerin merkezi bir yerde toplanması gerekmektedir. Çalışmanın temel amacı model performansından ziyade veri gizliliğini korumak olduğundan geleneksel modellerin performanslarını karşılaştırmak uygun değildir. Federe öğrenme yapısı kullanılarak beyin tumörü sınıflandırması yapılan çalışmalar da vardır. Islam ve diğ. Birleşik Krallık veri kümesinden 22 beyin tumörü hastasının verileri ile çalışmışlardır. Çalışmalarında tumör var ve tumör yok olarak iki farklı sınıf vardır. DenseNet121, VGG19 ve InceptionV3 modellerinden ortalama bir CNN modeli oluşturarak federe öğrenme yapısı içine entegre etmişlerdir. Çalışmaları sonucunda %91.05'lik bir doğruluk elde etmişlerdir. Viet ve diğ., tez çalışmamız kapsamında kullandığımız veri kümesi içerisinden 3064 adet örneği alarak ConvXNet modeli geliştirmişlerdir ve bu modeli federe öğrenme yapısı içerisinde kullanarak %98'lik bir başarı elde etmişlerdir. Çalışma kapsamında yapılan beyin tümörü sınıflandırma çalışmasında federe öğrenme önerilmiş ve federe öğrenme sayesinde kritik nedenlerin gizlilik gereksinimleri karşılanmıştır.

Gizlilik sorununu ele almak amacıyla federe öğrenme umut verici bir alternatif çalışma olarak ortaya çıkmıştır ve verileri paylaşmaya gerek kalmadan birden çok kaynaktan gelen verileri kullanarak makine öğrenmesi modellerinin geliştirilmesini sağlamıştır. Federe öğrenme sayesinde MRI görüntülerinin hasta bilgilerini içeren meta bilgileri de kuruluşlardan dışarı çıkmamaktadır. Meta bilgiler, hastanın adı, yaşı gibi kişisel bilgileri içermektedir. Federe öğrenme, algoritmaların veri gizliliği sağlayarak işbirliği içinde öğrenmesini sağlayan bir öğrenme yöntemidir (Zhang ve diğ., 2021). Diğer bir deyişle, bir makine öğrenmesi modelinin, veri alışverişi yapılmadan, birçok bağımsız cihazdan elde edilen yerel verilerle bir sunucu üzerinde işbirlikçi eğitimidir.

Federe öğrenme, tıbbi verileri bulunduğu kurumun dışına çıkarmadan işbirlikçi modeller aracılığıyla tahminler yapılmasını sağlamaktadır. Model eğitim süreci, katılan her kurumda yerel olarak gerçekleştirilmektedir. Yerel model ağırlıkları daha sonra sunucu olarak kullanılacak cihazdaki model üzerinde kullanılmak üzere sunucuya iletilmektedir. Tüm merkezi olmayan süreçte karşılaşılan bazı zorluklar da vardır. Bunlar; istatistiksel heterojenlik, veri gizliliği ve dağıtılmış optimizasyondur. Bu çalışmada çeşitli optimizasyon yöntemleri aracılığıyla dağıtık optimizasyon ve diferansiyel gizlilik yoluyla veri gizliliği üzerine odaklanılmıştır. Dağıtılmış

optimizasyon, yerel modellerin yerel veriler üzerinde eğitilmesidir, merkezi sunucu ise küresel güncellemelerin toplanmasından sorumludur. Merkezi sunucuda toplanan güncellemeler optimize edilmelidir. Şekil 1.1'de federe öğrenmenin genel işleyişi gösterilmiştir.



Şekil 1.1: Genel federe öğrenme süreci

Çalışma kapsamında, Federe öğrenmenin sağladığı gizlilik avantajından yararlanarak işbirlikçi model geliştirmek için Kaggle'da (URL-1, 2021) bulunan beyin tümörü veri kümesi kullanılmıştır. Veri kümesi federe öğrenme için dört parçaya (3 yerel, bir sunucu) bölünmüştür.

Federe öğrenme yaklaşımında, eğitim süreci her biri istemci olarak hareket eden üç istemciye dağıtılmıştır ve merkezi bir sunucu eğitim sürecini koordine etmiştir. İstemci cihazlar, CNN modelini lokal veri setlerinde eğitmiştir ve güncel ağırlıkları merkezi sunucuya göndermiştir. Merkezi sunucu daha sonra ağırlıkları toplamıştır ve global modeli güncellemiştir. Daha sonra güncellenen model ağırlıklarını daha da geliştirmek için istemci cihazlara geri göndermiştir. Bu süreç belirlenen tur sayısı kadar devam etmiştir.

Simüle edilen her cihazda yerel veri miktarı eşit olduğundan, çalışmada FedAvg, FedAvg ile kuantum Federe öğrenme ve Hata Toleranslı FedAvg stratejileri kullanılmıştır. Veri gizliliğinin etkilerini görmek için sonuçlar arasındaki farkı gözlemlemek adına en başarılı strateji olan FedAvg stratejisine diferansiyel gizlilik yöntemi uygulanmıştır. Yapılan deneyler sonucunda verilerin cihazlar arasında paylaşılmadan veya bir sunucuda toplanmadan cihazlar arası işbirliği içerisinde model geliştirilebildiği ve ilgili stratejilerden FedAvg'ın en başarılı strateji olduğu görülmüştür.

Sonuç olarak, başarılı bir tedavi planlaması için beyin tümörlerinin doğru sınıflandırılması çok önemlidir ve makine öğrenmesi modellerinin kullanımı sağlık alanında büyük potansiyel göstermiştir. Bununla birlikte, tıbbi verilerin paylaşılmasıyla ilgili gizlilik endişeleri, geleneksel makine öğrenmesi yöntemlerinin uygulanmasını sınırlamıştır. Federe öğrenme, veri gizliliğini ve güvenliğini korurken birden çok kaynaktan gelen verileri kullanan makine öğrenmesi modellerinin geliştirilmesine izin vererek bu soruna umut verici bir çözüm sunmaktadır.

Tez çalışmasının birinci bölümünde beyin tumörü sınıflandırma için kullanılan geleneksel makine öğrenmesi modelleri ve Federe öğrenmenin neden geleneksel makine öğrenmesi yöntemlerinin yerine kullanılması gerektiği anlatılmıştır. İkinci bölümde Derin Öğrenme konularından bahsedilecektir. Üçüncü bölümde Federe Öğrenme bileşenlerinden şifreleme yöntemleri, haberleşme mimarileri, iletişim protokolü, federe öğrenme türleri, optimizasyon algoritmaları, veri bölümleme ve sıkıştırma metotlarından detaylıca anlatılacaktır. Ek olarak federe öğrenme avantaj ve dezavantajlarından, federe öğrenme çerçeveleri anlatılacaktır. Dördüncü bölümde; yapılan deneysel çalışmalar, önerilen yöntemlerin birbiri ile karşılaştırılması ve elde edilen sonuçların niceliksel ve niteliksel değerlendirilmesi yapılacaktır. Sonuçlar ve

öneriler bölümünde, elde edilen sonuçlar yorumlanacak ve çalışmanın katkıları tartışılacaktır. Ayrıca gelecek için yapılabilecek çalışmalar için önerilerde bulunulacaktır.

BÖLÜM 2. DERİN ÖĞRENME

2.1. Derin Öğrenme

Derin öğrenme, görüntü ve konuşma tanıma, doğal dil işleme ve karar verme gibi karmaşık görevleri yerine getirme becerisi nedeniyle son yıllarda popülerlik kazanan makine öğrenmesinin bir alt kümesidir (Janiesch ve diğ., 2021). Derin öğrenme algoritmaları, insan beyninin işleyişini simüle eden yapay sinir ağlarına dayanmaktadır. Derin öğrenme algoritmalarının temel özelliklerinden biri, büyük miktarda veriden öğrenme yetenekleridir. Derin öğrenme algoritmasına veri kümesinin girdisinin sağlandığı ve tahmin edilen çıktılar ile gerçek çıktılar arasındaki farkı en aza indirmek için parametrelerini ayarladığı, eğitim adı verilen bir süreç ile elde edilmektedir.

Nöron, bir veya daha fazla kaynaktan girdi alan, girdi üzerinde matematiksel bir işlem gerçekleştiren ve bir çıktı üreten temel bir işlem birimidir ve derin öğrenme algoritmaları, her biri belirli bir görevi yerine getiren çok sayıda yapay nöron katmanından oluşmaktadır. Bir katmanın çıktısı, bir sonraki katmana girdi olarak beslenmekte ve algoritmanın, verilerin giderek karmaşıklaşan özelliklerini öğrenmesine olanak tanımaktadır. Derin öğrenme algoritmaları girdi katamnı, gizli katman ve çıkış katmanı olmak üzere genelde üç katmandan oluşmaktadır. Girdi katmanı, resimler veya metin gibi ham verileri temsil ederken, gizli katman girdi katmanından gelen bilgilerin işlendiği katmandır ve çıktı katmanı, sınıflandırma veya tahmin gibi nihai çıktıyı üretmektedir. Tipik bir derin öğrenme sinir ağında, girdi katmanı, ham girdi verilerini alan ilk nöron katmanıdır. Giriş verileri, resimler, metin veya sayısal veriler gibi birçok biçimde olabilir. Girdi katmanı, girdi verilerindeki her özellik için bir nörondan oluşmaktadır. Örneğin, bu çalışmada giriş verisi 128x128 piksellik bir görüntüdür ve giriş katmanında 16384 nöron bulunmaktadır. Bir yapay sinir ağındaki gizli katmanlar, girdi katmanı ile çıktı katmanı arasında kalan katmanlardır. Gizli katman sayısı ve her gizli katmandaki nöron sayısı, görevin karmaşıklığına ve mevcut veri miktarına bağlı

olarak değişebilmektedir. Gizli katmandaki her nöron bir önceki katmandaki nöronlardan girdi almaktadır ve girdilerin ağırlıklı toplamını hesaplamaktadır. Ağırlıklı toplam daha sonra, modelin doğrusallığını bozan bir aktivasyon fonksiyonundan geçirilmektedir. Gizli katmandaki her bir nöronun çıktısı daha sonra bir sonraki katmana girdi olarak iletilmektedir. Çıkış katmanı, bir sinir ağındaki nöronların son katmanıdır ve modelin son çıktısını üretmektedir. Çıkış katmanındaki nöronların sayısı, görevin doğasına bağlıdır. Örneğin, bir ikili sınıflandırma görevinde, çıktı katmanında iki sınıftan birine ait olma olasılığını veren bir nöron olacak iken bu çalışmadaki gibi çok sınıflı bir sınıflandırma görevinde, her sınıf için o sınıfa ait olma olasılığını veren bir nöron olacaktır.

Derin öğrenme lineer cebir ve hesaplamaya dayanmaktadır. Bir sinir ağındaki herbir nöron, ağırlıkların fonksiyonun katsayıları ve girdilerin değişkenler olduğu doğrusal bir fonksiyon olarak temsil edilebilmektedir. Nöronun çıktısı daha sonra doğrusal olmayan bir aktivasyon fonksiyonundan geçirilmektedir. Aktivasyon fonksiyonlarını detaylandırmak gerekirse;

Sigmoid fonksiyonu, herhangi bir giriş değerini 0 ile 1 arasında bir değere eşleyen yaygın bir aktivasyon fonksiyonudur. Aşağıdaki gibi tanımlanır:

$$f(x) = \frac{1}{(1 + e^{-x})} \tag{2.1}$$

Burada x giriş değeridir.

Sigmoid işlevi, çıktının iki sınıftan birine ait olma olasılığı olduğu ikili sınıflandırma problemlerinde kullanışlıdır.

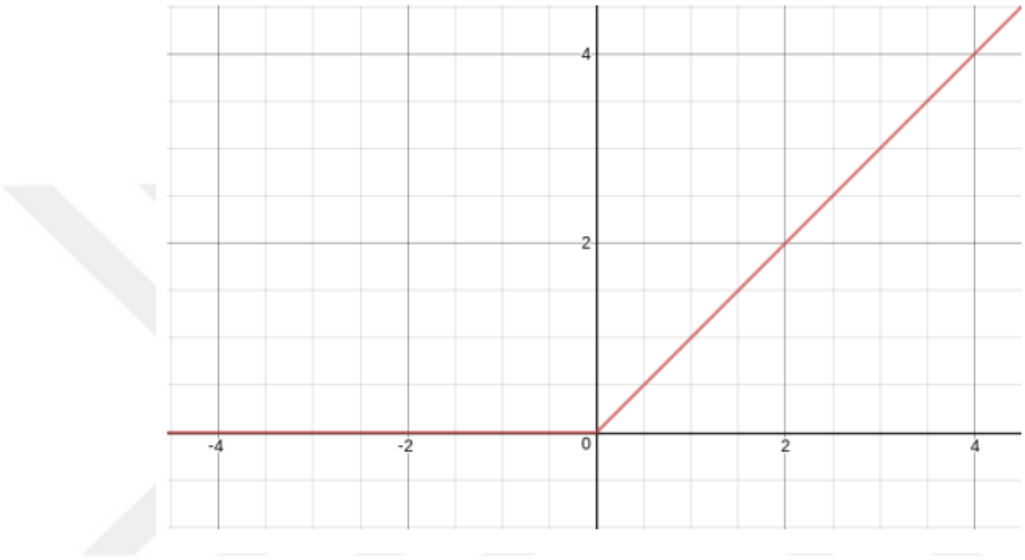
ReLU aktivasyon fonksiyonu, herhangi bir negatif giriş değerini sıfıra ve herhangi bir pozitif giriş değerini kendisine eşleyen doğrusal olmayan bir fonksiyondur. Aşağıdaki gibi tanımlanır:

$$\mathrm{f(x)} = \max(0, \mathrm{x}) \tag{2.2}$$

ReLU işlevi, basit, hesaplama açısından verimli olduğu ve pratikte iyi çalıştığı gösterildiği için derin öğrenmede yaygın olarak kullanılır. ReLU'da giriş sıfırdan küçük veya sıfıra eşitse ReLU işlevinin çıkışının sıfır olmakta ve giriş sıfırdan büyükse giriş

kendisine eşit olmaktadır (Wang ve diğ., 2020). Fonksiyon parçalı lineerdir yani hesaplama açısından verimlidir ve sigmoid veya hiperbolik tanjant gibi diğer aktivasyon fonksiyonlarında meydana gelebilecek yok olan gradyan problemini önlemektedir.

Uygulamada ReLU, basitliği ve etkinliği nedeniyle CNN'lerin gizli katmanlarında genellikle bir aktivasyon işlevi olarak kullanılmaktadır. Şekil 2.1 ReLU'nun çalışma mantığını gösteren grafiği göstermektedir.



Şekil 2.1: ReLU çalışma mantığı (Agarap ve diğ., 2018)

Tanh aktivasyon fonksiyonu, sigmoid fonksiyonuna benzer, ancak herhangi bir giriş değerini -1 ile 1 arasında bir değere eşler. Aşağıdaki gibi tanımlanır:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.3}$$

Softmax aktivasyon fonksiyonu, çıktının her bir sınıfa ait olma olasılığını temsil etmesi gereken çok sınıflı sınıflandırma problemlerinde yaygın olarak kullanılır. Aşağıdaki gibi tanımlanır:

$$f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{2.4}$$

Burada x_i, çıkış katmanındaki i. nöron için giriş değeridir ve toplam, çıkış katmanındaki tüm nöronlar için alınmaktadır.

Softmax fonksiyonu, çıkış katmanındaki nöronların çıkışlarının toplamının 1 olmasını sağlayarak çok sınıflı sınıflandırma problemleri için uygun hale getirmektedir (Kandhro ve diğ., 2020).

Genel olarak, aktivasyon fonksiyonunun seçimi, çözülmekte olan probleme ve sinir ağının yapısına bağlıdır. Uygun bir aktivasyon fonksiyonu seçerek sinir ağı, girdi ve çıktı verileri arasındaki doğrusal olmayan karmaşık ilişkileri öğrenebilmektedir.

Nöronların ağırlıkları, ağırlıklara göre kayıp fonksiyonunun kısmi türevlerinin hesaplanmasını içeren gradyan iniş gibi bir optimizasyon algoritması kullanılarak eğitim sırasında güncellenmektedir.

Optimizasyon, derin öğrenmenin önemli bir bileşenidir çünkü derin öğrenme modelinin amacı, tahmin edilen çıktı ile gerçek çıktı arasındaki farkı en aza indirmektir. Fark, tahmin edilen çıktı ile gerçek çıktı arasındaki hatayı ölçen bir kayıp fonksiyonu ile ölçülmektedir. Optimizasyon algoritması, kayıp fonksiyonunu en aza indirmek için sinir ağının ağırlıklarını ayarlamak için kullanılmaktadır.

Derin öğrenmede en sık kullanılan optimizasyon algoritmaları şunlardır (Tian ve diğ., 2023):

- Gradyan iniş, sinir ağının ağırlıklarına göre kayıp fonksiyonunun gradyanını hesaplayarak çalışan birinci dereceden bir optimizasyon algoritmasıdır. Gradyan, kayıp fonksiyonundaki en dik artışın yönünü temsil etmektedir. Gradyan iniş, sinir ağının ağırlıklarını degradenin ters yönünde yinelemeli olarak güncelleyerek çalışmaktadır. Bu işlem, kayıp fonksiyonu minimize edilene kadar tekrarlanmaktadır.

- Stokastik Gradyan İnişi (SGD), her yinelemede eğitim verilerinin rastgele bir alt kümesini kullanan bir gradyan iniş çeşididir. SGD, eğitim kümesindeki her örnek modele sunulduktan sonra sinir ağının ağırlıklarını güncellemektedir. Güncellemeler daha sık olduğundan, yaklaşım daha hızlı yakınsama sağlayabilmektedir ancak optimizasyon sürecinde daha fazla gürültüye de neden olabilmektedir.

- Adam, gradyan iniş ve stokastik gradyan inişin avantajlarını birleştiren bir optimizasyon algoritmasıdır. Zamanla değişen ve kayıp fonksiyonunun gradyanına uyum sağlayan uyarlanabilir öğrenme oranlarını kullanmaktadır. Adam, genellikle diğer optimizasyon algoritmalarından daha hızlı yakınsadığı ve pratikte iyi çalıştığı gösterildiği için popüler bir seçimdir.
- Adagrad, geçmiş gradyan bilgilerine dayalı olarak her parametrenin öğrenme oranını uyarlayan bir optimizasyon algoritmasıdır. Adagrad, parametrelerin gradyanlarının yüksek varyansa sahip olduğu seyrek veri kümelerinde iyi performans göstermektedir.

Derin öğrenmede optimizasyon tahmin edilen çıktı ile gerçek çıktı arasındaki farkı en aza indirmek için modelin ağırlıklarını ayarlamasında çok önemli bir etkiye sahiptir. Gradyan iniş, SGD, Adam ve Adagrad, derin öğrenmede en sık kullanılan optimizasyon algoritmaları arasındadır. Optimizasyon algoritmasının seçimi, sinir ağının yapısına, veri kümesinin boyutuna ve mevcut hesaplama kaynaklarına bağlıdır.

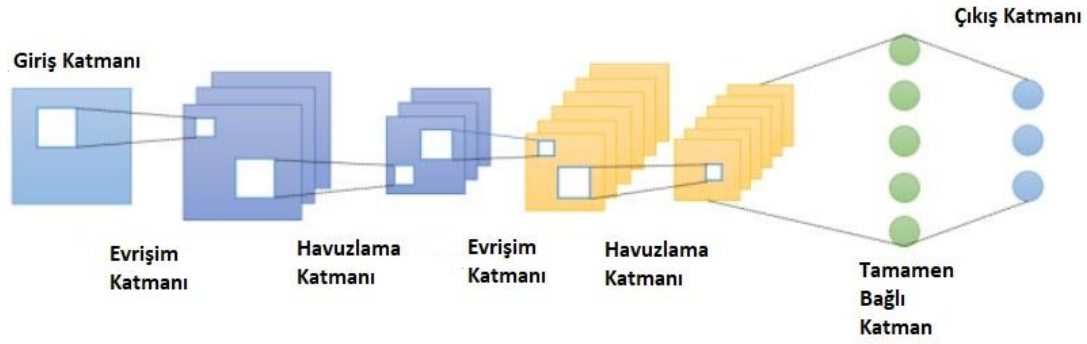
Kayıp işlevi, sinir ağının tahmin edilen çıktısı ile gerçek çıktı arasındaki farkı ölçmektedir. Eğitimin amacı, ağdaki nöronların ağırlıklarını ayarlayarak yapılan kayıp fonksiyonunu en aza indirmektir. Ağırlıkları ayarlama işlemi, geri yayılım olarak bilinir ve ağdaki ağırlıklara göre kayıp fonksiyonunun kısmi türevlerinin hesaplanmasını ve türevlerin ağırlıkları güncellemek için kullanılmasını içermektedir.

2.1.1. Evrişimli sinir ağları (CNN)

Evrişimli Sinir Ağları (CNN), çeşitli görsel tanıma görevlerinde en son teknolojiye sahip sonuçlar elde ederek bilgisayarlı görü alanında devrim yaratan bir sinir ağları sınıfıdır. CNN, girdinin bir görüntü veya bir görüntü dizisi olduğu görüntü sınıflandırma, nesne algılama ve segmentasyon gibi görevler için özellikle yararlıdır (Biratu ve diğ., 2019).

CNN arkasındaki ana fikir, görüntülerde bulunan uzamsal korelasyondan yararlanmaktır. Tipik bir görüntüde, komşu piksellerin benzer değerlere sahip olması muhtemeldir ve aynı nesnenin veya özelliğin parçalarını temsil edebilir. CNN, giriş görüntüsü üzerinde küçük bir filtre veya çekirdek gezdirerek uzamsal korelasyonu

yakalamak için evrişimli katmanlar kullanmaktadır. Şekil 2.2 CNN'lerin genel mimarisini göstermektedir.



Şekil 2.2 : CNN mimarisi (Zhang ve diğ., 2019)

Bir CNN'deki ilk katman, görüntünün ham piksel değerlerini girdi olarak alan giriş katmanıdır. Giriş katmanını takip eden katmanlar, giriş görüntüsü üzerinde bir dizi filtre veya çekirdek evrilerek özellik çıkarımı gerçekleştirmektedir. Her filtre, girdi görüntüsündeki belirli bir özelliği veya modeli vurgulayan bir özellik haritası üretmektedir. CNN'lerde, bir özellik haritası, girdi görüntüsünde belirli bir özelliğin varlığını veya yokluğunu temsil eden iki boyutlu bir dizidir (Brinker ve diğ., 2018). Her bir özellik haritası, girdi görüntüsü üzerinde bir filtre veya çekirdek gezdirerek ve sonuca doğrusal olmayan bir aktivasyon fonksiyonu uygulanarak elde edilmektedir. Özellik haritasının boyutu, giriş görüntüsünün boyutuna, filtrenin boyutuna ve evrişim işleminin adımına bağlıdır. Her özellik haritası, giriş görüntüsündeki kenarlar, köşeler veya lekeler gibi belirli bir özelliği veya deseni vurgulamaktadır. Evrişim katmanındaki filtreler, kayıp fonksiyonunu en aza indirmek için ağırlıklarını ayarlayarak eğitim sürecinde öğrenilmektedir. Böylece filtreler, CNN'nin eldeki görev için en ayırt edici olan bir dizi özelliği öğrenmesini sağlamaktadır. Filtreler için dikkat edilebilir iki adet parametre vardır. Bunlar dolgu (padding) ve adımlar (strides) şeklindedir (Xu ve diğ., 2023). Dolgu, giriş hacminin uzamsal boyutlarını korumak için kullanılan bir tekniktir. Evrişimli işlemlerde, filtre giriş hacmi üzerinde kaymakta ve her konumda evrişim işlemini uygulamaktadır ancak filtre giriş hacminin kenarlarına tam olarak sığmayacağı için filtre giriş hacminin kenarlarına uygulanamamaktadır. Bu sorunu çözmek için giriş hacminin kenarlarına dolgu uygulanmaktadır. Dolgu, giriş hacminin etrafına filtrenin

giriş hacminin kenarlarına uygulanmasını sağlayan sıfırlardan oluşan bir kenarlık eklemeyi içermektedir. Giriş hacmine uygulanan dolgu miktarı, dolgu hiper parametresi ayarlanarak kontrol edilebilmektedir. Adım parametresi ise filtrenin giriş hacmi boyunca hareket ettirildiği adım boyutunu kontrol eden bir hiper parametredir. Adım 1 olarak ayarlandığında, filtre her seferinde bir piksel hareket etmektedir ancak adım 1'den büyük bir değere ayarlandığında, filtre bazı pikselleri atlayarak çıktı hacminin boyutunu azaltmaktadır. Daha büyük adımlar kullanmak, ağın hesaplama karmaşıklığını azaltabilir ve fazla uydurmanın önlenmesine yardımcı olabilmektedir.

Hem dolgu hem de adım, evrişimli bir katmanın çıktı hacminin boyutunu etkilemektedir. Çıktı hacminin boyutunu hesaplama formülü aşağıdaki gibidir:

$$O = \frac{W - F + 2P}{S + 1} \tag{2.5}$$

O çıkış hacminin boyutu, W giriş hacminin boyutu, F filtrenin boyutu, P dolgu miktarı ve S adımdır.

Ayrıca, birden çok özellik haritası oluşturmak için girdi görüntüsüne birden çok filtre uygulanabilmektedir. Her filtre, belirli bir özelliği yakalamak için tasarlanmıştır ve ortaya çıkan özellik haritaları, girdi görüntüsünün daha sağlam bir temsilini elde etmek için birleştirilmektedir (Ghanbari ve diğ., 2021). CNN'lerde filtreler, belirli özellikleri çıkarmak için giriş görüntüsü üzerinde kayan küçük matrislerdir. Her filtre, giriş görüntüsündeki kenarlar, köşeler veya lekeler gibi belirli bir özelliği veya deseni yakalamak için tasarlanmıştır. Filtrenin ağırlıkları, kayıp işlevini en aza indirmek için eğitim sürecinde öğrenilmektedir ve CNN'nin eldeki görev için en ayırt edici olan bir dizi filtreyi öğrenmesine olanak tanımaktadır. Filtreler tipik olarak 3x3 veya 5x5 gibi tek boyutlu kare matrislerdir ancak daha karmaşık görevler için daha büyük filtreler de kullanılabilmektedir. Evrişim işlemi sırasında filtre, giriş görüntüsü üzerinde kaymakta ve filtre değerleri ile giriş görüntüsündeki karşılık gelen piksel değerleri arasındaki iç çarpımı hesaplamaktadır. Sonuç özellik haritasında, filtre tarafından o konumda yakalanan özelliğin varlığını veya yokluğunu temsil eden tek bir değerdir. CNN'ler, farklı ağırlıklara sahip birden çok filtre kullanarak girdi görüntüsünden birden çok özelliği çıkarabilmekte ve birden çok özellik haritası oluşturabilmektedir. Her filtre,

farklı yönlerdeki kenarlar gibi belirli bir özelliği yakalamak için tasarlanmıştır ve ortaya çıkan özellik haritaları, giriş görüntüsünün daha sağlam bir temsilini elde etmek için birleştirilmektedir. Çoklu özellik haritaları daha sonra, uzamsal çözünürlüğü azaltmak ve en göze çarpan özellikleri yakalamak için bir havuzlama katmanından geçirilmektedir. Özellik haritalarının kullanımı, CNN'lerin önemli bir yönüdür ve girdi verilerinin hiyerarşik temsillerini öğrenmelerini sağlamaktadır. Bir CNN'nin ilk katmanları kenarlar ve köşeler gibi düşük seviyeli özellikleri öğrenirken, daha derindeki katmanlar şekiller ve nesneler gibi daha karmaşık özellikleri öğrenmektedir. Hiyerarşik temsil, CNN'lerin görüntülerdeki karmaşık kalıpları yakalamasına ve çeşitli bilgisayarla görme görevlerinde son teknoloji ürünü sonuçlar elde etmesine olanak tanımaktadır.

Çıkarılan özellik haritaları daha sonra modelde doğrusal olmamayı sağlamak için ReLU veya Sigmoid gibi doğrusal olmayan bir aktivasyon fonksiyonundan geçirilmektedir.

Ayrıca CNN'ler özellik haritalarının uzamsal çözünürlüğünü azaltmak ve en göze çarpan özellikleri yakalamak için havuzlama katmanlarını kullanmaktadırlar. Havuzlama katmanları, evrişimli katmanlar tarafından üretilen özellik haritalarının uzamsal boyutlarını azaltmak için CNN'lerde kullanılmaktadır (Kumari ve diğ., 2022). Özellik haritasını, havuzlama bölgeleri veya alıcı alanlar olarak adlandırılan, birbiriyle örtüşmeyen küçük bölgelere ayırarak ve her bölge için tek bir çıktı değeri hesaplayarak çalışmaktadırlar. Bu şekilde en göze çarpan özellikleri korurken özellik haritasının boyutunu azaltmaktadır. En yaygın havuzlama türü, her havuzlama bölgesindeki maksimum değeri veren maksimum havuzlamadır. Maksimum havuzlama işlemi şu şekilde tanımlanır:

$$y_{i,j,k} = \max 1_{p=1}^{S} \max 1_{q=1}^{S} x_{(i-1)S+p,(j-1)S+q,k} \quad (2.6)$$

burada $y_{i,j,k}$, (i,j), konumundaki k'ıncı özellik haritasının çıktı değeridir, $x_{(i-1)S+p,(j-1)S+q,k}$ ise k'ıncı özellik haritasındaki $(i-1)S+p, (j-1)$ konumundaki giriş değeridir ve S, havuzlama bölgesinin boyutudur. Diğer bir havuzlama türü, her bir havuzlama bölgesindeki ortalama değeri hesaplayan ortalama havuzlamadır. Ortalama havuzlama işlemi şu şekilde tanımlanır:

$$y_{i,j,k} = \frac{1}{S^2} \sum_{p=1}^{S} \sum_{q=1}^{S} x_{(i-1)S+p,(j-1)S+q,k} \tag{2.7}$$

Burada $y_{i,j,k}$ ve $x_{(i-1)S+p,(j-1)S+q,k}$ maksimum havuzlama işlemi gibi tanımlanmaktadır.

Havuzlama katmanları, bitişik havuzlama bölgelerinin belirli bir miktarda örtüştüğü örtüşen havuzlama bölgeleriyle veya havuzlama bölgeleri arasındaki örtüşmeyi azaltan havuzlama bölgesi boyutundan daha büyük adımlarla da kullanılabilmektedir. Bu, CNN'lerin artan hesaplama karmaşıklığı pahasına özellik haritalarında daha ayrıntılı ayrıntıları yakalamasına olanak tanımaktadır.

Birkaç evrişimli ve havuzlama katmanından sonra çıktı evrişimli katmanlar tarafından çıkarılan özelliklere dayalı olarak sınıflandırma veya regresyon gerçekleştiren tam bağlantılı bir katmana beslenmektedir. Bir CNN'nin çıktı katmanı, bir sınıflandırma görevindeki sınıflar üzerinde bir olasılık dağılımı üretmek için tipik olarak bir softmax aktivasyon fonksiyonu kullanılmaktadır.

CNN modeli oluştururken yukarıdaki genel bilgiler dışında belli başlı parametrelerin de ayarlanması gerekmektedir. Bunlardan biri Bırakma (Dropout)dır. Bırakma, CNN'lerde fazla uydurmayı önlemek için kullanılabilen bir düzenlileştirme tekniğidir. Bırakmanın arkasındaki temel fikir, eğitim sırasında ağdaki nöronların bir kısmını rastgele bırakmaktır. Bunu yaparak ağ, belirli nöronların aktivasyonuna bağlı olmayan daha sağlam özellikleri öğrenmeye zorlanmaktadır. Eğitim sırasında, ağdaki her nörona, standart olarak 0,5 civarında bırakılma olasılığı verilir fakat bu değer eldeki probleme göre değişebilmektedir. Bırakılan nöronlar, ağın ileri veya geri geçişine dahil edilmemektedir böylece ağın kapasitesi etkili bir şekilde azaltılır ve nöronların birlikte adaptasyonunu önlemektedir. Test zamanında ise tüm ağ kesinti olmadan kullanılmaktadır. Bununla birlikte, aktivasyonların genel ölçeğinin aynı kalmasını sağlamak için her bir nöronun çıktısı, eğitim sırasında mevcut olma olasılığı ile çarpılmaktadır. Bırakma, aşırı uydurmayı önleyerek, CNN'lerin genelleştirme performansını iyileştirmeye ve onları gerçek dünya uygulamaları için daha uygun hale

getirmeye yardımcı olabilmektedir fakat fazla bir bırakma oranı belirlemek aşırı bilgi kaybına sebep olacağı için modelin öğrenememesine de sebep olabilmektedir. Ayarlanması gereken başka bir parametre ise düzenlileştirme (regularization) olabilir. Düzenlileştirme işlemi de probleme göre değişmektedir. Düzenlileştirme, CNN'lerde fazla uydurmayı önlemek için kullanılabilecek bir dizi tekniktir (Chen ve diğ., 2016). Düzenli hale getirmenin arkasındaki temel fikir, ağın kayıp işlevine bir ceza terimi eklemektir; bu şekilde ağ yeni verilere daha iyi genelleştiren daha basit ve sorunsuz işlevleri öğrenmeye teşvik edilmektedir. CNN'lerde yaygın olarak kullanılan bir düzenleme tekniği, ağdaki ağırlıkların karelerinin toplamıyla orantılı olarak kayıp fonksiyonuna bir ceza terimi ekleyen L2 düzenlemesidir. L2 düzenlemesi, ağı daha küçük ağırlık değerleri öğrenmeye teşvik eder ve böylece ağ kapasitesini azaltarak fazla uyumu önlemeye yardımcı olabilmektedir (Shi ve diğ., 2022).

$$L_2 = \lambda \sum_{w} w^2 \tag{2.8}$$

burada λ, düzenlileştirme gücüdür ve w, ağdaki bir ağırlık parametresidir.

CNN'lerde kullanılabilecek bir başka düzenleme tekniği, ağdaki ağırlıkların mutlak değerlerinin toplamına orantılı bir ceza terimi ekleyen L1 düzenlemesidir. Bu düzenleme, ağı girdi verilerinin boyutsallığını azaltmaya ve aşırı uydurmayı önlemeye yardımcı olabilecek seyrek özellikleri öğrenmeye teşvik etmektedir (Ghosh ve diğ., 2009).

$$L_1 = \lambda \sum_{w} |w| \tag{2.9}$$

burada λ, düzenlileştirme gücüdür ve w, ağdaki bir ağırlık parametresidir.

Her iki durumda da, düzenlileştirme gücü λ, eğitim kaybını en aza indirme ve ağdaki ağırlıkların boyutunu en aza indirme arasındaki değiş tokuşu kontrol etmektedir. λ değerini ayarlayarak, düzenlileştirme derecesi kontrol edilebilmektedir.

Düzenlileştirme, CNN'lerin eğitiminde önemli bir tekniktir çünkü aşırı uydurmayı önlemeye ve ağın genelleştirme performansını iyileştirmeye yardımcı olmaktadır.

CNN'in mekansal değişmezlik, parametre paylaşımı, hiyerarşik özellik öğrenme, yerelleştirme ve veri verimliliği gibi birçok avantajı bulunmaktadır (Yamashita ve diğ., 2018; Pattabiraman ve diğ., 2022).

• Mekansal Değişmezlik : CNN'ler, konumları, yönleri veya boyutları ne olursa olsun bir görüntüdeki nesneleri tanıyabilmektedir. Giriş görüntüsünün tüm bölümlerine aynı filtreleri uygulayan ve böylece çeviri için değişmez olan bir dizi özellik haritası oluşturan evrişimli katmanların kullanılmasıyla elde edilmektedir.

• Parametre Paylaşımı : CNN'ler, belirli bir özellik haritasındaki tüm konumlar için aynı ağırlık kümesini kullanmaktadır. Böylece öğrenilmesi gereken parametre sayısını azaltarak CNN'leri bellek kullanımı ve eğitim süresi açısından daha verimli hale getirmektedir.

• Hiyerarşik Özellik Öğrenme : CNN'ler, birden çok katmanın kullanımıyla giderek daha karmaşık ve soyut özellikleri öğrenebilmektedir. Ağın alt katmanları kenarlar ve köşeler gibi basit özellikleri öğrenirken, üst katmanlar şekiller ve dokular gibi daha karmaşık özellikleri öğrenmektedir.

• Yerelleştirme : CNN'ler yalnızca görüntüleri sınıflandırmakla kalmaz aynı zamanda içlerindeki nesnelerin konumunu da belirleyebilmektedir. Bir video akışındaki nesneleri izleme gibi görevlerde yardımcı olabilecek nesne algılama ve yerelleştirme gibi tekniklerin kullanılmasıyla elde edilmektedir.

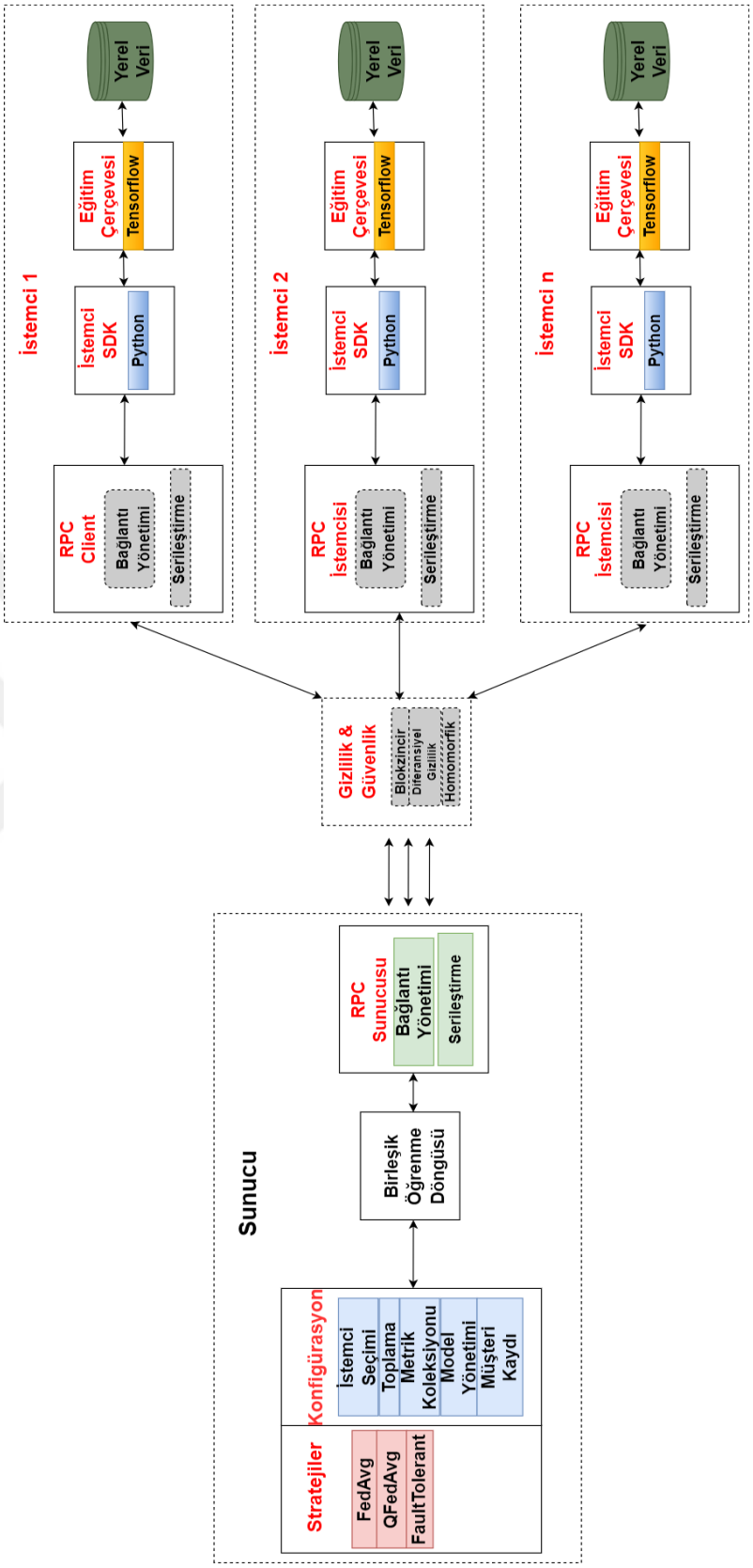
• Veri Verimliliği : CNN'ler, özellikleri hiyerarşik olarak öğrenme yetenekleri sayesinde nispeten küçük miktarlarda eğitim verileriyle iyi performans gösterebilmektedir. Özellikle büyük miktarda etiketlenmiş veri elde etmenin zor veya pahalı olabileceği tıbbi görüntüleme gibi görevlerde önemlidir.

Sonuç olarak, CNN, çeşitli görsel tanıma görevlerinde son teknoloji sonuçlar elde ederek bilgisayar görüşü alanında devrim yaratan güçlü bir sinir ağları sınıfıdır. CNN'ler, özellikleri çıkarmak için evrişimli katmanları ve özellik haritalarının uzamsal çözünürlüğünü azaltmak için katmanları bir araya getirerek görüntülerde bulunan uzamsal korelasyondan yararlanmaktadır. CNN'ler tarafından öğrenilen hiyerarşik

temsil, görüntülerdeki karmaşık kalıpları yakalamalarına ve çeşitli bilgisayarla görme görevlerinde son teknoloji sonuçlara ulaşmalarına olanak tanımaktadır.

BÖLÜM 3. FEDERE ÖĞRENME

Federe öğrenme, birden fazla cihazın verilerini paylaşmadan ortak bir modeli ortaklaşa eğitmesine izin veren bir makine öğrenmesi tekniğidir. Yaklaşım, dağıtılmış bir ortamda makine öğrenmesi modellerini eğitmenin gizliliği koruyan ve daha verimli bir yolunu sağlamaktadır. Federe öğrenme süreci, katılan tüm cihazlara gönderilen merkezi bir model oluşturarak başlamaktadır. Her cihaz daha sonra modeli yerel verileri üzerinde eğitir ve güncellenen model parametrelerini merkezi sunucuya geri gönderir. Sunucu, tüm cihazlardan gelen güncellemeleri toplar ve bunları merkezi modeli güncellemek için kullanır. Bu süreç, model istenilen doğruluk düzeyine yakınsayana kadar yinelemeli olarak tekrarlanır. Federe öğrenme yaklaşımı, verilerin hassas olduğu ve paylaşılamadığı durumlarda veya verilerin merkezi bir konuma aktarılamayacak kadar büyük olduğu durumlarda kullanışlıdır. Şekil 3.1'de görüldüğü üzere federe öğrenme bir çok bileşenden oluşmaktadır.



Şekil 3.1 : Federe öğrenme bileşenleri

3.1. Şifreleme Yöntemleri

Federe öğrenmede, güvenli iletişim ve veri gizliliği, verilerin gizliliğini korumak için çok önemlidir. Verilerin gizliliğini sağlamak için farklı şifreleme yöntemleri kullanılmaktadır.

3.1.1. Homomorfik şifreleme

Homomorfik şifreleme, hesaplamaların şifre çözme olmadan şifreli metin üzerinde yapılmasına izin veren bir şifreleme türüdür (Naehrig ve diğ., 2011). Başka bir deyişle, şifrelenmiş veriler, verilerin şifresinin çözülmesine gerek kalmadan doğrudan hesaplamalar için kullanılabilmektedir. Homomorfik şifreleme, hesaplamalar için kullanılırken verilerin gizliliğini korumak için de kullanılabilir. Homomorfik şifreleme için matematiksel formül şöyledir:

$$C = E(m) \tag{3.1}$$

burada C şifreli metindir, E şifreleme işlevidir ve m düz metindir.

Veriler genellikle farklı cihazlara veya kuruluşlara dağıtıldığından, federe öğrenmede gizliliği ve güvenliği korumak için önemli bir özelliktir.

Homomorfik şifreleme iki ana kategoriye ayrılabilir: tamamen homomorfik şifreleme (FHE) ve kısmen homomorfik şifreleme (PHE) (Scheibner ve diğ., 2021). Tamamen homomorfik şifreleme, şifrelenmiş veriler üzerinde herhangi bir hesaplama türünün gerçekleştirilmesine olanak sağlarken kısmen homomorfik şifreleme, yalnızca belirli hesaplama türlerinin gerçekleştirilmesine izin vermektedir. Tamamen homomorfik şifreleme için temel matematiksel formül şudur:

$$E(m_1) \cdot E(m_2) = E(m_1 + m_2) \tag{3.2}$$

E(m), m mesajının şifrelendiğini ifade eder.

Homomorfik şifrelemenin matematiksel temeli, kafes adı verilen matematiksel yapılara dayanmaktadır. Kafesler, karmaşık veri yapılarını matematiksel olarak kolayca manipüle edilebilecek bir şekilde temsil etmek için kullanılabilen bir tür ayrık matematiksel yapıdır (Hu ve diğ., 2023). 3.2'de verilen matematiksel formül, iki düz metin mesajının şifreli metinleri üzerinde işlem yaparak, düz metin mesajlarının toplamının şifreli metnini elde edebileceğimizi göstermektedir. Bu özellik homomorfik toplama olarak bilinir. Benzer şekilde, kısmen homomorfik şifreleme için iki tür vardır ve bunlar toplama için homomorfik şifreleme ve çarpma için homomorfik şifrelemedir. Toplama için kısmen homomorfik şifrelemenin temel matematiksel formülü 3.3'te verilmiştir.

$$E(m_1) + E(m_2) = E(m_1 + m_2) \tag{3.3}$$

Çarpma için kısmen homomorfik şifreleme için matematiksel formül 3.4'te verilmiştir.

$$E(m_1) \cdot E(m_2) = E(m_1 \cdot m_2) \tag{3.4}$$

Tamamen Homomorfik Şifreleme hem homomorfik toplamayı hem de çarpmayı desteklemekte iken Kısmen Homomorfik Şifreleme yalnızca homomorfik toplamayı veya çarpmayı desteklemektedir.

Homomorfik şifrelemeyi uygulamak için kullanılan birkaç farklı teknik vardır:

• Tam Sayıları Çarpanlara Ayırmaya Dayalı Yöntemler: Bu yöntemler, büyük tam sayıları asal çarpanlarına ayırmanın zorluğuna dayanmaktadır (Mittal ve diğ., 2022). Bu yöntemin örnekleri arasında Rivest-Shamir-Adleman (RSA) şifreleme sistemi ve Rabin şifreleme sistemi bulunmaktadır.

• Eliptik Eğri Şifrelemeye Dayalı Yöntemler: Bu yöntemler, verileri şifrelemek ve şifresini çözmek için eliptik eğrilerin özelliklerini kullanmaktadır. Bu yöntemin örnekleri arasında ElGamal kripto sistemi ve Eliptik Eğri Entegre Şifreleme Düzeni (ECIES) kripto sistemi yer almaktadır (Hong ve diğ., 2016).

• Kafes Tabanlı Yöntemler: Bu yöntemler, kafeslerin özelliklerini temel almaktadır ve Gentry-Halevi şifreleme sistemini ve Nth Degree Truncated Polynomial Ring Units (NTRU) şifreleme sistemini içermektedir (Kadykov ve diğ., 2021).

Federe öğrenmede homomorfik şifrelemenin kullanılması, verilerin gizliliğini ve güvenliğini koruyarak verilerin şifre çözmeye gerek kalmadan işlenmesine ve analiz edilmesine olanak tanımaktadır. Bununla birlikte, homomorfik şifrelemenin hesaplama karmaşıklığı, onu büyük ölçekli hesaplamalar için yavaş ve verimsiz hale getirebilmektedir. Ek olarak, homomorfik şifrelemenin uygulanması ve kullanılması, önemli uzmanlık ve kaynaklar gerektirir, bu da daha küçük kuruluşların veya bireylerin uygulamasını zorlaştırmaktadır.

3.1.2. Diferansiyel gizlilik

Diferansiyel gizlilik, hassas bilgileri korumak için kullanılan ve bilgilerden yararlı bilgiler elde edilmesine izin veren bir tekniktir. Veri noktalarının tanımlanmasını önlemek için verilere parazit eklenmesini içermektedir. Verilere eklenen gürültü miktarı, gizlilik bütçesi adı verilen bir parametre tarafından kontrol edilmektedir. Gizlilik bütçesi ne kadar yüksek olursa, verilere o kadar fazla gürültü eklenmektedir.

Resmi olarak, diferansiyel gizlilik şu şekilde tanımlanabilir (Adnan ve diğ., 2022):

D bir veri kümesi olsun ve M, D'yi girdi olarak alan ve bir aralıkta bir çıktı üreten rastgele bir algoritma olsun. En fazla bir kayıtta farklılık gösteren herhangi iki veri kümesi D_1 ve D_2 için ve S çıktılarının herhangi bir alt kümesi için M'nin (ε, δ)-diferansiyel gizliliği karşıladığı söylenmektedir.

$$P[M(D_1) \in S] \leq e^{\epsilon} P[M(D_2) \in S] + \delta \tag{3.5}$$

Burada ε ve δ sağlanan mahremiyet miktarını kontrol eden parametrelerdir. ε parametresi, verilere eklenen gürültü miktarının bir ölçüsüdür.

Denklem 3.5'teki M algoritmasından elde edilen belirli bir çıktının, girdi olarak D1 veya D2'nin kullanılması durumunda yaklaşık olarak aynı olasılıkta olduğunu ifade eder. ε parametresi sağlanan mahremiyet miktarını kontrol etmektedir. Daha büyük ε

değerleri daha az mahremiyet sağlar. δ parametresi, algoritmanın diferansiyel gizliliği karşılayacağına dair daha fazla güvence sağlayan daha küçük δ değerleri ile başarısızlık olasılığını kontrol etmektedir. Diferansiyel gizliliğin önemli bir yönü, mekanizma tarafından sunulan mahremiyet koruma seviyesini belirleyen gizlilik parametresi olan ε'nun seçimidir. Daha küçük bir ε değeri, daha yüksek düzeyde bir gizlilik korumasına karşılık gelmektedir ancak veri kullanımının azalmasına sebep olmaktadır. Öte yandan, daha büyük bir ε değeri, daha yüksek veri kullanımı sağlamaktadır ancak mahremiyet korumasının azalmasına sebep olmaktadır. Bu nedenle, ε seçimi, mahremiyet ve veri faydası arasında bir değiş tokuştur ve belirli uygulamaya ve kabul edilebilir risk düzeyine bağlıdır.

Diferansiyel gizlilik, uygulamada veriler analiz edilmeden önce verilere rastgele gürültü eklenerek elde edilebilmektedir. Bunu yapmanın bir yolu, verilere Laplace veya Gaussian gürültüsünü eklemektir (Wu ve diğ., 2022); bu gürültüler, sıfır etrafında ortalanmış ve aşağıdaki gibi bir olasılık yoğunluk fonksiyonuna sahip bir gürültü türüdür:

$$f(x; \mu, b) = \frac{1}{2b} \times e^{-\frac{|x-\mu|}{b}} \tag{3.6}$$

burada μ dağılımın merkezidir ve b dağılımın yayılmasını kontrol eder.

Genel olarak, diferansiyel mahremiyet, bireylerin mahremiyetini korumak için önemli bir tekniktir ve yine de verilerden faydalı içgörüler elde edilmesine izin vermektedir. Bir algoritma tarafından sağlanan gizlilik miktarını ölçmek için matematiksel bir çerçeve sağlar ve ek gizlilik koruması katmanları sağlamak için homomorfik şifreleme gibi diğer gizliliği koruyan tekniklerle birlikte kullanılabilmektedir.

Diferansiyel gizlilik teknikleri Rastgele Yanıt ve Sorgu kısıtlama olarak iki ana kategoriye ayrılabilmektedir.

Rastgele yanıt, yayınlanmadan önce verilere rastgele gürültü ekleyerek hassas bilgileri korumak için kullanılan bir tekniktir. Rastgele yanıtta, her bireyin yanıtı, sorulan sorunun duyarlılığına dayalı bir olasılık dağılımına göre rasgele belirlenmektedir (Lai

ve diğ., 2022). Rastgele yanıt tekniklerinin iki ana türü, toplamsal ve çarpımsal gürültü modelleridir.

Toplam gürültü modelinde, stokastik bir değer oluşturmak için gerçek değere rastgele bir gürültü değeri eklenmektedir. Çarpımsal gürültü modelinde ise rastgele bir gürültü değeri gerçek değerle çarpılarak stokastik bir değer oluşturulmaktadır. Stokastik değer daha sonra açıklanmaktadır. Gürültü değerlerinin olasılık dağılımı, tipik olarak, bireylerin mahremiyetini korurken verilerin istatistiksel olarak doğru kalmasını sağlamak için seçilmektedir. Sorgu kısıtlama, bir veritabanına yapılabilecek sorgu sayısını sınırlamak için kullanılan bir tekniktir. Sorgu sayısını sınırlayarak, bir saldırganın bireyler hakkında hassas bilgileri keşfetmesini engellemek mümkündür (Shen ve diğ., 2020). İki ana sorgu kısıtlama tekniği türü k-anonimlik ve l-çeşitliliktir.

k-anonimlik, her bireyin en az k-1 diğer bireylerden ayırt edilemez olmasını sağlayarak bir veri kümesindeki bireylerin mahremiyetini korumak için kullanılan bir tekniktir. Bu, bireyleri niteliklerine göre kümelere ayırarak ve ardından niteliklerini tüm küme için geçerli olan bir genellemeyle değiştirerek elde edilmektedir. Amaç, bir saldırganın veri kümesindeki herhangi bir kişiyi yüksek bir güvenle tanımlayamamasına neden olmaktadır.

l-çeşitlilik, bir veri kümesindeki her grubun belirli bir özellik için en az l farklı değer içermesini sağlamak için kullanılan bir tekniktir. Verilerin öznitelik için değerlerin dağılımı gibi her gruba daha fazla bilgi eklenerek elde edilmektedir. Amaç, bir saldırganın belirli bir gruptaki üyeliklerine dayanarak bireyler hakkında hassas bilgiler çıkaramamasına neden olmaktadır.

Her iki teknik de istatistiksel analize izin verirken bir veri kümesindeki bireylerin mahremiyetini korumak için kullanılabilmektedir.

3.2. Haberleşme Mimarileri

Federe öğrenmede belirli kullanım durumu ve gereksinimlere bağlı olarak üç ana haberleşme mimarileri bulunmaktadır.

3.2.1. Merkezi haberleşme mimarisi

Bu tür bir mimaride birden çok istemciden toplanan verileri kullanarak bir makine öğrenmesi modelini eğitmekten merkezi bir sunucu sorumludur. Sunucu, istemcilerden verileri alır, toplar ve model parametrelerini güncellemek için kullanır. Merkezi yaklaşım, geleneksel makine öğrenmesine benzer ancak istemcilerin verilerini gizli tutarken öğrenme sürecine katılmalarına izin verme avantajına sahiptir.

Merkezileştirilmiş haberleşme mimarisi, sunucunun öğrenme sürecini yönettiği ve istemcilerin yalnızca sunucuya yerel güncellemeler sağladığı bir Federe öğrenme mimarisi çeşididir (Nikham ve diğ., 2019). Başka bir deyişle, sunucu, istemci güncellemelerini toplayan ve genel modeli güncelleyen merkezi koordinatör görevi görmektedir. Ortak model geliştirme süreci aşağıdaki adımlara ayrılabilir.

1. Sunucu başlatma: Sunucu, öğrenme süreci için başlangıç noktası olarak kullanılacak genel modeli başlatmaktadır.

2. İstemci seçimi: Sunucu, bilgi işlem yetenekleri, ağ bağlantısı veya veri dağıtımı gibi belirli kriterlere dayalı olarak bir istemci alt kümesi seçmektedir.

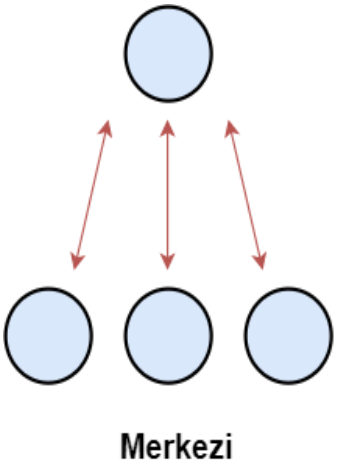
3. Model dağıtımı: Sunucu, genel modelin geçerli sürümünü seçilen istemcilere dağıtmaktadır.

4. Yerel model güncellemesi: Her istemci, modeli yerel verilerini kullanarak eğitmektedir ve model parametrelerini yerel kayıp işlevine göre güncellemektedir.

5. Model toplama: Sunucu, istemcilerden güncellenen modelleri toplar ve bunları basit ortalama alma gibi önceden tanımlanmış bir toplama algoritması kullanarak yeni bir genel modelde toplamaktadır.

6. Model dağıtımı: Sunucu, yeni genel modeli istemcilere dağıtmaktadır ve işlem 4. adımdan itibaren tekrarlanmaktadır.

Merkezi haberleşme mimarisinde merkezi bir koordinatörün kullanılması, merkezi olmayan haberleşme mimarisine kıyasla gelişmiş model yakınsama, öğrenme süreci üzerinde daha iyi kontrol ve karmaşık algoritmaların daha kolay uygulanması gibi çeşitli avantajlara sahiptir. Şekil 3.2'de görüldüğü üzere merkezi haberleşme mimarisinde bir sunucu istemciler ile haberleşmektedir.



Şekil 3.2 : Merkezi haberleşme

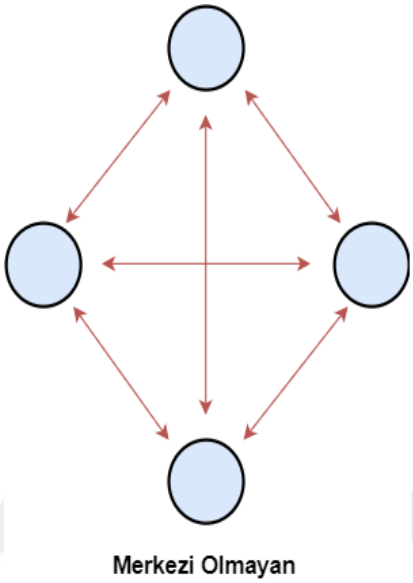
Gizliliği ve güvenliği sağlamak için, güvenli çok taraflı hesaplama ve homomorfik şifreleme gibi çeşitli şifreleme teknikleri kullanılabilmektedir. Ek olarak, bireysel istemcilerin verilerinin gizliliğini daha fazla korumak için diferansiyel gizlilik uygulanabilmektedir ki bu çalışmanın bir parçasında diferansiyel gizlilik tercih edilmiştir.

Kullanıcıların verilerinin birbirine çok benzediği yani veri kümesinde aynı özelliklere sahip değişkenlerin bulunduğu, verilerin hassas veya özel olmadığı durumlarda merkezi haberleşme mimarisi tercih edilebilmektedir. Bunun nedeni, merkezi haberleşmede, verilerin birleştirme için merkezi bir sunucuya gönderilmesi ve merkezi sunucunun, modelin doğruluğunu arttırmak için daha gelişmiş algoritmalar ve optimizasyonlar gerçekleştirebilmesidir. Ek olarak, merkezi Federe haberleşme mimarisi, veri gizliliğinin ve güvenliğinin son derece önemli olduğu sağlık veya finans gibi verilerin tek bir sahibinin veya denetleyicisinin olduğu uygulamalar için uygundur.

3.2.2. Merkezi olmayan haberleşme mimarisi

Bu tür bir haberleşme mimarisinde merkezi bir sunucu yoktur. Bunun yerine istemciler, model parametrelerini ve gradyanları birbirleriyle değiştirerek modeli işbirliği içinde eğitmektedirler (Chellapandi ve diğ., 2023). Merkezi olmayan yaklaşım eşler arası öğrenme olarak da bilinmektedir. Merkezi olmayan haberleşme mimarisi, merkezi bir otoritenin olmadığı veya ölçeklenebilirlik ve hata toleransına ihtiyaç duyulan

senaryolarda kullanışlıdır. Şekil 3.3'de görüldüğü üzere merkezi haberleşme mimarisinde istemciler kendi aralarında haberleşmektedir.



Şekil 3.3 : Merkezi olmayan haberleşme

Merkezi olmayan haberleşme mimarisi merkezi bir sunucuya ihtiyaç duymadan eğitim sürecini birden fazla düğüm veya cihaza dağıtan bir tür Federe öğrenmedir. Bu mimaride her düğüm kendi verileri üzerinde yerel eğitim gerçekleştirir ve yalnızca gerekli güncellemeleri diğer düğümlerle paylaşmaktadır. Merkezi olmayan yaklaşım, düğümler arasında aktarılan veri miktarını azaltmaktadır böylece gizliliği arttırmakta ve iletişim maliyetlerini düşürmektedir.

Merkezi olmayan haberleşme, düğümlerin birbirleriyle doğrudan iletişim kurduğu eşler arası bir ağda çalışmaktadır. Her düğümün kendi veri kümesi vardır ve kendi modelini yerel olarak eğitmektedir. Modeller, her bir düğümün güncellemelerini diğer düğümlerle paylaştığı ve yeni bir model oluşturmak için bunları bir araya getirdiği bir mutabakat süreciyle güncellenmektedir. Mutabakat süreci, güncellemelerin ortalamasının alınması veya daha karmaşık bir algoritmanın kullanılması gibi farklı biçimler alabilmektedir.

Merkezi olmayan haberleşme mimarisinde de verilerin gizliliği korunmaktadır çünkü veriler hiçbir zaman cihazlardan dışarı çıkmamaktadır. Bunun yerine, yalnızca model güncellemeleri düğümler arasında paylaşılmaktadır. Bu, hassas bilgilerin cihazda

kalarak veri ihlali riskini azalttığı anlamına gelmektedir. Buna ek olarak, eğitim süreci çok sayıda cihaza dağıtılabildiğinden, eğitim için çok büyük veri kümelerinin kullanılmasına olanak sağladığından, merkezi olmayan haberleşme mimarisi yüksek düzeyde ölçeklenebilmektedir.

Merkezi olmayan haberleşme mimarisinin olası bir dezavantajı, fikir birliği sürecine duyulan ihtiyaç nedeniyle daha uzun bir eğitim süresi gerektirebilmesidir. Ancak bu sorun daha hızlı algoritmaları kullanılarak veya merkezi olmayan haberleşme mimarisi, merkezi veya yarı merkezi yöntemlerle birleştiren hibrit bir yaklaşım kullanılarak azaltılabilmektedir.

Merkezi olmayan haberleşme mimarisi, tıbbi veya finansal uygulamalar gibi veri gizliliğinin son derece önemli olduğu senaryolarda özellikle yararlıdır. Verilerin çok sayıda sensör veya cihaz tarafından üretildiği Nesnelerin İnterneti (IoT) uygulamaları gibi verilerin yüksek oranda dağıldığı durumlarda da kullanılabilmektedir.

3.2.3. Hibrit haberleşme mimarisi

Hibrit haberleşme mimarisi, merkezi ve merkezi olmayan Federe haberleşme mimarisinin bir birleşimidir. Hibrit yaklaşımda, istemciler gruplara ayrılmaktadır ve her grubun yerel model güncellemelerini toplamaktan sorumlu merkezi bir sunucusu vardır. Merkezi sunucular daha sonra modellerini bir araya getirmek ve genel modeli güncellemek için birbirleriyle iletişim kurmaktadır.

Hibrit haberleşme mimarisi, istemcilerde veya ağda kaynak kısıtlamalarının olduğu ancak yine de gizliliğin korunmasına ihtiyaç duyulan durumlarda kullanışlıdır. Hibrit yaklaşım, hem merkezi hem de merkezi olmayan yaklaşımların güçlü yönlerinden yararlanarak kaynakların daha verimli kullanılmasına olanak tanımaktadır. Şekil 3.4'de hibrit haberleşme mantığı gösterilmiştir ve görüldüğü üzere her istemci grubundan sorumlu birer sunucu vardır.

Hibrit haberleşme süreci aşağıdaki adımları içermektedir:

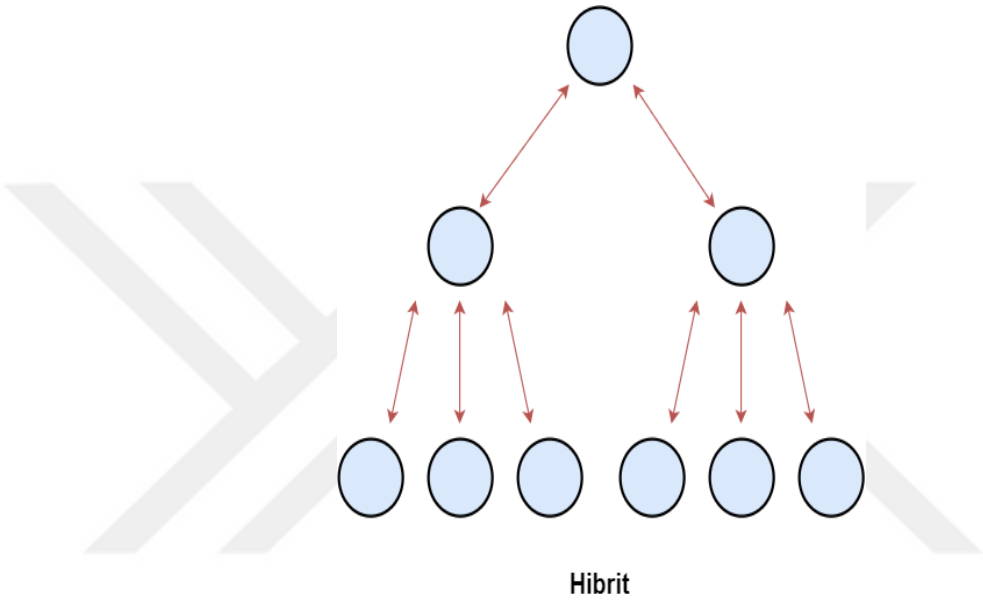
1. İstemciler, coğrafi konum, donanım özellikleri vb. gibi çeşitli faktörlere dayalı olarak daha küçük alt gruplara ayrılmaktadır.

2. Yerel modeller, merkezi olmayan yaklaşıma benzer şekilde her istemcinin verileri üzerinde eğitilmektedir.

3. Yerel modeller, her bir alt grubun merkezi sunucusunda toplanmaktadır.

4. Her alt grubun merkezi sunucuları, yerel modellerini bir araya getirmek ve genel modeli güncellemek için birbirleriyle iletişim kurmaktadır.

5. Global model, çıkarım veya ileri eğitim için dağıtılmaktadır.



Şekil 3.4 : Hibrit haberleşme

Hibrit haberleşme mimarisinin avantajları arasında gelişmiş model doğruluğu, azaltılmış iletişim maliyetleri ve verimli kaynak kullanımı yer almaktadır. Ancak hibrit yaklaşım, model birleştirmenin artan karmaşıklığı ve merkezi sunucular arasında daha fazla koordinasyon ihtiyacı gibi bazı zorluklarla da karşılaşabilmektedir.

3.3. İletişim Protokolü

Federe Öğrenme, ham verilerini paylaşmalarını gerektirmeden bir makine öğrenmesi modelini eğitmek için farklı cihazlar veya düğümler arasındaki iletişimi içermektedir. Bu nedenle, Federe öğrenme sistemlerinde optimum performansı elde etmek için güvenilir ve verimli bir iletişim protokolü gereklidir.

Federe öğrenmede kullanılan popüler bir iletişim çerçevesi Uzaktan Prosedür Çağrısı (gRPC)'dır. gRPC, Google tarafından geliştirilmiş, istemci ve sunucu uygulamaları arasında iletişimi sağlayan açık kaynaklı bir uzaktan prosedür çağrısı (RPC) sistemidir. Veri alışverişi için Protocol Buffers veri serileştirme mekanizmasını kullanır ve Python, C++ ve Java gibi çeşitli programlama dillerini desteklemektedir (Sliwa ve diğ., 2021).

gRPC, Federe öğrenme için çeşitli avantajlar sağlamaktadır. İlk olarak, yüksek performanslı iletişim için tasarlanmıştır ve büyük hacimli verileri hızlı ve verimli bir şekilde işleyebilmektedir. Aktarım için HTTP/2 kullanır ve hem tekli hem de akışlı RPC'leri destekler, bu da onu farklı türde Federe öğrenme görevlerini işleyecek kadar esnek kılmaktadır. gRPC'nin bir başka avantajı da, Federe öğrenme için gerekli olan çift yönlü iletişimi desteklemesidir. Hem istemcinin hem de sunucunun, Federe öğrenme sistemindeki düğümler arasında daha dinamik ve etkileşimli bir iletişim sağlayarak istekleri ve yanıtları başlatabileceği anlamına gelmektedir.

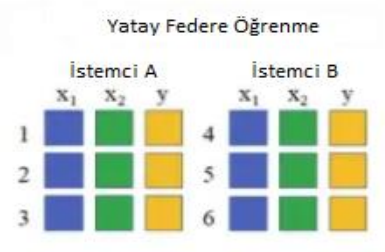
Ayrıca gRPC, iletişim sisteminin hataya dayanıklı olmasını ve ağdaki beklenmeyen arızaların üstesinden gelebilmesini sağlayan yerleşik yük dengeleme ve yük devretme desteği sağlamaktadır. Ayrıca, Federe öğrenme sistemindeki düğümler arasında değiş tokuş edilen verilerin güvenli ve özel olmasını sağlayan Aktarım Katmanı Güvenliğini (TLS) kullanarak iletişim için güvenli bir kanal sağlamaktadır.

3.4. Federe Öğrenme Türleri

Federe öğrenme yatay federe öğrenme, dikey federe öğrenme ve bölünmüş öğrenme olarak üç farklı öğrenme türüne sahiptir.

3.4.1. Yatay Federe öğrenme

Yatay Federe Öğrenme, aynı türde verilere ancak farklı veri örneklerine sahip birden fazla istemciyi içeren bir Federe öğrenme türüdür. Bu tür bir öğrenme, amaç tek bir cihazda veya makinede depolanamayacak kadar büyük olan büyük bir veri kümesi üzerinden bir model eğitmek olduğunda kullanışlıdır. Şekil 3.5’de istemcilerin aynı özellikleri paylaştığı görülmektedir.



Şekil 3.5 : Yatay öğrenme (Chen ve diğ., 2020)

Federe Yatay Öğrenme'de istemciler yerel modellerini ilgili veri kümelerinde eğitir ve güncellenen model parametrelerini merkezi bir sunucuya göndermektedir. Sunucu, tüm istemcilerden gelen parametreleri toplar ve daha sonra istemcilere geri gönderilen genel modeli günceller. Global modeli güncelleme işlemi, istenen doğruluk düzeyine ulaşılana kadar tekrarlanmaktadır.

Federe Yatay Öğrenimin ana avantajı, veri gizliliğini korurken çok büyük veri kümeleri üzerinde modellerin eğitilmesine izin vermesidir. İstemciler yalnızca güncellenen model parametrelerini sunucuya gönderdiği için ham veriler istemci cihazlarda kalmakta ve verilerin gizliliği korunmaktadır.

Matematiksel olarak, Federe Yatay Öğrenme aşağıdaki gibi temsil edilebilir:

N istemci olsun ve her *i* istemcisinin kendi yerel veri kümesi D_i olsun. Global model θ olarak gösterilsin. *t* yinelemesinde, istemci *i*'nin yerel modeli $\theta_{i,t}$ olarak temsil edilir ve eğitimden sonra güncellenen yerel model $\theta_{i,t+1}$ olur. Yerel modellerin toplanmasından sonra güncellenen küresel model, θ_{t+1} olarak gösterilir.

Federe Yatay Öğrenim için güncelleme denklemi aşağıdaki gibi gösterilebilir:

$$\theta_{t+1} = \sum_{i=1}^{n} \left(w_i \cdot \theta_{i,t+1} \right) \tag{3.8}$$

burada w_i, istemci *i*'ye atanan ağırlığı temsil etmektedir.

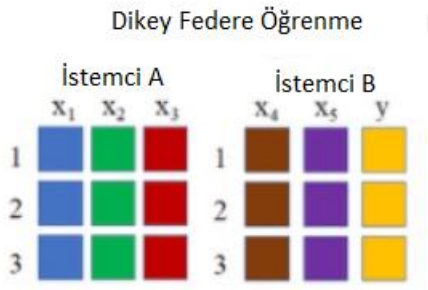
Bu şekilde, tüm istemcilerden güncellenen yerel modellerin ağırlıklı toplamı alınarak global model güncellenmektedir. Her istemciye atanan ağırlıklar, yerel veri

kümelerindeki örnek sayısı veya hesaplama yetenekleri gibi çeşitli yöntemlerle belirlenebilmektedir.

Bununla birlikte, yatay Federe öğrenme aynı zamanda veri heterojenliği, iletişim maliyetleri ve model yakınsaması gibi sorunlar da dahil olmak üzere çeşitli zorluklar sunmaktadır. Her bir katılımcı tarafından tutulan verilerdeki farklılıklara rağmen modelin tatmin edici bir çözüme yakınsayabilmesini sağlamak, yatay Federe öğrenmede önemli bir zorluktur.

3.4.2. Dikey federe öğrenme

Federe Dikey Öğrenme, verilerin dikey olarak bölümlendiği başka bir Federe öğrenme türüdür. Katılan her cihazın aynı örnek grubu (sıralar) için farklı özelliklere (sütunlar) sahip olduğu anlamına gelmektedir (Dongqi ve diğ., 2022). Örneğin, bir sağlık hizmeti senaryosunda, bir hastanede hasta adları bulunurken diğerinde tıbbi test sonuçları bulunabilir böyle bir senaryoda yatay Federe öğrenme yerine dikey Federe öğrenme tercih edilmelidir.



Şekil 3.6: Dikey öğrenme (Chen ve diğ., 2020)

Federe Dikey Öğrenme'nin ana zorluğu, verilerin bölümlenmesinin geleneksel makine öğrenmesi tekniklerini gerçekleştirmeyi zorlaştırmasıdır (Li ve diğ., 2023). Bu sorunun üstesinden gelmek için, dikey olarak bölümlenmiş veriler üzerinde gizliliği koruyan hesaplamalar için homomorfik şifreleme kullanılabilmektedir. Federe Dikey Öğrenme'de, modelleri bir araya getirmekten ve katılımcı cihazlara geri göndermekten merkezi bir sunucu sorumludur.

Federe Dikey Öğrenme algoritması aşağıdaki gibi özetlenebilir:

1. Katılan her cihaz, dikey olarak bölümlenmiş verilerini şifreler.

2. Her cihaz, şifrelenmiş veriler üzerinde yerel bir model eğitir.

3. Yerel modeller merkezi sunucuya gönderilir.

4. Merkezi sunucu modelleri toplar ve güncellenen genel modeli katılımcı cihazlara geri gönderir.

5. Katılan cihazlar, küresel modelin şifresini çözer ve işlemi tekrarlar.

6. Federe Dikey Öğrenme'deki toplama adımı, güvenli toplam ve güvenli iç çarpım protokolleri kullanılarak yapılır. Güvenli toplam protokolü, merkezi sunucunun, her katılımcı cihazın model ağırlıklarını, bireysel model ağırlıklarını bilmeden özetlemesini sağlar. Güvenli dahili ürün protokolü, merkezi sunucunun model ağırlıklarının ve özellik değerlerinin iç çarpımını ikisini de bilmeden hesaplamasını sağlar.

Güvenli toplam protokolü matematiksel olarak şu şekilde tanımlanabilir:

i'nci cihazın model ağırlıkları w_i olsun ve toplu model ağırlıkları w' olsun. Güvenli toplam protokolü şu şekilde tanımlanır:

$$w' = E\left[\sum_{i=1}^{n}(w_i + e_i)\right] \tag{3.9}$$

burada E, şifreleme işlevidir ve e_i, gizlilik için her model ağırlığına eklenen ek gürültüdür.

Benzer şekilde güvenli iç çarpım protokolü de matematiksel olarak şu şekilde tanımlanabilir:

$$z' = E\left[\sum_{i=1}^{n}(w_i \cdot x_i + e_i)\right] \tag{3.10}$$

i'nci cihazın model ağırlıkları w_i olsun ve *i*'inci cihazın özellik değerleri x_i olsun. Burada *E* şifreleme işlevidir, e_i gizlilik için her model ağırlığına eklenen ek gürültüdür ve *z'* model ağırlıkları ve özellik değerlerinin şifrelenmiş nokta çarpımıdır.

3.4.3. Bölünmüş öğrenme

Bölünmüş öğrenme, merkezi öğrenmeye izin verirken veri gizliliğini sağlamak amacıyla federe öğrenmede kullanılan başka bir yaklaşımdır. Bölünmüş öğrenmede, model yerel ve uzak kısım olmak üzere iki kısma ayrılmakta ve veriler buna göre bölünmektedir (Singh ve diğ., 2019). Yerel kısım, yerel veriler üzerinde eğitilir ve çıktılar, bilgileri toplayan ve genel modeli güncelleyen uzak kısma gönderilmektedir.

Bölünmüş öğrenmede, veriler yerel cihazda tutulur ve sunucu ile yalnızca çıktılar paylaşılır. Böylece ağ üzerinden gönderilmesi gereken bilgi miktarı azaltılmaktadır ve veriler cihazdan asla çıkmadığından gizlilik arttırılmaktadır. Ayrıca, yerel model bağımsız olarak güncellendiğinden, her cihazın kendine özgü veri dağıtımını yansıtan kişiselleştirilmiş bir modeli olabilmektedir.

Resmi olarak, bölünmüş öğrenme şu şekilde tanımlanabilir: X girdi verileri, y hedef etiketler ve f model olsun. Bölünmüş öğrenme şu şekilde formüle edilebilir:

1. w_l parametreleriyle yerel modeli kullanarak $h = g(x; w_l)$ gizli temsilini hesaplayın; burada g, w_l ağırlıklarına sahip bir sinir ağıdır.
2. h gizli temsilini uzak sunucuya gönderir.
3. $y_{\text{hat}} = f(h; w_g)$ nihai çıktısını w_g parametreleriyle global modeli kullanarak hesaplayın; burada f, w_g ağırlıklarına sahip bir sinir ağıdır.
4. w_g global modelinin ağırlıklarını güncellemek için tahmini çıktı y_{hat} ile gerçek etiket y arasındaki farkı kullanır.

Genel olarak, bölünmüş öğrenme, merkezileştirilmiş öğrenmeye izin verirken mahremiyet sağlayabildiğinden, Federe öğrenme için umut verici bir yaklaşımdır. Bununla birlikte, hala yeni bir tekniktir ve potansiyelini ve sınırlamalarını tam olarak anlamak için daha fazla araştırmaya ihtiyaç vardır.

3.5. Optimizasyon Algoritmaları

Optimizasyon algoritmaları olarak da bilinen toplama yöntemleri, Federe öğrenmenin ayrılmaz bir parçasıdır. Global modelin doğruluğunu arttırmak için farklı katılımcıların modellerini toplama sürecinde hayati bir rol oynamaktadırlar. Federe öğrenmede, birden fazla istemcinin verileri vardır ve her istemci kendi verileri üzerinde yerel bir model eğitmektedir. Yerel modeller daha sonra verileri paylaşmadan daha doğru bir küresel model oluşturmak için toplanmaktadır. Toplama yöntemleri, yerel modellerin verimli bir şekilde birleştirilmesini ve küresel modelin doğruluğunun en üst düzeye çıkarılmasını sağlamaktadır. Bunu global modelin kayıp fonksiyonunu minimize ederek gerçekleştirmektedirler. Federe Ortalama Alma (FedAvg), Federe Stokastik Gradyan Düşüşü(FedSDG), Hata Toleransı Federe Ortalama Alma(Fault Tolerance FedAvg) dahil olmak üzere farklı toplama yöntemleri mevcuttur. Toplama yönteminin seçimi, ağın boyutu, veri miktarı ve mevcut hesaplama kaynakları gibi çeşitli faktörlere bağlıdır.

3.5.1. Federe ortalama alma

Federe Ortalama Alma (FedAvg), farklı cihazlarda eğitilmiş yerel modelleri toplayarak küresel bir model öğrenmeyi amaçlayan Federe öğrenmede kullanılan popüler bir toplama yöntemidir. FedAvg, Google tarafından ilk olarak 2016 yılında dağıtılmış öğrenmenin zorluklarına bir çözüm olarak önerilmiştir (McMahan ve diğ., 2016). FedAvg'nin arkasındaki temel fikir, daha sonra küresel modeli güncellemek için kullanılan birden fazla cihazdaki yerel modellerin ortalamasını hesaplamaktır (McMahan ve diğ., 2016). Bu şekilde FedAvg, makine öğrenmesi modellerinin gizliliği koruyan ve dağıtılmış eğitimine olanak tanır. FedAvg algoritması aşağıdaki matematiksel formülle temsil edilebilir:

$$w^{t+1} = \frac{\sum_{i=1}^{n} m_i w_i^t}{\sum_{i=1}^{n} m_i} \tag{3.11}$$

burada w^t, t yinelemesindeki genel modeli, w_i^t, t yinelemesindeki i aygıtındaki yerel modeli ve m_i, i aygıtındaki veri noktalarının sayısını temsil etmektedir.

FedAvg algoritması, ilk olarak eğitim sürecine katılmak için bir cihaz alt kümesi seçerek çalışmaktadır. Seçilen her cihaz, yerel bir modeli kendi verileri üzerinde eğitmekte ve ardından güncellenen modeli merkezi bir sunucuya göndermektedir. Sunucu daha sonra yeni bir eğitim turu başlatmak için cihazlara geri gönderilen yeni bir küresel model oluşturmak için FedAvg'yi kullanarak yerel modelleri toplamaktadır. FedAvg'nin gizliliği ve güvenliği korurken, Federe öğrenmede makine öğrenmesi modellerinin doğruluğunu iyileştirmede etkili olduğu görülmüştür.

FedAvg'ın yakınsamasını analiz etmek için çeşitli çalışmalar yapılmıştır (Glasgow ve diğ., 2021; Wang ve diğ., 2021). Çalışmalar, belirli varsayımlar altında, her istemcinin eğitim verilerinin yalnızca bir alt kümesine erişimi olmasına rağmen FedAvg'ın küresel bir optimal çözüme yakınsayabileceğini göstermiştir.

FedAvg, öğrenme hızı, parti boyutu, yerel eğitim yinelemelerinin sayısı ve kaç cihazdan veri alacağının sayısı gibi çeşitli hiperparametrelerin ayarlanmasını gerektirir. Hiperparametrelerin optimum değerleri, belirli soruna ve veri kümesine bağlı olarak değişebilir ve bu nedenle dikkatli ayarlama gerektirir.

Federe Ortalama'nın bazı avantajları ve dezavantajları vardır.

Federe Ortalama'nın Avantajları:

•Veri Gizliliği: FedAvg, ham veriler yerine yalnızca model güncellemeleri paylaşıldığından, katılan her cihazın verilerini gizli tutmasına izin vermektedir.

• Azaltılmış İletişim Maliyetleri: FedAvg, büyük miktarda veriyi merkezi bir sunucuya aktarmak yerine sunucuya yalnızca küçük model güncellemelerinin gönderilmesine izin vermektedir. Özellikle sınırlı bant genişliğine sahip cihazlar için faydalı olabilecek iletişim maliyetlerinin azalmasına neden olmaktadır.

•Daha İyi bir Model Genelleştirme: Ağırlıklı bir ortalama alma yöntemi kullanan FedAvg, belirli herhangi bir cihazın verilerine fazla uydurmanın önlenmesine yardımcı olmaktadır. Bu, modelin görünmeyen veriler üzerinde iyi performans gösterme yeteneği olan daha iyi model genellemesi ile sonuçlanabilmektedir.

•Ölçeklenebilirlik: FedAvg, çok sayıda cihaza sahip olanlar da dahil olmak üzere çeşitli Federe öğrenme senaryolarında kullanılabilmektedir ve iletişim maliyetlerinde önemli artışlar olmadan daha fazla cihazı barındıracak şekilde ölçeklenebilmektedir.

Federe Ortalama'nın Dezavantajları:

•Heterojen Veriler: Her cihazın verileri farklı olabileceğinden, veri dağılımı bağımsız ve aynı şekilde dağıtılmış (IID) olmayabilir, bu da iyi bir model yakınsaması elde etmeyi daha zor hale getirebilmektedir. Bu sorunu ele almak için, istemcileri benzer dağıtımlardan öğrenmeye teşvik etmek için amaç işlevine ek düzenleme terimleri getiren FedProx ve FedNova gibi çeşitli FedAvg varyasyonları önerilmiştir.

•Veri Dengesizliği: Federe bir öğrenme ortamında, her cihaz tarafından tutulan veri örneklerinin sayısı eşit olmayabilir, veri örneklerinin eşit olmaması da bazı cihazların model eğitim sürecinde yeterince temsil edilmemesine neden olabilmektedir.

•Straggler Sorunu: FedAvg'da, bazı cihazların yerel eğitim yinelemelerini tamamlaması daha uzun sürerse toplama süreci geciktirebilmekte ve genel eğitim sürecini yavaşlatabilmektedir.

Federe Ortalama üzerine çeşitli geliştirmeler yapılarak Ağırlıklı Federe Ortalama (Ağırlıklı FedAvg) ortaya çıkarılmıştır. Ağırlıklı Federe Ortalama, istemcilerin model güncellemeleri üzerinde farklı düzeylerde etkiye sahip olmasına izin veren bir FedAvg çeşididir. Her istemciye eğitim sürecindeki önemini yansıtan bir ağırlık atayarak elde edilmektedir.

3.5.2. QFedAvg

QFedAvg, istemcilerin veri dağıtımları arasındaki farkları hesaba katmak için temel Federe ortalama alma yaklaşımını ek bir terimle genişleten bir FedAvg çeşididir. QFedAvg, FedAvg stratejisine bir "q" parametresi ekleyerek fark yaratır (Li ve diğ., 2021; Li ve diğ., 2019). Standart FedAvg'de, verilerinin heterojenliğine bakılmaksızın tüm istemciler model güncellemesine eşit olarak katkıda bulunmaktadır. Bununla birlikte, pratik Federe öğrenme senaryolarında, istemciler, temsili olmayan ve önyargılı güncellemelere yol açabilecek farklı veri dağıtımlarına sahip olabilmektedir. QFedAvg,

model güncellemelerini toplarken istemci dağıtımlarının farklılığını göz önünde bulundurarak bu sorunu ele almayı amaçlamaktadır.

Matematiksel olarak, QFedAvg şu şekilde ifade edilebilir:

$$w^{t+1} = \frac{\sum_{i=1}^{n} q_i w_i^t}{\sum_{i=1}^{n} q_i} \tag{3.13}$$

burada q_i, veri dağılımı ile bir referans dağılımı arasındaki Kullback-Leibler (KL) sapmasına dayalı olarak i. istemciye atanan ağırlıktır. KL sapması, iki olasılık dağılımı arasındaki farkı ölçer ve genellikle bir farklılık ölçüsü olarak kullanılır. Referans dağılımı genellikle genel veri dağılımı veya bunun bir alt kümesi olarak ayarlanmaktadır.

q_i ağırlığı şu şekilde hesaplanır:

$$q_i = exp(-\lambda \times KL(p_i||q)) \tag{3.14}$$

p_i, i. istemcinin verilerinin olasılık dağılımı ve q, referans dağılımıdır. Hiper parametre lambdası, düzenlileştirmenin gücünü kontrol eder ve genel modelin önemi ile istemciye özel güncellemeler arasında denge sağlamaya yardımcı olmaktadır.

Federe öğrenmede, her istemci tarafından gönderilen güncellemeler tipik olarak oldukça büyük olabilen ve önemli miktarda iletişim bant genişliği gerektirebilen kayan noktalı sayılar biçimindedir. QFedAvg, güncellemeleri sunucuya gönderilmeden önce sıkıştırmak için niceleme kullanarak bu sorunu gidermektedir (Alistarh ve diğ., 2016). Özellikle QFedAvg, iki adımlı bir niceleme işlemi kullanmaktadır. Her şeyden önce, güncellemeler stokastik niceleme adı verilen bir teknik kullanılarak az sayıda bite (tipik olarak 1-2 bit) nicelenmektedir. Böylece güncellemelerin boyutunu azaltmakta ve iletişimin verimliliğini arttırmaktadır. Nicemlenen güncellemeler rasgele döndürme (Xing ve diğ., 2020) adı verilen bir teknik kullanılarak sıkıştırılmaktadır. Bu yöntem, güncellemeleri, etkili bir şekilde daha küçük bir alt uzaya sıkıştıran rastgele bir matris

kullanarak yüksek boyutlu bir uzayda döndürmeyi içermektedir. Sıkıştırılmış güncellemeler daha sonra daha az bant genişliği kullanılarak sunucuya gönderilebilmektedir. Güncellemeler sayısallaştıkça ve sıkıştırıldıkça, genel iletişim yükü azalmakta ve Federe öğrenmeyi daha büyük ve daha karmaşık modellere ölçeklendirmeyi mümkün kılmaktadır.

FedAvg'da olduğu gibi QFedAvg'da da çeşitli parametrelerin probleme göre ayarlanması gerekmektedir. Örneğin, her turda eğitim için kullanılacak istemcilerin oranı, her turda değerlendirme için kullanılacak istemcilerin sayısı, her turda eğitim için olması gereken minimum istemci sayısı gibi parametreler eğitim öncesi probleme göre ayarlanmaktadır. Hiper parametrelerin optimum değerleri, belirli soruna ve veri kümesine bağlı olarak değişebilir ve bu nedenle dikkatli ayarlama gerektirmektedir.

QFedAvg'ın FedAvg'a göre çeşitli avantajları vardır. İlk olarak, temsili olmayan verilerin etkisini azaltarak daha iyi yakınsama ve doğruluğa yol açabilmektedir. İkinci olarak, farklı veri dağıtımlarına sahip istemcilerin etkisini azaltarak mahremiyeti artıran bir etki sağlayabilmektedir. Üçüncüsü, heterojen verilerin ve etki alanı uyarlama görevlerinin işlenmesini sağlayabilmektedir.

QFedAvg'ın da bazı sınırlamaları vardır. KL sapmasının hesaplanması, istemcilerin veri dağıtımlarına erişim gerektirmektedir ve bu durum gizliliğe duyarlı ayarlarda zor olabilmektedir. Ayrıca, lambda'nın hiperparametre ayarı yanıltıcı olabilmekte ve yanlış ayarlar yetersiz sonuçlara yol açabilmektedir. Son olarak, KL sapma hesaplamasının ek hesaplama maliyeti, eğitim süresini ve kaynak tüketimini artırabilmektedir.

3.5.3. Hata toleranslı Federe ortalama

Hata Toleransı Federe Ortalama Alma (FT-FedAvg), Federe öğrenme ortamında düğüm hatalarına ve veri eskimesine karşı sağlamlık sağlayan Federe Ortalama (FedAvg) algoritmasının bir uzantısıdır. Geleneksel FedAvg'de, düğümler (istemciler) yerel olarak eğitilmiş model güncellemelerini, genel bir model güncellemesi elde etmek için ortalamalarını alan merkezi bir sunucuya göndermektedir ancak güvenilir olmayan düğümlerin veya iletişim gecikmelerinin varlığında, süreç etkilenebilmekte ve optimal olmayan küresel model güncellemelerine yol açabilmektedir. FT-FedAvg, düğümlerin modellerini çoğaltarak ve her düğümün modelinin en az bir başka düğüm tarafından

eğitilmesini sağlayarak sistemin düğüm hatalarından ve bayatlığından kurtulmasına izin vererek çeviklik sağlamaktadır.

FT-FedAvg, düğümleri gruplara ayırarak ve her gruba eğitilmesi için bir dizi model atayarak çalışmaktadır. Modeller, aynı genel modelle başlatılmaktadır ve atanan düğümler tarafından yerel veriler kullanılarak eğitilmektedir. Daha sonra eğitilen modellerin ortalaması alınarak tüm düğümlere gönderilen yeni bir genel model oluşturulmaktadır. Başarısızlık veya gecikme nedeniyle eğitim görevini tamamlamayan istemciler, eğitimlerine devam etmek için yeni genel modeli kullanabilmekte ve modelin parametrelerinin tüm düğümlerde hala senkronize olmasını sağlamaktadır. Böylece FT-FedAvg, herhangi bir istemci cihazda sorun olduğunda eğitimi durdurmak yerine, eğitim turlarının sorunsuz olan cihazlarla devam etmesini sağlamaktadır (Morell ve diğ., 2022; Yang ve diğ., 2021). Ve yalnızca sorunsuz çalışan cihazlardan güncelleme almaktadır. FT-FedAvg, hata düzeltme kodlaması ve uyarlamalı toplama olan iki temel mekanizma ile bunları sağlamaktadır. Hata düzeltme kodlaması, her istemci tarafından gönderilen model güncellemelerine fazlalık eklemeyi içermektedir. Bu sayede sunucunun fazlalık bilgilerin kodunu çözerek eksik veya bozuk güncellemeleri kurtarması sağlanmaktadır. Etkilenen istemcilerin güncellemelerini yeniden iletmelerini gerektirmeden arızalardan ve kesintilerden kurtulmayı mümkün kılmaktadır. Uyarlanabilir toplama, her istemcinin güvenilirliğine dayalı olarak sunucu tarafından kullanılan toplama ağırlıklarının dinamik olarak ayarlanmasını içermektedir. Toplama sırasında, güvenilir güncellemeler geçmişi olan istemcilere daha yüksek bir ağırlık verilirken, başarısızlık veya kapalı kalma süresi yaşayan istemcilere daha düşük bir ağırlık verilmektedir. Daha düşük bir ağırlık, güvenilir olmayan istemcilerin küresel model üzerindeki etkisini azaltmaya ve genel performansı iyileştirmeye yardımcı olmaktadır. Küresel modelin zayıf bağlantıya sahip düğümlerden gelen yavaş veya gecikmeli güncellemelerden etkilenmemesi sağlanmış olmaktadır.

Matematiksel olarak, FT-FedAvg şu şekilde formüle edilebilir:

$$w^{t+1} = \frac{\sum_{i=1}^{n} w_i^t \cdot w_{i,s(i)}^t}{\sum_{i=1}^{n} w_{i,s(i)}^t} \tag{3.15}$$

burada w_i^t, i düğümünün t yinelemesindeki modelidir, $w_{i,s(i)}^t$, s(i) düğümünün modelidir; burada s(i), her bir i düğümünü diğerine atayan bir fonksiyondur aynı model üzerinde eğitimi tamamlamış düğüm s(i) ve n, düğümlerin toplam sayısıdır. s(i) fonksiyonu rasgele veya deterministik olarak seçilebilir ve fazlalığı sağlamak için her bir düğüm birden fazla düğüme atanabilir.

Genel olarak, FT-FedAvg, düğüm arızaları ve iletişim gecikmelerinin varlığında eğitimli modelin doğruluğunu ve sağlamlığını sağlayan, Federe öğrenmeye hataya dayanıklı bir çözüm sunmaktadır. FT-FedAvg, Federe öğrenmede iletişim başarısızlıkları sorununun üstesinden gelmek için umut verici bir yaklaşımdır. İstemcilerin iletişim hatalarından kurtulmalarına ve eğitim sürecine katılımlarını sürdürmelerine olanak tanıyarak sistemin sağlamlığını artırabilmektedir. Özellikle başarısızlıkların yaygın olduğu ve genel sistemin performansını önemli ölçüde etkileyebileceği büyük ölçekli Federe öğrenme sistemlerinde kullanışlıdır. Bununla birlikte FT-FedAvg bazı sınırlamalara sahiptir. En büyük dezavantajı, özellikle iletişim arızalarının sık olduğu durumlarda, sistemin iletişim yükünü önemli ölçüde artırabilmesidir. Diğer bir sınırlama ise FT-FedAvg'ın eğitim verisi dağıtımının durağan olduğunu varsaymasıdır ki bu gerçek dünya uygulamalarında her zaman geçerli olmayabilir. Başka bir deyişle, modeli eğitmek için kullanılan eğitim verilerinin dağılımının, sistemin başarısız bir düğümden kurtarıldığı zamanlar da dahil olmak üzere tüm eğitim süreci boyunca aynı kaldığını varsaymaktadır fakat gerçek dünyadaki uygulamalarda durum her zaman böyle olmayabilmektedir. Eğitim süreci sırasında veya sistem bir arızadan kurtulurken veri dağılımı değişirse, durağan veri dağılımı varsayımı ihlal edilebilir, bu da yetersiz performansa ve hatta FT-FedAvg algoritmasının başarısız olmasına yol açabilmektedir.

3.6. Veri Bölümleme

Verilerin eğitim için cihazlar arasında nasıl dağıtılacağını belirlediği için veri bölümleme, Federe öğrenmede kritik bir adımdır. Federe öğrenmede, veriler tipik olarak istemciler veya düğümler olarak da bilinen cihazlar arasında bölünmektedir. Veri bölümlemenin amacı, eğitim sırasında cihazlar arasında ihtiyaç duyulan iletişim

miktarını en aza indirirken, her cihazın genel veri dağılımının temsili bir örneğine erişimi olmasını sağlamaktır.

Federe öğrenmede veri bölümleme için birkaç yaygın strateji vardır:

1. Rastgele bölümleme: Federe öğrenmede verileri birden fazla cihaz arasında bölmek için kullanılan yaygın bir tekniktir. Rastgele bölümleme yaklaşımında, veriler rasgele çakışmayan alt kümelere bölünür ve her alt küme farklı bir cihaza atanır (Mahmud ve diğ., 2019). Alt kümelerin sayısı genellikle Federe öğrenme sürecine katılan cihazların sayısına eşittir. Rastgele veri bölümleme, veriler hakkında herhangi bir ek bilgi gerektirmediğinden nispeten basit ve uygulanması kolaydır. Uygulama kolaylığının yanında bazı sınırlamaları vardır. Ana sınırlamalardan biri, bazı aygıtların diğerlerinden çok daha fazla veriye sahip olduğu yüksek düzeyde dengesiz veri bölümlerine yol açabilmesidir. Dengesiz veri bölümlemeleri, daha az veriye sahip cihazların model eğitim sürecine yeterince katkı sağlayamaması durumunda düşük performansa yol açabilmektedir.

2. Tabakalı bölümleme: Bu yaklaşım, verilerin önceden tanımlanmış kategorilere veya etiketlere göre bölümlenmesini içermektedir (Hu ve diğ., 2020). Örneğin, bir sınıflandırma görevinde, veriler sınıf etiketlerine göre bölümlere ayrılabilmektedir. Tabakalı bölümleme, her cihazın her sınıf veya kategoriyi temsil eden bir örneğe sahip olmasını sağlamaya yardımcı olabilir ve modelin genel doğruluğunu artırabilmektedir.

3. Küme tabanlı bölümleme: Verilerin bazı benzerlik ölçütlerine göre kümeler halinde gruplandığı bir veri bölümleme yaklaşımıdır. Amaç, her kümenin birbirine benzer ve diğer kümelerdekilere benzemeyen veri noktaları içermesini sağlamaktır. Bu bölümleme yaklaşımı, veri dağıtımını korumak ve katılan cihazlar arasındaki iş yükünü dengelemek için Federe öğrenmede yaygın olarak kullanılmaktadır. Federe öğrenmede küme tabanlı bölümleme gerçekleştirmenin birkaç yöntemi vardır. Yaygın bir yaklaşım, veri noktalarının benzerliklerine göre k kümeler halinde kümelenmesini içeren k-ortalamalı kümelemedir (Ghosh ve diğ., 2020). Başka bir yaklaşım, veri noktalarının benzerliklerine göre bir küme hiyerarşisinde gruplandırılmasını içeren hiyerarşik kümelemedir. Küme tabanlı bölümlemenin rastgele bölümlemeye göre birçok avantajı vardır. Birincisi, verilerin cihazlar arasında dağılımının benzer olmasını sağlamaya yardımcı olabilmekte ve bu da federe öğrenme modelinin performansını

artırabilmektedir. Ek olarak, verileri benzerliklere dayalı olarak kümeleyerek, verilerdeki diğer bölümleme yaklaşımlarında belirgin olmayabilecek kalıpları ve eğilimleri belirlemeye yardımcı olmasına faydalıdır. Bununla birlikte, küme tabanlı bölümleme için bazı sınırlamalar da vardır. Potansiyel sorunlardan biri, yüksek düzeyde gürültü veya aykırı değerlere sahip veri kümeleri için uygun olmayabilmesidir çünkü veri noktaları kendi kümelerine atanabilmektedir (Hard ve diğ., 2018). Bu durum da verilerin genel dağıtımını etkileyebilme potansiyeline sahiptir. Ek olarak, küme tabanlı bölümleme, özellikle büyük veri kümeleri için veya daha karmaşık kümeleme algoritmaları kullanılırken hesaplama açısından pahalı olabilmektedir.

4. Özellik tabanlı bölümleme: Verileri, verilerin özelliklerine veya özniteliklerine göre bölümlere ayıran bir bölümleme yöntemidir. Bu yöntemde, veriler belirli bir özelliğin veya niteliğin değerlerine göre alt kümelere bölünmektedir. Örneğin, görüntü sınıflandırma görevlerinde veriler, görüntülerin renk, doku veya şekil özelliklerine göre bölümlenebilir. Özellik tabanlı bölümleme, farklı özelliklerin farklı dağıtımlara veya önem düzeylerine sahip olduğu senaryolarda faydalı olabilmektedir (Cheung ve diğ., 2020). Verileri özelliklere göre bölümlere ayırarak model, belirli özellik veya nitelikle en alakalı olan veri alt kümeleri üzerinde eğitilebilir ve modelin doğruluğunu ve verimliliğini artırabilir. Özellik tabanlı bölümlemenin bir örneği, verileri farklı cihazlardaki farklı özelliklere göre bölen "yatay" bölümleme yaklaşımıdır. Her cihaza, üzerinde eğitim yapılacak bir özellik alt kümesi ve karşılık gelen veri örnekleri atanmaktadır. Her cihazdan eğitilen modeller daha sonra tüm özellikler üzerinde eğitilmiş genel bir model elde etmek için bir araya getirilebilmektedir. Özellik tabanlı bölümlemeye başka bir örnek, verileri aynı özelliğin farklı özelliklerine göre farklı cihazlar arasında bölen "dikey" bölümleme yaklaşımıdır (Das ve diğ., 2021). Örneğin, bir sağlık hizmeti veri kümesinde, bir cihaz kan basıncı özelliğini eğitmek üzere atanabilirken, başka bir cihaz kolesterol özelliğini eğitmek üzere atanabilmektedir. Her cihazdan eğitilen modeller daha sonra tüm özellikler üzerinde eğitilmiş genel bir model elde etmek için birleştirilebilmektedir. Özellik tabanlı bölümleme, Federe öğrenme sistemlerinin performansını ve verimliliğini artırabilen güçlü bir tekniktir ancak optimum bölümleme yaklaşımını belirlemek için veri kümesindeki özelliklerin dikkatli bir şekilde değerlendirilmesi ve analiz edilmesi gerekmektedir.

Veri bölümleme stratejisinin seçimi, verilerin belirli özelliklerine ve eldeki göreve bağlı olacaktır. Genel olarak, eğitim sırasında cihazlar arasında ihtiyaç duyulan iletişim miktarını en aza indirirken, her cihazın genel veri dağılımının temsili bir örneğine erişimi olmasını sağlamak önemlidir.

Bazı senaryolarda, verilerin dağılımı zaman içinde değişebilir ve dinamik veri bölümleme gerektirebilmektedir. Dinamik veri bölümleme, bölümleme stratejisinin değişen veri dağılımına uyarlanmasını içermektedir ve uyarlama işlemi zorlu olabilir ancak etkili federe öğrenme için gerekli olabilmektedir.

Son olarak, belirli zorlukları ele almak için birden fazla yöntemi birleştiren veri bölümlemeye yönelik hibrit yaklaşımlar olduğunu belirtmek gerekmektedir. Örneğin, Federe öğrenme algoritmasının etkinliğini korurken mahremiyet endişelerini ve iletişim yükünü dengelemek için rastgele bölümleme ve katmanlı bölümlemenin bir kombinasyonu kullanılabilmektedir. Buradaki fikir, verileri çakışmayan alt kümelere bölmek için önce rasgele bölümlemeyi kullanmak ve ardından daha fazla bölüm oluşturmak için her bir alt kümeye katmanlı bölümleme uygulamaktır. Hibrit bir yaklaşım kullanmanın avantajı, hem rasgele bölümlemenin hem de katmanlı bölümlemenin faydalarından yararlanabilmesidir. Rastgele bölümleme, her cihazın farklı bir veri kümesi almasını sağlamaya yardımcı olabilirken, katmanlı bölümleme, verilerde daha ince taneli kalıpların yakalanmasına yardımcı olabilir ve potansiyel olarak modelin performansını artırabilmektedir.

Bununla birlikte, hibrit yaklaşımların uygulanmasının daha karmaşık olabileceğini ve tek bir bölümleme yöntemi kullanmaktan daha fazla hesaplama kaynağı gerektirebileceğini belirtmekte fayda vardır. Ek olarak, hibrit bir yaklaşımın etkinliği, veri kümesinin ve öğrenme görevinin belirli özelliklerine bağlı olacaktır. Bu nedenle, belirli bir Federe öğrenme senaryosu için en uygun yaklaşımı belirlemek üzere farklı veri bölümleme yöntemlerini dikkatlice değerlendirmek ve karşılaştırmak önemlidir.

3.7. Sıkıştırma Metotları

Federe öğrenme, birden fazla cihazın verileri merkezileştirmeden bir model eğitim görevi üzerinde işbirliği yapmasını sağlamaktır. Federe öğrenmedeki ana zorluklar, sınırlı iletişim bant genişliğini, hesaplama gücünü ve cihazların depolama kapasitesini

içermektedir. Ana zorluklar, modelin performansından ödün vermeden iletişim yükünü azaltmak için yeni sıkıştırma tekniklerinin geliştirilmesini gerekli kılmaktadır.

Niceleme tabanlı sıkıştırma, iletimden önce modelin parametrelerini sıkıştırmak için federe öğrenmede yaygın olarak kullanılmaktadır. Bu yöntem model parametrelerinin bit sayısını azaltmaktadır (Sattler ve diğ., 2020). Seyreltme tabanlı sıkıştırma ise modelin bazı parametrelerini sıfıra ayarlayarak sıkıştırmayı gerçekleştirmektedir. Degrade sıkıştırma ve bilgi damıtma gibi sıkıştırma yöntemleri de tercih edilebilmektedir.

3.8. Federe Öğrenme Avantajları

Federe öğrenme mekanizması, geleneksel merkezi makine öğrenmesi yaklaşımlarına göre çeşitli avantajlar sağlamaktadır (Huang ve diğ., 2020).

1. Geliştirilmiş veri gizliliği: Federe öğrenme, kuruluşların hassas veriler üzerinde, bu verileri üçüncü taraflarla paylaşmak zorunda kalmadan modeller eğitmesine olanak tanımaktadır. Veri gizliliği düzenlemelerinin sıkı olduğu sağlık veya finans gibi alanlarda özellikle önemlidir. Geleneksel bir merkezi öğrenme yaklaşımında, veriler kullanıcılardan toplanmaktadır ve eğitim için merkezi bir sunucuya gönderilmektedir. Bununla birlikte, Federe öğrenme ile veriler kullanıcı cihazlarında kalmakta ve bu da veri ihlallerini ve yetkisiz erişimi önlemeye yardımcı olmaktadır. Veriler kullanıcı cihazlarında kaldığından, kullanıcıların verilerini başkalarıyla paylaşmalarına gerek yoktur. Verilerin kötüye kullanılması veya yetkisiz erişim riskini azaltmaktadır. Federe öğrenme ile kullanıcılar, verileri üzerinde daha fazla kontrole sahip olmaktadır. Eğitim sürecine katılıp katılmamayı seçebilmektediler böylece ne kadar veri katkı sağlayacaklarına kendileri karar verebilmektedirler. Bu durum kullanıcılar ve verilerini toplayan kuruluşlar arasında güven oluşturmaya yardımcı olmaktadır. Veri gizliliği adına Federe öğrenim, eğitim süreci sırasında kullanıcı verilerini korumak için şifreli veri aktarımını kullanmaktadır. Şifreli aktarımlar verilerin gizli ve güvenli kalmasını sağlamaktadır.

2. Kaynakların verimli kullanımı: Federe öğrenme ile eğitim süreci birden çok cihaza veya düğüme dağıtılarak kaynakların daha verimli kullanılması sağlanmaktadır. Böylece tüm verileri depolamak ve işlemek için merkezi bir sunucuya olan ihtiyacı

azaltarak daha hızlı eğitim süreleri ve daha düşük kaynak maliyetleri avantajları sunmaktadır. Verilerin hassas veya büyük olduğu ve işlenmek üzere tamamının merkezi bir konuma iletilmesini pratik olmayan hale getirdiği durumlarda özellikle önemlidir. Federe öğrenme ise verilerin cihazların kendilerinde yerel olarak işlenmesini sağlayarak büyük miktarda verinin merkezi bir sunucuya iletilmesi ihtiyacını azaltmaktadır. Ek olarak, Federe öğrenme, hesaplama birden çok cihaza dağıtıldığı için modelleri eğitmek için gereken süreyi ve enerjiyi azaltabilmektedir. Bu, önemli maliyet tasarrufları ve makine öğrenmesi açısından daha sürdürülebilir bir yaklaşımla sonuçlanabilir. Genel olarak, Federe öğrenme, kaynak kullanımının verimliliğini artırabilir ve makine öğrenmesinde daha sürdürülebilir uygulamalar sağlayabilmektedir.

3. Azaltılmış iletişim maliyetleri: Federe öğrenme yalnızca merkezi sunucu ile katılımcı cihazlar arasında model güncellemelerinin değiş tokuşunu gerektirdiğinden, büyük miktarda verinin iletilmesiyle ilişkili iletişim maliyetlerini büyük ölçüde azaltabilmektedir. Geleneksel bir merkezi makine öğrenmesi yaklaşımında, model eğitimi için tüm veriler merkezi bir sunucuya iletilmektedir. Bu durum hesaplama açısından pahalı olabilir ve özellikle veri kümesi büyükse veya kullanıcı sayısı yüksekse, büyük miktarda iletişim bant genişliği gerektirebilir.

Federe öğrenmede sunucu ve istemci cihazlar arasında ham veriler yerine yalnızca model güncellemeleri iletilmektedir. Böylece sunucu ile istemciler arasında gereken iletişim miktarı büyük ölçüde azaltılmakta ve genel iletişim maliyetlerini azaltmaktadır. Ayrıca niceleme ve seyrekleştirme gibi sıkıştırma teknikleri kullanılarak, her bir iletişim turunda iletilmesi gereken veri miktarı daha da azaltılabilmektedir. Özellikle yavaş veya güvenilir olmayan internet bağlantılarına sahip mobil cihazlar gibi istemcilerin sınırlı iletişim bant genişliğine sahip olduğu durumlarda önemlidir.

İletişim maliyetlerinin düşürülmesi sadece istemciler için değil, aynı zamanda sunucu için de faydalıdır. Daha az iletişim ile sunucu daha fazla sayıda istemciyi işleyebilir ve eğitim daha hızlı tamamlanarak sistemin genel verimliliği arttırılabilmektedir.

4. Geliştirilmiş model doğruluğu: Federe öğrenme, daha büyük ve daha çeşitli cihazlardan alınan veriler üzerinde eğitim alarak daha sağlam ve doğru modellere yol açabilmektedir. Bu durum modellerin daha çeşitli verilere erişebileceği anlamına gelmektedir ve daha iyi genelleme ve gelişmiş doğruluk sağlayabilmektedir. Veri

dağıtımı değişimlerine dayanıklılık da gelişmiş model doğruluğuna katkı sunmaktadır. Geleneksel makine öğrenmesinde, verilerin dağılımı önemli ölçüde değişirse, model yeni veriler üzerinde iyi performans göstermeyebilir. Federe öğrenme de ise modeller birden fazla cihazdan gelen yeni veriler üzerinde sürekli olarak eğitilebildiği için bu sorunun hafiflemesine yardımcı olabilmektedir. Sürekli eğitim modellerin verilerin dağılımındaki değişikliklere uyum sağlamasına ve zaman içinde doğruluklarını arttırmasına yardımcı olabilmektedir. Model doğruluğunu geliştirmede en önemli husus birden fazla taraf arasında işbirliği yapılabilmesidir. Model, daha büyük ve daha çeşitli bir veri kümesi üzerinde eğitilebileceğinden, gelişmiş doğruluk da sağlanabilecektir. Ek olarak, işbirliği, verilerdeki hataların veya önyargıların belirlenmesine ve düzeltilmesine yardımcı olabilmektedir.

5. Gerçek zamanlı güncellemeler: Federe öğrenme, yeni veriler kullanıma sunuldukça modellerin gerçek zamanlı olarak güncellenmesini sağlamaktadır. Geleneksel merkezi öğrenmede, model büyük bir veri kümesi üzerinde gruplar halinde eğitilmekte ve güncellemeler, model parametrelerini güncellemek için merkezi bir sunucuya gönderilmektedir. Bu süreç yavaştır ve önemli miktarda zaman ve kaynak gerektirmektedir. Ancak Federe öğrenmede eğitim birden çok cihaza dağıtıldığı için güncellemeler gerçek zamanlı olarak işlenebilir. Modelin sürekli olarak eğitilebileceği ve yeni veriler kullanılabilir hale geldikçe hızla güncellenebileceği ve gerçek zamanlı karar vermeyi mümkün kıldığı anlamına gelmektedir. Özellikle altta yatan verilerin sürekli değiştiği veya yeni veri edinme maliyetinin yüksek olduğu durumlarda kullanışlıdır.

6. Ölçeklenebilirlik: Federe öğrenme, büyük ve coğrafi olarak dağılmış cihaz popülasyonlarını barındıracak şekilde kolayca ölçeklendirilebilmektedir. Böylece Nesnelerin İnterneti (IoT) ve uç bilgi işlem gibi uygulamalar için ideal bir çözüm haline gelmektedir. Federe öğrenme, kuruluşların milyonlarca cihaza dağıtılan devasa veri kümeleri üzerinde modellerin eğitmesine olanak tanımaktadır. Ölçeklenebilirlik, veri kümesini, ayrı cihazlar tarafından işlenebilen çok sayıda daha küçük veri kümesine bölerek elde edilmektedir. Federe öğrenme, cihazların modelleri yerel olarak öğrenmesine izin verdiği ve cihazlar ile merkezi bir sunucu arasında veri aktarımı ihtiyacını ortadan kaldırdığı için büyük veri kümelerine ölçeklenebilmektedir. Ayrıca, birden fazla cihazın aynı modelden aynı anda öğrenmesine izin vererek hesaplama

gereksinimlerinde önemli azalmalar sağlamaktadır. Bu dağıtılmış öğrenme yaklaşımı, kuruluşların daha önce mümkün olandan çok daha büyük veri kümeleriyle modelleri eğitmesine olanak tanıyarak daha doğru modellere ve gelişmiş performansa yol açmaktadır. Sonuç olarak, Federe öğrenimin ölçeklenebilirliği Federe öğrenmeyi sağlık, finans ve telekomünikasyon gibi büyük ölçekli makine öğrenmesi uygulamaları gerektiren sektörler için ideal bir çözüm haline getirmektedir.

Genel olarak, Federe öğrenme, geleneksel merkezi öğrenme yöntemlerine göre bir dizi avantaj sunarak, veri gizliliğini ve verimliliğini korurken makine öğrenmesi yeteneklerini geliştirmek isteyen kuruluşlar için güçlü bir araç haline getirmektedir.

3.9. Federe Öğrenme Zorlukları

1. Veri heterojenliği: Federe öğrenmenin önemli zorluklarından biri, farklı cihazlar arasındaki veri heterojenliğidir. Federe öğrenmede, veriler birden fazla cihaza dağıtılır ve her cihazın farklı veri dağıtımları, veri boyutları ve veri kalitesi olabilir (Huang ve diğ., 2020). Bu durum heterojenliğe yol açmaktadır dolayısıyla doğruluk ve mahremiyetten ödün vermeden farklı cihazlardan modellerin toplanmasında önemli bir zorluk ortaya çıkabilmektedir.

Heterojenlik, modeli eğitmek için kullanılan verilerin dağılımının test sırasında karşılaşılan verilerin dağılımından farklı olduğu, etki alanı kayması olarak bilinen bir olguya yol açabilmekte, gerçek dünya senaryolarında dağıtıldığında ise modelin düşük performansına neden olmaktadır. Bu zorluğun üstesinden gelmek için Federe öğrenme yaklaşımlarının, veri heterojenliği sorununu ele alan etki alanı uyarlaması, veri normalleştirme ve veri arttırmı gibi teknikleri içermesi gerekmektedir.

Etki alanı uyarlaması, kaynak ve hedef alanlar arasındaki özellik dağılımlarını hizalayarak modeli yeni hedef alanlara uyarlamayı amaçlayan bir tekniktir. Maksimum ortalama tutarsızlık veya çekişmeli eğitim gibi yöntemler kullanılarak iki alan arasındaki tutarsızlığı en aza indirerek elde edilebilmektedir. Veri normalleştirme, veri dağılımının tüm cihazlarda benzer olmasını sağlamak için özellikleri farklı cihazlar veya konumlar arasında standartlaştırmayı içeren başka bir tekniktir. Özellikleri sıfır ortalama ve birim varyansa sahip olacak şekilde ölçeklendirerek veya toplu normalleştirme gibi diğer normalleştirme teknikleri kullanılarak elde edilebilir. Veri arttırma, eğitim

kümesini çoğaltmak için sentetik veri üretmeyi ve böylece model için mevcut eğitim verisi miktarını arttırmayı içeren bir tekniktir. Verilerin cihazlar arasında dağılımında önemli bir dengesizlik olduğu veya verilerin oldukça seyrek veya gürültülü olduğu durumlarda yararlı olabilir.

Veri heterojenliğinden daha spesifik bahsetmek gerekirse bağımsız olmayan ve aynı şekilde dağıtılan verilere değinmek gerekmektedir. IID olmayan veriler, Federe öğrenmedeki başlıca zorluklardan biridir. Geleneksel makine öğrenmesi ayarlarında, veriler tipik olarak ortak bir dağıtımdan bağımsız ve aynı şekilde örneklenmektedir. Bununla birlikte, Federe öğrenmede, veriler genellikle farklı veri dağılımlarına sahip olabilen çeşitli kaynaklardan toplanır ve IID olmayan verilerle sonuçlanır.

Bu sorun, bir veri kümesi üzerinde eğitilen modeller başka bir veri kümesi üzerinde iyi performans göstermeyebileceğinden, katılan tüm istemcilere genellenebilen eğitim modellerinde zorluklara yol açabilmektedir. Sonuç olarak, federe öğrenmede IID olmayan verileri hesaba katabilecek yöntemlerin geliştirilmesi esastır.

Ortak bir yaklaşım, amacın birden çok görev veya veri kümesinde iyi performans gösterebilen paylaşılan bir modeli öğrenmek olduğu Federe meta-öğrenmeyi kullanmaktır. Başka bir yaklaşım, modelin büyük bir veri kümesi üzerinde eğitildiği ve ardından her istemcideki yerel veriler üzerinde ince ayarın yapıldığı transfer öğrenimini kullanmaktır.

Ek olarak, her istemcinin global modele katkısının veri dağılımına göre ağırlıklandırıldığı ağırlıklı toplama gibi, Federe öğrenme algoritmasını IID olmayan verileri işlemek üzere ayarlamak için kullanılabilecek çeşitli yöntemler vardır. Alternatif olarak, her istemcide eğitilen yerel modeller, verilerin IID olmayan doğasına uyum sağlamak için önceden işlenebilir veya sonradan işlenebilir.

Genel olarak, veri heterojenliği, Federe öğrenmede önemli bir zorluktur, ancak uygun tekniklerin kullanılmasıyla, dağıtılmış cihazlar veya konumlar arasında etkili ve doğru model eğitimi sağlamak için ele alınabilir.

2. İletişim sınırlamaları:

İletişim sınırlamaları, özellikle çok sayıda katılımcının olduğu büyük ölçekli uygulamalar için Federe öğrenme önemli bir zorluktur. İletişim darboğazı, yavaş veya

güvenilir olmayan ağ bağlantıları, sınırlı bant genişliği veya yüksek gecikme süresi gibi çeşitli faktörlerden kaynaklanabilmektedir. Model birleştirme sürecinde gecikmelere neden olarak daha uzun eğitim sürelerine ve düşük performansa yol açabilmektedir. Ek olarak, katılımcıların kaybolan veya geciken mesajları telafi etmek için daha fazla veri iletmesi gerekebileceğinden, iletişim sınırlamaları da daha yüksek iletişim maliyetlerine neden olabilir.

Zorlukların üstesinden gelmek için, araştırmacılar çeşitli yaklaşımlar önerdiler. Yaklaşımlardan biri katılımcılar ile merkezi sunucu arasında iletilen veri miktarını azaltmak için sıkıştırılmış veya nicelenmiş modeller kullanmaktır. Başka bir yaklaşım ise büyük miktarda veriyi daha verimli bir şekilde işleyebilen GRPC veya Apache Kafka gibi özel iletişim protokollerini kullanmaktır. Ayrıca, bazı araştırmalar, hesaplamaların ve veri depolamanın katılımcı cihazlara daha yakın bir yere dağıtıldığı ve ağ üzerinden iletilmesi gereken veri miktarının azaldığı uç bilişim kullanımını araştırmıştır.

Bu çabalara rağmen, iletişim sınırlamaları, özellikle büyük ölçekli uygulamalar için federe öğrenmede önemli bir sorun olmaya devam etmektedir. Araştırmacılar, daha verimli iletişim protokollerinin geliştirilmesi, uç bilgi işlemin kullanımı ve sıkıştırma ve niceleme tekniklerinin optimizasyonu dahil olmak üzere, bu zorluğun üstesinden gelmek için yeni yaklaşımlar ve çözümler keşfetmeye devam etmektedir. Federe öğrenme, iletişim sınırlamalarını ele alarak, daha büyük ve daha karmaşık uygulamaları barındıracak şekilde ölçeklendirilebilir ve daha düşük iletişim maliyetleriyle daha hızlı ve daha doğru model eğitimi sağlamaktadır.

3. Model yanlılığı ve adalet:

Model yanlılığı ve adalet, Federe öğrenmede başka bir zorluktur. Federe öğrenme kullanılarak eğitilen modeller, belirli veri gruplarına karşı yanlı olabilir ve bu da haksız tahminlere yol açabilmektedir. Bu, verilerin eşit olmayan şekilde dağıtılmasından veya verilerin her cihazda temsil edilememesinden kaynaklanabilir. Taraflı veriler üzerinde eğitilen modeller, mevcut önyargıları daha da sürdürebilmektedir. Bu zorluğun üstesinden gelmek için çeşitli teknikler önerilmiştir. Yaklaşımlardan biri, modelin tahminlerinin herhangi bir belirli gruba karşı tarafsız olmasını sağlamayı amaçlayan eğitim sürecine adalet kısıtlamalarını dahil etmektir. Başka bir yaklaşım, mevcut

önyargıları belirlemek ve azaltmak için eğitim verileri üzerinde yanlılık analizi yapmaktır. Ek olarak, sunucuda toplanan farklı cihazlardan gelen model ağırlıklarını birleştirirken kullanılan optimizasyon algoritmalarından bazıları da bu soruna çözüm olarak kullanılmaktadır.

Bununla birlikte, veri toplamadan model eğitimi ve değerlendirmeye kadar sürecin herhangi bir aşamasında önyargılar ortaya çıkabileceğinden, Federe öğrenmede adaleti sağlamanın karmaşık ve devam eden bir zorluk olduğuna dikkat etmek önemlidir.

4. Cihaz heterojenliği:

Cihaz heterojenliği, Federe öğrenmeye katılan cihazlar arasındaki farklılıkları ifade etmektedir. Bu farklılıklar donanım, yazılım veya ağ yetenekleri olabilmektedir. Örneğin, bazı aygıtlar daha güçlü bilgi işlem gücüne sahipken, diğerleri sınırlı depolama kapasitesine veya kararsız ağ bağlantılarına sahiptir. Heterojenlik, önemli performans düşüşüne ve iletişim yüküne yol açabileceğinden, Federe öğrenmede önemli bir zorluk oluşturmaktadır.

Cihaz heterojenliğini ele almak için literatürde çeşitli teknikler önerilmiştir. Yaklaşımlardan biri, katılımcı cihazların yeteneklerine dayalı olarak model güncellemelerini dinamik olarak ayarlayabilen uyarlanabilir öğrenme algoritmaları kullanmaktır. Örneğin, daha güçlü bilgi işlem gücüne sahip cihazlara daha karmaşık görevler atanabilirken, sınırlı yeteneklere sahip cihazlara daha basit görevler atanabilir.

Başka bir yaklaşım, model güncellemelerinin boyutunu azaltabilen ve bunların ağ üzerinden iletilmesini kolaylaştırabilen model sıkıştırma tekniklerini kullanmaktır. Sınırlı depolama kapasitesine veya ağ bant genişliğine sahip cihazlar için özellikle yararlı olabilir.

Cihaz heterojenliğini ele almaya yönelik bir başka yaklaşım ise aktif cihaz seçimini kullanmaktır. Aktif cihaz seçimi, Federe öğrenme görevi için en uygun olan belirli özelliklere sahip cihazları seçme sürecidir. Cihaz seçme süreci, hesaplama gücüne, ağ bant genişliğine ve kullanılabilir depolama kapasitesine göre cihazların seçilmesini içerebilmektedir. Görev için en uygun cihazları seçerek Federe öğrenme sisteminin genel performansı ve verimliliği iyileştirilebilmektedir.

Cihaz heterojenliğini ele almaya yönelik bir başka yaklaşım, hata toleransı tekniklerini kullanmaktır. Bazı cihazlar arızalansa veya kullanılamaz hale gelse bile Federe öğrenme sisteminin çalışmaya devam etmesini sağlamak için hata toleransı teknikleri kullanılmaktadır. Teknikler, verilerin ve modellerin birden fazla cihazda çoğaltılmasını içerebilir, böylece bir cihaz arızalanırsa sistem diğer cihazları kullanarak çalışmaya devam edebilmektedir. Ek olarak, öğrenme sürecini cihazların performansına ve kaynakların mevcudiyetine göre ayarlamak için uyarlanabilir öğrenme oranı çizelgeleri ve dinamik kaynak tahsisi gibi teknikler kullanılabilmektedir.

3.10. Federe Öğrenme Frameworkleri

Federe öğrenme çerçeveleri, Federe öğrenme modellerini geliştirmek ve dağıtmak için bir platform sağlayan yazılım araçlarıdır. Bu çerçeveler, dağıtılmış öğrenme algoritmalarının uygulanmasını kolaylaştırır ve veri gizliliğini ve güvenliğini korurken mobil cihazlar veya uzak sunucular gibi merkezi olmayan veri kaynakları üzerinde makine öğrenmesi modellerinin eğitimine olanak tanımaktadır.

3.10.1. Flower

Flower, Federe öğrenme modelleri oluşturmak ve dağıtmak için bir dizi araç ve API sağlayan açık kaynaklı bir Federe öğrenme çerçevesidir (Beutel ve diğ., 2022). Arka uç kitaplıkları olarak PyTorch ve TensorFlow'u kullanır ve çok çeşitli makine öğrenmesi algoritmalarını desteklemektedir. Bu çalışmada Flower çerçevesi tercih edilmiştir. Flower'ın avantajları aşağıdaki gibidir:

1. Basitleştirilmiş Geliştirme: Flower, Federe öğrenme için geliştiricilerin sistem altyapısı yerine algoritma tasarımına ve değerlendirmesine odaklanmasını sağlayan basitleştirilmiş bir API sağlamaktadır. API'ler, Federe öğrenme sistemlerini uygulamayı kolaylaştırır ve denemeye başlamak için gereken süreyi azaltmaktadır.

2. Dağıtılmış Bilgi İşlem: Flower çerçevesi, Federe öğrenme için esnek ve verimli bir dağıtılmış bilgi işlem platformu sağlamaktadır. Kullanıcıların, hesaplamaları birden fazla cihaz ve makine arasında kolayca dağıtmasına izin vererek, büyük ölçekli veri kümelerindeki modelleri dağıtılmış bir şekilde eğitmeyi mümkün kılmaktadır.

3. Özelleştirilebilir Algoritmalar: Flower çerçevesi, kullanıcıların farklı Federe öğrenme algoritmalarını kolayca özelleştirmesine ve bunlarla deney yapmasına olanak tanımaktadır. Çok çeşitli makine öğrenmesi modellerini ve optimizasyon algoritmalarını destekleyerek farklı kullanım durumları için çeşitli Federe öğrenme algoritmalarının uygulanmasını sağlamaktadır.

4. Gizliliği Koruma: Flower çerçevesi, diferansiyel gizlilik ve güvenli toplama gibi çeşitli gizliliği koruma teknikleri için yerleşik destek sağlamaktadır. Hassas kullanıcı verilerinin korunmasına yardımcı olmakta ve Federe öğrenme sürecinin gizliliğin korunmasını sağlamaktadır.

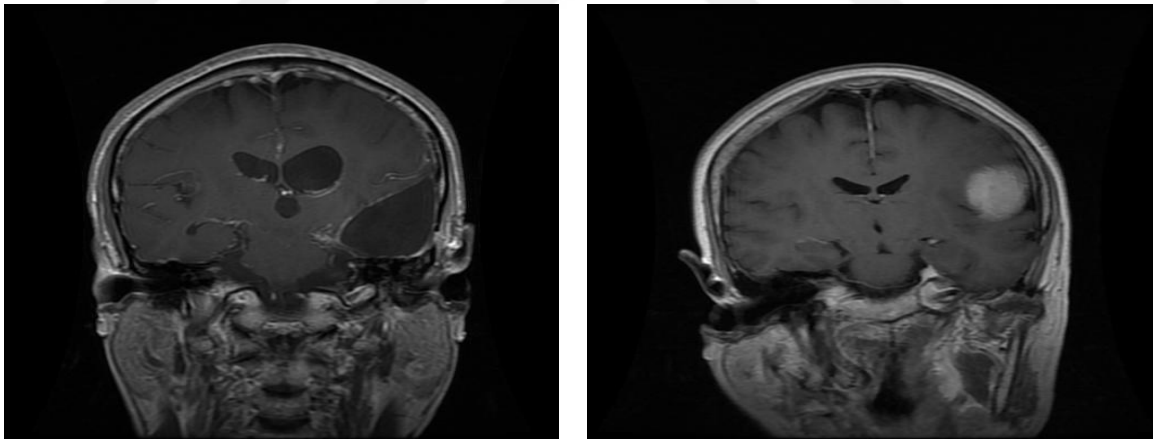
5. Aktif Geliştirme: Flower çerçevesi, bir geliştirici topluluğu tarafından aktif olarak geliştirilmekte ve sürdürülmektedir. Yeni özellikler ve iyileştirmelerle düzenli olarak güncellendiği ve hataların hızla düzeltildiği anlamına gelmektedir.

Özetle, Flower çerçevesi, federe öğrenme sistemleri oluşturmak için kullanıcı dostu ve esnek bir platform sağlamaktadır. Basitliği, dağıtılmış bilgi işlem yetenekleri, özelleştirilebilirliği, gizliliği koruyan özellikleri ve aktif geliştirmesi, onu federe öğrenme algoritmalarını uygulamak ve bunlarla deney yapmak için güçlü bir araç haline getirmektedir. Flower çerçevesi yukarıda bahsedilen özellikleri sebebiyle çalışmamız da tercih edilmiştir. Flower dışında Google tarafından geliştirilen TensorFlow Federe (TFF), PySyft, WeBank tarafından geliştirilen FATE, Microsoft tarafından geliştirilen FLUTE ve FedML çerçeveleri de bulunmaktadır.

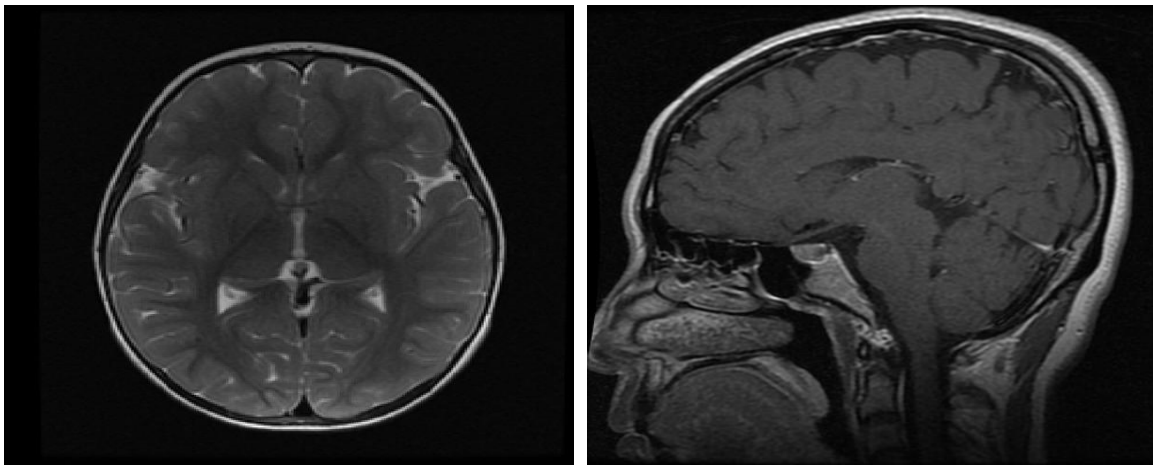
BÖLÜM 4. DENEYSEL ÇALIŞMA

4.1. Veri Kümesi

Çalışma kapsamında, halka açık Kaggle web sitesinden sağlanan MRI veri kümesi kullanılmıştır (URL-1, 2021). Bu veri kümesi figshare, SARTAJ ve Br35h isimli veri kümelerinin birleştirilmesi ile oluşturulmuştur (URL-2, 2017; URL-3, 2020; URL-4, 2020). Veri kümesi başın önünden, yanından, arkasından ve tepesinden alınan MRI görüntülerinden oluşmaktadır. Veri kümesinin tamamı 7023 örnek içermektedir. Veri kümesi glioma, meningioma, pituitary ve tümörsüz olmak üzere 4 farklı sınıf içermektedir. Örnekler Şekil 4.1 ile verilmiştir. Sınıfların dağılımı Şekil 4.2'de gösterilmiştir. Örnekler 512 x 512 boyutundadır.



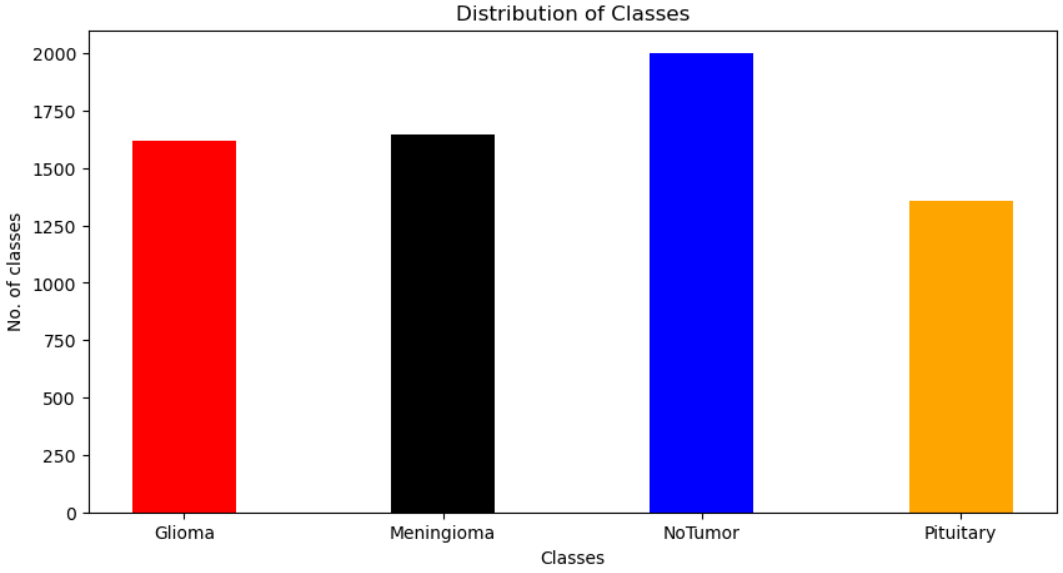
(a) (b)



(c) (d)

Şekil 4.1: Her sınıf için veri örneği (a) Glioma (b) Meningioma (c) No Tumor (d) Pituitary

Ön işleme çalışmalarında görüntüler 128 x 128 boyutuna düşürülmüştür. Burada amaç, görüntülerin parametrelerini ve dolayısıyla model çalışmalarının maliyetlerini azaltmaktır. Veri kümesi eğitim ve test olarak ikiye ayrılmıştır. İstemciler ve sunucular birbirinden ayrılmadan önce eğitim için toplam 5712 örnek ve test için 1311 örnek bulunmaktadır.



Şekil 4.2: Sınıfların dağılımı

4.2. İstemciler Arası Veri Paylaşımı

Federe öğrenmede, yüksek model performansı elde etmek için cihazlar arasında veri paylaşımı esastır. Verilerin paylaşılması, katılan cihazların işbirliği yapmasına ve genel veri dağıtımında iyi performans gösteren bir modeli topluca öğrenmesine olanak tanımaktadır. Ayrıca veri kümesindeki sınıfların cihazlar arasındaki dağılımı da kritiktir. Federe öğrenmede, veriler genellikle farklı cihazlar arasında dağıtılır ve her cihazın benzersiz bir veri alt kümesi vardır. Her aygıtın tuttuğu veri alt kümesi tipik olarak kimliği belirsizdir; bu, sınıfların dağılımının aygıtlar arasında farklı olabileceği anlamına gelmektedir. Bu durum modelin bir cihazın verilerinde iyi, diğerinin verilerinde kötü performans gösterdiği önyargılı model eğitiminde sorunlara yol açabilmektedir. Bu çalışmada cihazlar arası dağılımı Tablo 4.1'de verilmiştir. Çalışmada üç istemci ve bir sunucu simüle edilmiştir. Tablo 4.1'de de görüldüğü gibi veriler istemciler ve sunucu arasında eşit dağılımda bölünülmüştür.

Tablo 4.1 : İstemciler ve sunucu arası veri paylaşımı

	Glioma	Meningioma	No Tumor	Pituitary
Client 1 - Train	330	335	398	365
Client 1 - Test	75	75	104	75
Client 2 - Train	330	336	398	364
Client 2 - Test	75	75	100	75
Client 3 - Train	330	334	398	364
Client 3 - Test	75	75	101	75
Server - Train	331	334	401	364
Server - Test	75	81	100	75

Federe öğrenmede, sınıf sayısının istemciler arasında eşit ve adil dağılımı, hem yerel modelin hem de küresel modelin geliştirilmesi üzerinde önemli bir etkiye sahip olabilmektedir. İlk olarak, sınıf sayısı istemciler arasında eşit olarak dağıtılmazsa, bazı istemciler belirli sınıflar hakkında diğerlerinden daha fazla veriye sahip olabilir ve bu da yanlı yerel model güncellemelerine yol açmaktadır. Bu, bazı istemcilerin küresel modelde gereğinden fazla veya yetersiz temsil edilmesiyle sonuçlanabilir ve bu da belirli sınıflarda düşük performans gösteren çarpık bir modele yol açmaktadır.

Öte yandan, sınıf sayısı istemciler arasında eşit olarak dağıtıldığında ise her istemci, her sınıf için küresel modelin geliştirilmesine katkıda bulunmak için eşit fırsatlara sahip olmaktadır. Eşit dağılım tüm sınıflarda iyi performans gösterebilen daha dengeli bir modelle sonuçlanmaktadır. Ayrıca, federe öğrenmede küresel modelin doğruluğu, her istemciden gelen yerel model güncellemelerinin kalitesine bağlıdır. Bu nedenle, istemciler arasında sınıf sayısının eşit ve adil bir şekilde dağıtılmasının sağlanması, önyargıyı azaltarak ve eğitim için kullanılan verilerin çeşitliliğini artırarak küresel modelin genel performansını iyileştirebilmektedir.

Genel olarak, etkili veri paylaşımı ve yönetimi, cihazların gizlilik ve güvenliği korurken verilerden toplu olarak öğrenmesine izin verdiği için federe öğrenmenin başarısı için kritik öneme sahiptir.

4.3. Evrişimli Sinir Ağı'nın Oluşturulması ve Uygulanması

Çalışma kapsamında cihazlar arası işbirliği için 3 katmanlı CNN modeli kullanılmıştır. Veri kümesi federe öğrenme yapısını gerçekleştirmek için dört parçaya bölündüğünde istemcilerdeki ve sunucudaki veri sayısı Tablo 4.1'de görüldüğü üzere azalmıştır. Dolayısıyla kompleks bir model geliştirmek modeli aşırı öğrenmeye götürebileceği için katman sayısı düşük tutulmak istenmiştir. Hem sunucu hem de istemci cihazlar aynı model mimariye sahiptir.

Federe öğrenme, veri gizliliğinden ödün vermeden dağıtılmış veri kümelerinde makine öğrenmesi modellerini eğitmek için umut verici bir yaklaşımdır. Bu yaklaşımda, yerel modeller, bireysel cihazlarda depolanan veriler üzerinde eğitilmekte ve bunların güncellemeleri, görünmeyen veriler üzerinde iyi performans gösterebilen küresel bir model oluşturmak için toplanmaktır. Bununla birlikte, federe öğrenmenin performansı

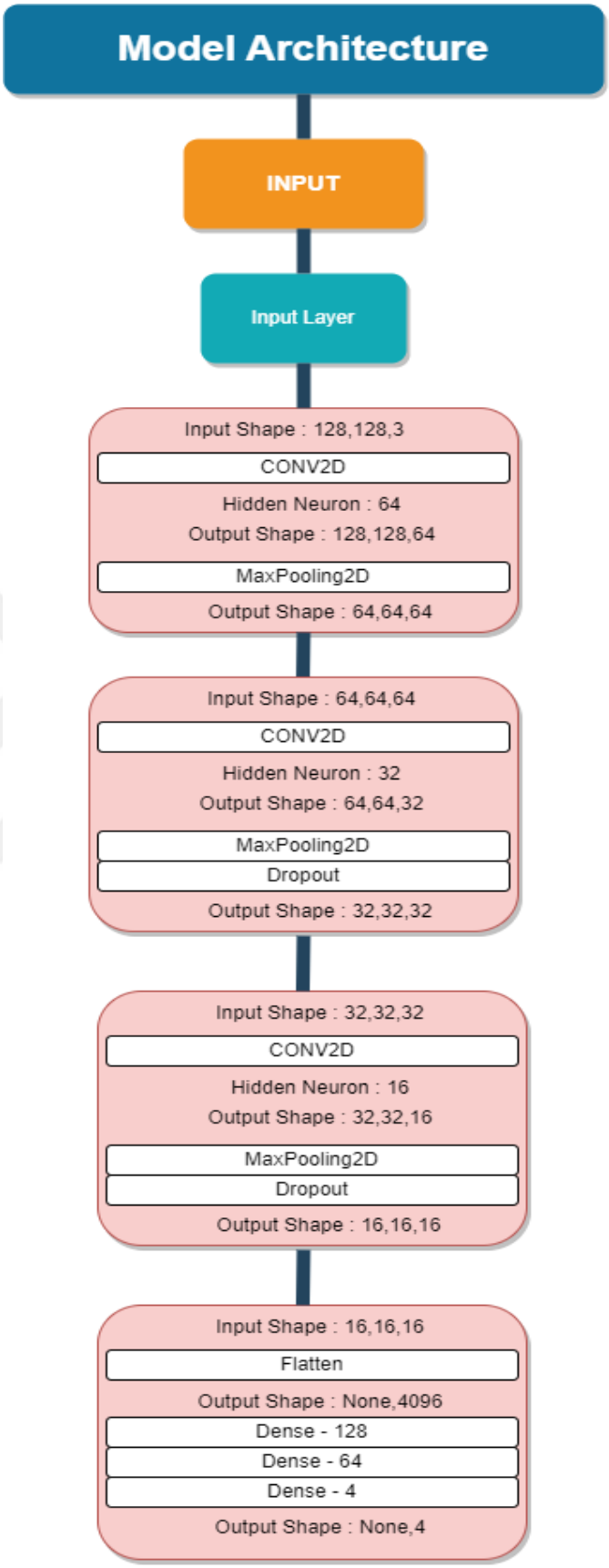
büyük ölçüde yerel modellerin kalitesine ve yeni verilere genelleme yapma becerilerine bağlıdır. Yerel modellerin kalitesini arttırmanın bir yolu, bir CNN mimarisi kullanmaktır.

CNN mimarisinin federe öğrenmeye uygun olmasının birkaç nedeni vardır. İlk olarak, federe öğrenmede bireysel cihazlarda depolanan veriler, genellikle CNN'ler tarafından verimli bir şekilde işlenebilen görüntüleri veya diğer görsel verileri içermektedir. CNN'ler, görüntüleri doğru bir şekilde sınıflandırmak için çok önemli olan pikseller arasındaki karmaşık kalıpları ve ilişkileri öğrenebilmektedir. Buna karşılık, karar ağaçları veya doğrusal modeller gibi geleneksel makine öğrenmesi modelleri, bu tür kalıpları yakalamakta zorlanabilir ve elle uygulanan özellikler gerektirebilmektedir.

İkincisi, CNN'ler farklı görüntü boyutlarına ve şekillerine kolayca uyarlanabilmektedir. Federe öğrenmede, veriler farklı çözünürlüklere ve en boy oranlarına sahip farklı cihazlardan gelebilmektedir. CNN'ler, önemli özellikleri korurken girdiyi alt örnekleyen evrişimli ve havuzlama katmanlarını kullanarak bu varyasyonları işleyebilir.

Üçüncüsü, aşırı uydurmayı önlemek ve genellemeyi iyileştirmek için CNN'ler düzenli hale getirilebilmektedir. Federe öğrenmede, yerel modeller küçük veri alt kümeleri üzerinde eğitilmekte ve bu da modeller çok karmaşıksa fazla uydurmaya yol açabilmektedir. CNN'ler, aşırı uyumu önlemek ve yerel modellerin performansını arttırmak için bırakma, ağırlık azaltma veya erken durdurma gibi tekniklerle düzenli hale getirilebilmektedir.

Model mimarisi Şekil 4.3'te verilmiştir.



Şekil 4.3: Model mimarisi

Bu metodolojide, CNN mimarisi kullanılarak Federe öğrenme için yerel bir model oluşturulmuştur. Model, sırasıyla 64, 32 ve 16 filtreli üç evrişimli katmandan oluşmaktadır ve her birini havuz boyutu (2,2) olan bir maksimum havuzlama katmanı izlemektedir. Fazla uydurmayı önlemek için her maksimum havuzlama katmanından sonra 0.2 oranında seyreltme katmanları eklenmiştir. Üçüncü maksimum havuzlama katmanının çıktısı düzleştirilmiştir ve her ikisi de ReLU aktivasyon fonksiyonlarına sahip sırasıyla 128 ve 64 nöronlu iki yoğun katmana beslenmiştir.

Model mimarisinde değerlendirilmesi gereken ilk hiperparametre, evrişim katmanlarındaki filtrelerin sayısıdır. Filtreler, belirli özellikleri çıkarmak için giriş görüntüsünü tarayan küçük matrislerdir ve her katmandaki filtre sayısı CNN'nin derinliğini belirlemektedir. Genel olarak, filtre sayısını arttırmak, CNN'nin karmaşık kalıpları yakalama yeteneğini geliştirebilir ancak aynı zamanda modele aşırı uyum sağlama riskini de arttırmaktadır. Bizim durumumuzda, üç evrişimli katman için sırasıyla 64, 32 ve 16 filtreye karar verilmiştir. Bu karar, görevle ilgili önceki bilgilere ve veri kümesinin boyutuna dayanmaktadır ve ayrıca optimum değerleri bulmak için bir ızgara arama yaklaşımı kullanarak farklı sayıda filtre denenmiştir.

Öğrenme hızı, gradyan iniş sırasında optimize edici tarafından atılan adım boyutunu belirleyen bir başka kritik hiper parametredir. Yüksek bir öğrenme oranı, minimumun aşılmasına ve yakınsamanın başarısız olmasına neden olabilirken, düşük bir öğrenme oranı, yavaş yakınsamaya veya yerel minimumda takılıp kalmaya neden olabilmektedir. Deney yoluyla ve modelin yakınsama davranışını izleyerek CNN modeli için 0.0001'lik bir öğrenme oranı seçilmiştir.

Parti boyutu, eğitim sürecinin her yinelemesinde kullanılan eğitim örneklerinin sayısını belirleyen başka bir hiper parametredir. Daha büyük bir parti boyutu, eğitim sürecini hızlandırabilir ancak aynı zamanda daha az genelleme becerisine ve daha fazla uyum sağlamaya yol açabilmektedir. CNN modelinde bir dizi deneyle optimal olduğunu belirlediğimiz 8'lik bir toplu iş boyutu kullanılmıştır.

Bırakma katmanları, modeli düzenli hale getirmeye ve fazla uydurmayı önlemeye yardımcı oldukları için CNN mimarisinin önemli bir parçasıdır. Oluşturulan CNN modelinde her maksimum havuzlama katmanından sonra 0,20 oranında bırakma katmanları eklenmiştir. Farklı bırakma oranları da denenmiştir ve 0,20'lik bir oranın

model karmaşıklığı ile düzenli hale getirme arasında iyi bir denge sağladığı görülmüştür.

Çalışmada aktivasyon fonksiyonu olarak ReLU kullanılmıştır. Son olarak, sınıflandırma amacıyla 4 nöronlu yoğun bir katman ve bir softmax aktivasyon fonksiyonu eklenmiştir.

CNN modelini optimize etmek için Adam optimize edici kullanılmıştır. Adam optimize edici, ağdaki her ağırlık için tarihsel gradyanların ve tarihsel kareli gradyanların üssel olarak azalan bir ortalamasını tutmaktadır. Ardından, her ağırlık güncellemesi için uyarlanabilir bir öğrenme oranı hesaplamak için bu hareketli ortalamaları kullanmaktadır. Bu, öğrenme oranının her ağırlık için uygun şekilde ölçeklenmesini ve güncellemelerin en dik inişe doğru yapılmasını sağlamaya yardımcı olmaktadır. Optimize edici, önceki deneylere ve ampirik gözlemlere dayalı olarak seçilen 0,0001'lik bir öğrenme oranı ve 0,0001'lik bir bozulma oranı ile başlatılmıştır.

Kullanılan kayıp fonksiyonu kategorik çapraz entropidir. İlk olarak, kategorik çapraz entropi kaybı fonksiyonu, bu çalışmadaki CNN modeli gibi çok sınıflı sınıflandırma görevlerini gerçekleştiren modeller için özellikle uygundur. Bunun nedeni, kayıp fonksiyonunun, sınıfların tahmin edilen olasılık dağılımı ile sınıfların gerçek olasılık dağılımı arasındaki farkı ölçmek için tasarlanmış olmasıdır. Başka bir deyişle, modelin tahmin edilen olasılıklarının gerçek olasılıklarla ne kadar örtüştüğünü ölçmektedir.

İkinci olarak, kategorik çapraz entropi, bu çalışmada kullanılan Adam optimize edici gibi stokastik gradyan iniş optimizasyon algoritmalarında kullanım için uygun kılan türevlenebilir bir fonksiyondur. Bunun nedeni, optimize edicinin model parametrelerini, parametrelere göre kayıp fonksiyonunun gradyanına dayalı olarak güncellemesidir. Son olarak, kategorik çapraz entropi, derin öğrenme alanında yaygın olarak kullanılan ve köklü bir kayıp fonksiyonudur ve birçok farklı türde sınıflandırma görevinde iyi çalıştığı gösterilmiştir.

Model, 5 dönem ve 8 parti boyutu ile eğitilmiştir. Bir modeli çok fazla dönem için eğitmek modelin eğitim veri kümesinde iyi performans gösterdiği ancak yeni, görünmeyen verilerde kötü performans gösterdiği aşırı uydurmaya yol açabilmektedir. Öte yandan, bir modeli çok az dönem için eğitmek modelin eğitim verilerinden yeterince öğrenmediği ve hem eğitim hem de test veri kümelerinde düşük performans gösterdiği yetersiz uyumla sonuçlanabilmektedir.

Bu çalışmada, model nispeten düşük görünebilecek 5 dönem için eğitilmiştir. Bununla birlikte, dönem sayısının seçimi, veri kümesinin boyutu ve karmaşıklığı, modelin mimarisi, üç istemciden gelen işbirlikçi model bilgileri ve mevcut bilgi işlem kaynakları dahil olmak üzere çeşitli faktörlere bağlıdır. Bu durumda, eğitim için kullanılan veri kümesinin nispeten küçük olması ve kullanılan CNN mimarisinin aşırı derecede karmaşık olmaması da ayrı faktörlerdir.

Dönem sayısı seçiminin, modelin doğrulama kümesindeki performansının ampirik değerlendirmesine dayanması gerektiğine dikkat etmek önemlidir. Başka bir deyişle, dönem sayısı, modelin hem eğitim hem de doğrulama setlerinde aşırı uyum olmadan iyi performans elde edeceği şekilde seçilmelidir. Bu çalışmada, modelin doğrulama kümesindeki performansının 5 dönem eğitimi sonrasında değerlendirilmesi ve tatmin edici bulunması, modelin eğitimi için 5 dönem seçimine yol açmıştır.

Genel olarak, çalışmadaki CNN mimarisi, aşırı uydurmayı önlemek için uygun etkinleştirme işlevleri ve düzenlileştirme teknikleri kullanılarak görüntü verilerini verimli bir şekilde işlemek ve onu dört olası sınıftan birine sınıflandırmak için tasarlanmıştır.

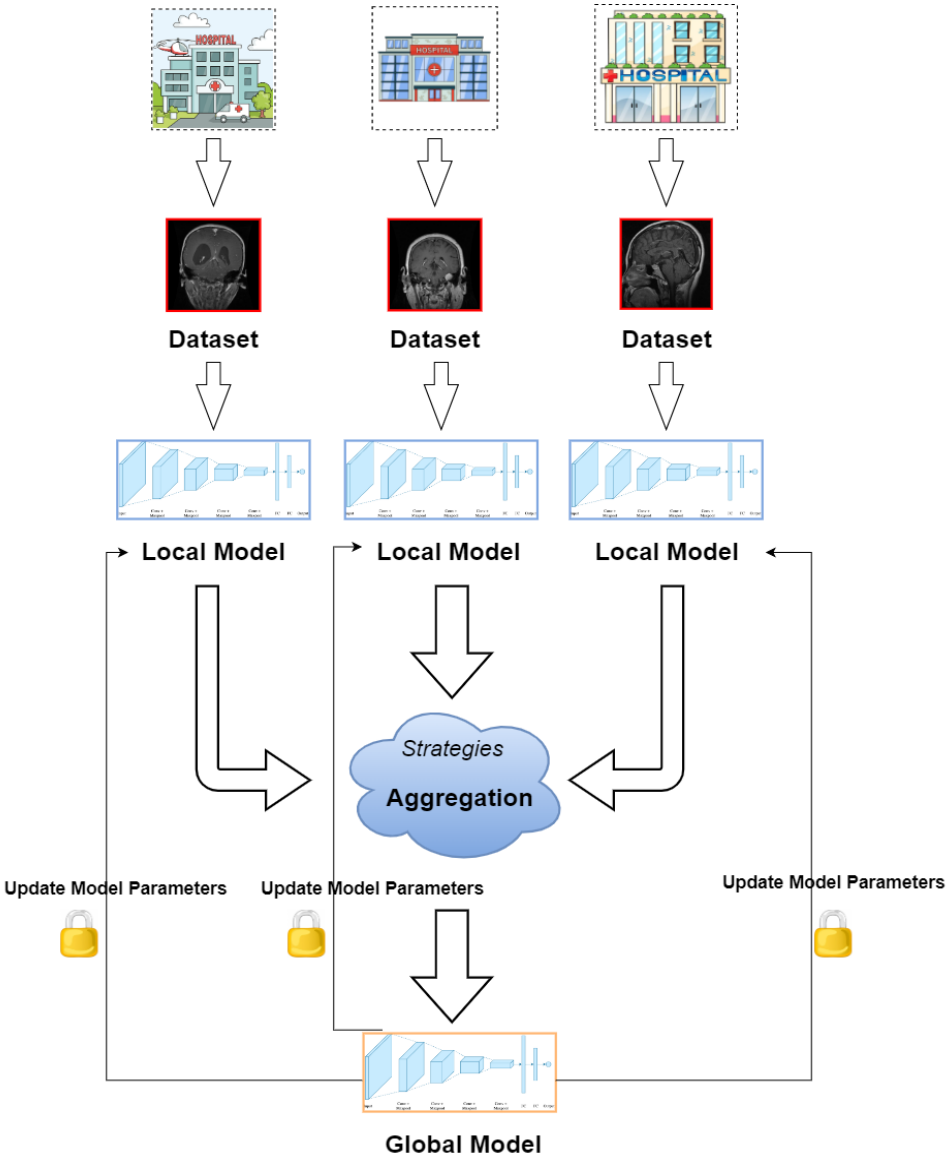
4.4. Eğitim

Federe öğrenme eğitim sürecinin simüle edildiği cihaz bilgileri Tablo 4.2'de verilmiştir.

Tablo 4.2 : Donanımsal bilgiler

İşletim Sistemi	Windows 11
İşlemci	12th Gen Intel(R) Core(TM) i5-12500H 2.50 GHz
RAM	16 GB
GPU	NVIDIA GeForce RTX 3050 Ti

Şekil 4.4'de Federe öğrenme eğitim süreci gösterilmiştir. Üç farklı istemcideki yerel veri setleri ile eğitilen yerel modeller sunucuya gönderilmekte ve sunucuda optimizasyon işlemi gerçekleşmektedir.



Şekil 4.4: Federe öğrenme eğitim süreci

İstemcilerde bulunan yerel modeller oluşturulduktan sonra Federe öğrenme entegrasyonu Flower çerçevesi kullanılarak sağlanmıştır. Yerel modellere ek olarak aynı CNN mimarisinde global model de sunucu tarafında oluşturulmuştur. Entegrasyon sonrası sistemin işlem sırası:

i) Sunucudaki veriler ile global modelin eğitilmesi,

ii) Global model parametrelerinin istemci cihazlara gönderilmesi,

iii) İstemci cihazların yerel verileri ile model eğitimini gerçekleştirmesi,

iv) İstemcilerin yerel modellerini değerlendirmesi,

v) İstemcilerin yerel model ağırlıklarını sunucuya göndermesi,

vi) Sunucuda toplanan yerel model ağırlıklarının optimize edilmesi,

vii) Global modelin tekrar eğitilmesi ve aynı sürecin belirlenen tur sayısı kadar gerçekleştirilmesi şeklindedir.

Çalışmada tur sayısı 10 olarak belirlenmiştir böylece yukarıdaki işlemler 10 tur boyunca devam edecektir. Tur sayısı güncellemelerin paylaşıldığı iletişim tur saıyısını ifade etmektedir. Böylece yerel modeller ve global model 10 kez güncellenmiş olacaktır.

Deneysel çalışmada, dağıtılmış bir veri kümesi üzerinde bir makine öğrenmesi modeli eğitmek için FedAvg, QFedAvg ve FT-FedAvg adlı Federe öğrenme stratejileri kullanılmıştır. Bu stratejiler, TensorFlow tarafından sağlanan Federe Öğrenme kitaplığı kullanılarak uygulanmaktadır. Strateji, birden fazla istemcinin verilerini gizli tutarken paylaşılan bir modeli işbirliği içinde eğitmesine izin vererek Federe öğrenmeyi kolaylaştırmak için tasarlanmıştır. Stratejilerde aşağıdaki parametreler kullanılmıştır.

• fraction_fit: Bu parametre, modeli eğitmek için kullanılacak istemcilerin oranını belirlemek için kullanılmaktadır. Çalışmada değeri 1'e ayarlanmıştır, mevcut tüm istemcilerin eğitim için kullanılacağı anlamına gelmektedir.

• fraction_eval: Bu parametre, modeli değerlendirmek için kullanılacak istemcilerin oranını belirlemek için kullanılmaktadır. Çalışmada değeri 1'e ayarlanmıştır, değerlendirme için mevcut tüm istemcilerin kullanılacağı anlamına gelmektedir.

• min_fit_clients: Bu parametre, eğitim sürecine katılması gereken minimum istemci sayısını belirtmektedir. Çalışmada üç istemci vardır ve eğitim sürecine hepsinin katılması istendiği için değer 3 olarak ayarlanmıştır, eğitim sürecinin yalnızca en az 3 istemci mevcut olduğunda başlayacağı anlamına gelmektedir. Bir istemci bile herhangi bir sebepten dolayı eğitim sürecine katılamazsa süreç işlemeyecektir.

• min_eval_clients: Bu parametre, değerlendirme sürecine katılması gereken minimum istemci sayısını belirtir. Çalışmada üç istemci vardır ve değerlendirme sürecine hepsinin katılması istendiği için değer 3 olarak ayarlanmıştır, değerlendirme sürecinin yalnızca

en az 3 istemci mevcut olduğunda başlayacağı anlamına gelmektedir. Bir istemci bile herhangi bir sebepten dolayı değerlendirme sürecine katılamazsa süreç işlemeyecektir.

• min_available_clients: Bu parametre, eğitim sürecini başlatmak için gereken minimum kullanılabilir istemci sayısını belirtir. Değer 3'e ayarlanmıştır. min_available_clients parametresinin min_fit_clients parametresinden farkı eğitim esnasında oluşabilecek herhangi bir sorundan dolayı eğitimden düşecek istemci olsa dahi sürecin devam etmesi için kullanılmasıdır.

• initialize_parameters: Bu parametre, ilk model parametrelerini belirtmek için kullanılır. fl.common.weights_to_parameters işlevi kullanılarak parametrelere dönüştürülen geçerli model ağırlıklarına ayarlanmaktadır.

Flower çerçevesi kullanılarak her bir stratejinin uygulama adımları aşağıdaki gibidir:

• Model mimarisinin tanımlanması: Federe öğrenme görevi için kullanılacak sinir ağı mimarisini evrişimli sinir ağı modeli oluşturularak tanımlanmıştır. Mimari, TensorFlow ile tanımlanmıştır.

• İstemci güncelleme işlevinin tanımlanması: Eğitim sırasında her istemcide yürütülecek işlev tanımlanmıştır. İşlev, lokal modelin eğitilmesinin ardından mevcut model ağırlıklarının iletilmesini ve güncellenen model ağırlıklarını döndürmektedir.

• Sunucu güncelleme işlevinin tanımlanması: Global model ağırlıklarını güncellemek için sunucuda yürütülecek işlev tanımlanmıştır. İşlev, tüm istemcilerden gelen model ağırlıklarını girdi olarak almakta ve güncellenmiş global model ağırlıklarını döndürmektedir.

• Değerlendirme işlevinin tanımlanması: Modelin performansını değerlendirmek için kullanılacak işlev tanımlanmıştır. İşlev, geçerli global model ağırlıklarını ve değerlendirme verilerini girdi olarak almakta ve değerlendirme metriklerini döndürmektedir. Yerel olarak tutulan test verilerindeki parametrelerin değerlendirilmesinden sorumludur. Ayrıca parametreleri ve yapılandırmayı girdi olarak

alır, modelin ağırlıklarını alınan parametrelere ayarlar ve global model parametrelerini yerel test verileri üzerinde değerlendirir. Değerlendirmenin kayıp, doğruluk, karmaşıklık matrisi ve F1 skorunu döndürür.

• Federe öğrenme stratejisinin tanımlanması: Eğitim sürecini düzenlemek için kullanılacak Federe öğrenme stratejileri tanımlanmıştır. Stratejiler istemcilerin bir alt kümesini rastgele seçmeyi, bu istemcilere mevcut global model ağırlıklarını göndermeyi ve her bir istemciden güncellenmiş ağırlıkları almayı içermektedir. Sunucu daha sonra yeni genel model ağırlıklarını elde etmek için bu ağırlıkların ortalamasını almaktadır.

• Federe öğrenme sürecini çalıştırma: Tanımlanan model mimarisini, istemci güncelleme işlevini, sunucu güncelleme işlevini, değerlendirme işlevini ve Federe öğrenme stratejisini kullanarak, Federe öğrenme sürecini belirtilen sayıda tur boyunca veya yakınsama sağlanana kadar çalıştırılır.

Verilerin cihazlar arasında nasıl dağıtıldığı, Federe Öğrenimin uygulanması için önemlidir. Veri dağılımları genellikle cihazlar arasında eşit dağıldığı ve aynı özelliklere sahip olduğu için çalışmada yapı olarak yatay öğrenme kullanılmıştır. Yatay öğrenme kullanılmasının nedeni, tüm istemcilerin her sınıfa sahip olmasıdır.

Çalışmanın başka bir yaklaşımında da FedAvg ile birlikte Diferansiyel Gizlilik kullanılmıştır. Diferansiyel gizlilik ilkesine göre, çalışmada model bilgilerini bir istemciden sunucuya gönderirken model parametrelerine gürültü eklenmiştir. Diferansiyel gizlilik, belirlenen dönem sayısı, eğitim örnekleri, toplu iş boyutu ve gürültü çarpanı dahil olmak üzere belirli parametreler ile birlikte hesaplanır. Ayriyeten bu hiperparametrelere ek olarak hesapta gizliliği sağlamak için eğitim sürecine eklenen gürültü seviyesini belirleyen bir parametrede fonksiyon içerisinde eklenmektedir. Kalan iki istemciye ise herhangi bir diferansiyel gizlilik uygulanmamıştır. Çalışmanın amacı, bir istemciye diferansiyel gizlilik eklendikten sonra hem diferansiyel gizlilik eklenen istemcinin hem de sunucu modelindeki performans değişimini gözlemlemektir. Diferansiyel gizlilik, model parametrelerine belirli bir oranda gürültü eklemektedir. Laplacian veya Gauss tipi gürültü kullanmak mümkündür. Diferansiyel gizlilik güvenlik

sağlarken performansı da düşürebilmektedir. Eklenen gürültü seviyesi arttıkça performans düşerken güvenlik artmaktadır.

4.5. Deneysel Sonuçlar

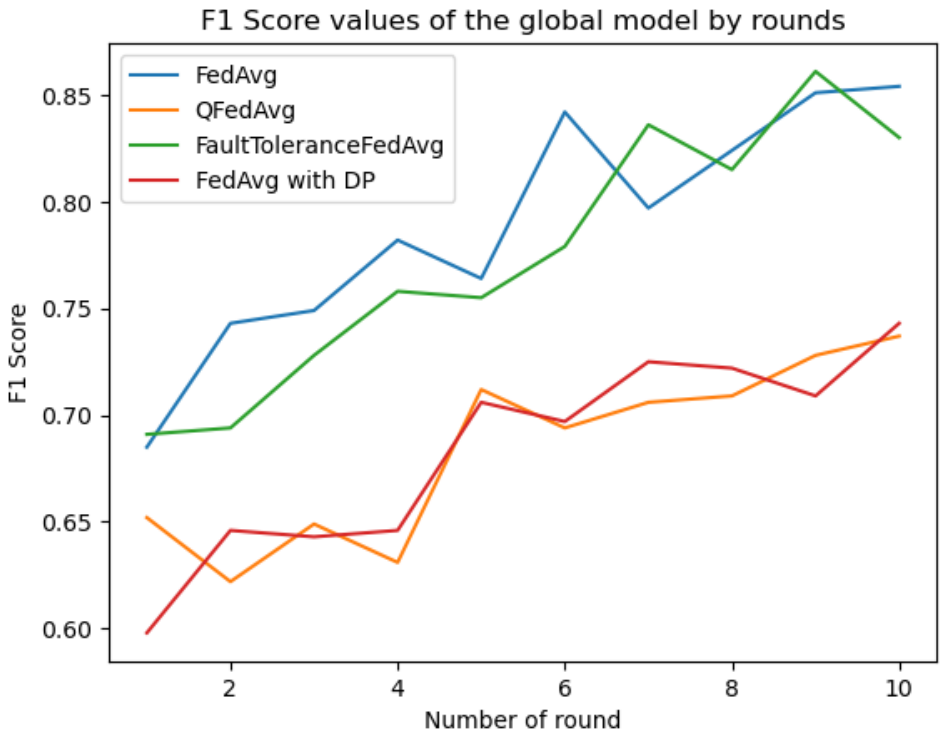
Optimize edicilerin performansı ve her turdan sonra modellerin ilerlemesi, F1 skoru metriğine göre karşılaştırıldı. Gerçek Pozitif (TP), modelin tümör numunelerini tümör olarak sınıflandırdığını gösterirken, Gerçek Negatif (TN) modelin tümör olmayan numuneleri tümör olmayan olarak sınıflandırdığını gösterir, Yanlış Pozitif (FP), modelin yanlış bir şekilde tümör olmayan numuneleri sınıflandırdığını gösterir. Yanlış Negatif (FN), modelin tümör örneklerini yanlış bir şekilde tümör olmayan olarak sınıflandırdığını gösterir. Bu performans metrikleri için formüller aşağıda verilmiştir.

$$Precision = \frac{TP}{TP + FP} \tag{4.1}$$

$$Recall = \frac{TP}{TP+FN} \tag{4.2}$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4.3}$$

Şekil 4.5'de sunucudaki global modelin tur sayısına göre F1 skor değerlerinin gelişimi gösterilmiştir.



Şekil 4.5: Tur sayısına göre global modelin F1-Skor gelişimi

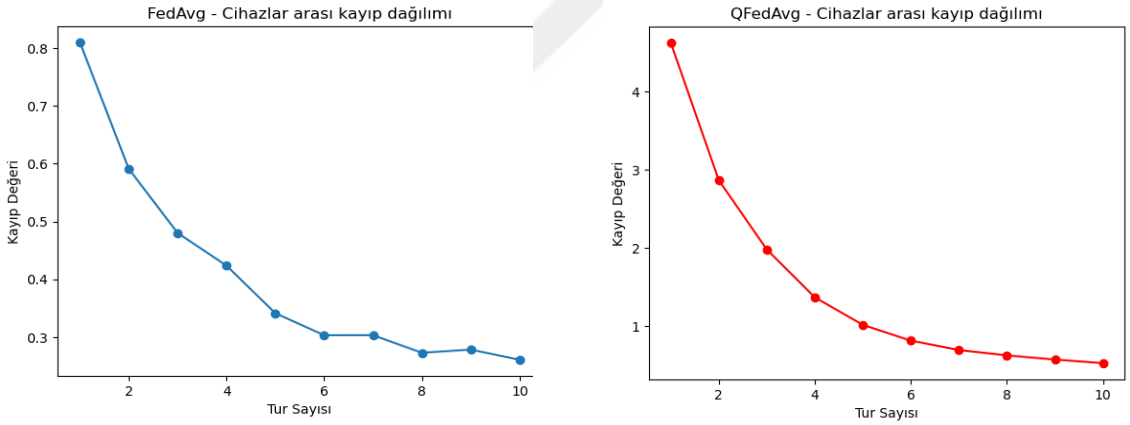
Modeller her turda 5 dönem eğitildi. Dört farklı komplikasyonda da lokal modeller ile birlikte global modeller de gelişim sağlamıştır. 10 turdan sonra, FedAvg en başarılı strateji gibi görünmektedir. Bu açıklanabilir bir sonuçtur çünkü sınıfların istemciler arasındaki dağılımı yaklaşık olarak eşittir. Bu nedenle FedAvg, bir ortalama alma stratejisi olduğu için başarılı olmuştur.

Tablo 4.3 : Stratejilere göre global modelin sunucudaki F1-Skoru değişimi

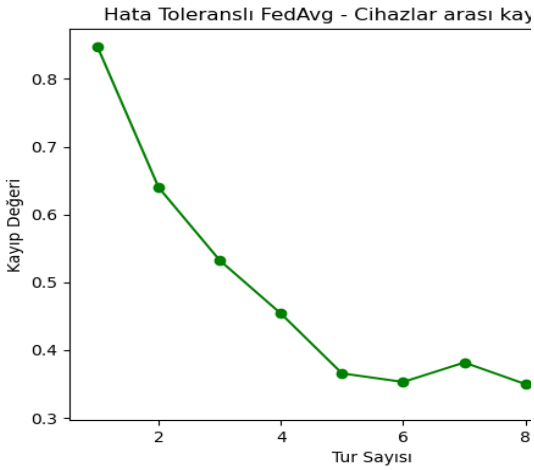
Tur Sayısı	**FedAvg**	**QFedAvg**	**Hata Toleranslı FedAvg**	**Diferansiyel Gizlilik ile FedAvg**
0	0.607	0.607	0.565	0.622
1	0.686	0.653	0.692	0.598
2	0.743	0.622	0.695	0.647
3	0.749	0.650	0.728	0.644
4	0.782	0.631	0.758	0.647
5	0.764	0.713	0.755	0.707
6	0.843	0.695	0.779	0.698
7	0.798	0.707	0.837	0.725
8	0.825	0.710	0.816	0.722
9	0.852	0.728	0.861	0.710
10	0.855	0.737	0.831	0.743

Tablo 4.3 incelendiğinde düşük F1 skorundan başlayan global modeller her bir strateji için zamanla gelişim göstermiştir. Tablodaki ilk satır sürecin başlamasını sağlayan küresel modelin herhangi bir istemciden model ağırlığı almadan kendi modelini eğitmesinin sonucundaki değerlendirmeyi göstermektedir. Her bir tur sonunda üç istemciden gelen yerel model ağırlıklarına göre global modeller gelişim göstermiştir. Görülmektedir ki 1. istemciye eklenen diferansiyel gizlilik global modelin öğrenme durumunu etkilemiştir. Global modelin gelişimi her tur sonunda pozitif anlamda olmamıştır bunun sebebi herhangi bir istemcinin o turda iyi öğrenim gerçekleştirememesinden kaynaklı olabilmektedir. Fakat nihai olarak her strateji durumunda da F1 skoru ilk tura göre son turda pozitif anlamda gelişim göstermiştir.

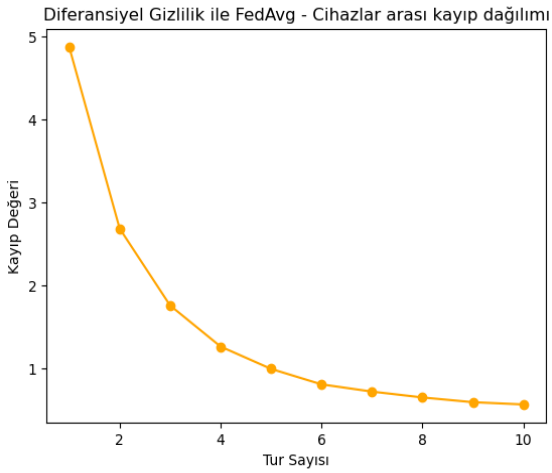
Şekil 4.6'daki grafikler stratejilere göre üç istemcinin ortak kayıp değerinin tur sayısı boyunca gelişimini göstermektedir. FedAvg ve FT - FedAvg, 10 tur sonunda kayıplarını en çok azaltan stratejilerdir.



(a) FedAvg (b) QFedAvg



(c) FT - FedAvg



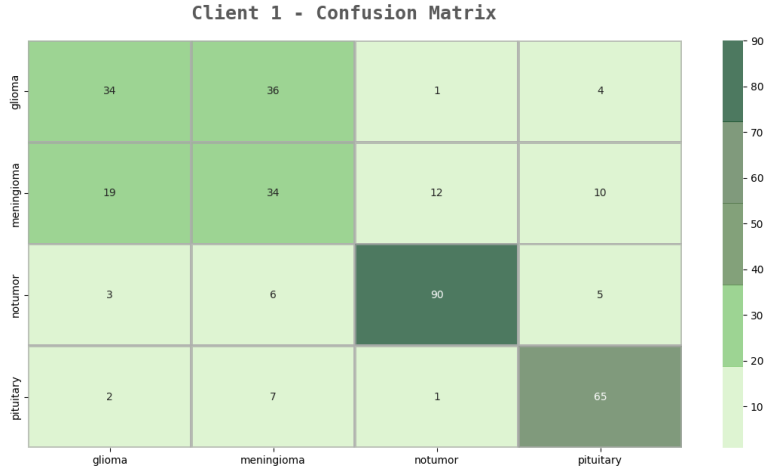
(d) Diferansiyel Gizlilik ile FedAvg

Şekil 4.6: İstemciler arasındaki ortak kayıp değerinin tur sayısı boyunca gelişimi (a) FedAvg (b) QFedAvg (c) FT - FedAvg (d) Diferansiyel Gizlilik ile FedAvg

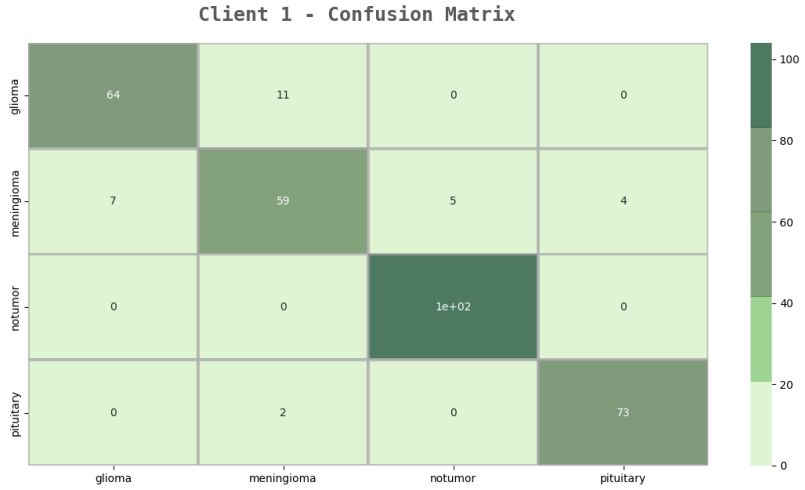
Federe öğrenme sürecinin 10 tur boyunca sürmesinin Şekil 4.6'daki grafikler incelendiğinde yeterli olduğu görülmektedir. Grafikler incelendiğinde istemci modellerinin bütün stratejiler için 6. turdan sonra gelişim göstermesinin oldukça azaldığı gözlemlenmektedir.

4.5.1. FedAvg strajesine göre istemci ve sunucu modelleri gelişimi

Şekil 4.7'de istemci 1, Şekil 4.8'de istemci 2, Şekil 4.9'da istemci 3 ve Şekil 4.10'da sunucu için modellerin gelişimini gösteren karmaşıklık matrisleri verilmiştir.



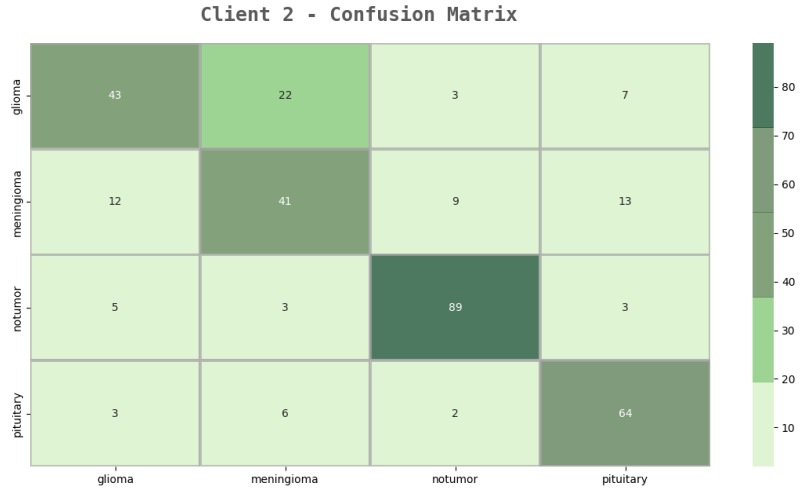
(a) Birinci tur



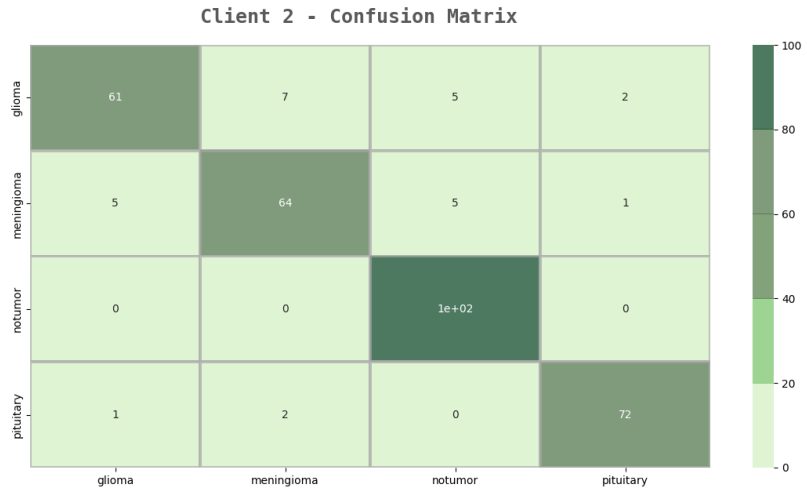
(b) Onuncu tur

Şekil 4.7: İstemci 1 karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

İstemci 1 için birinci turda glioma sınıfının doğru pozitif değeri 34 iken son tur olan onuncu turdaki doğru pozitif değeri 64'dür. Meningioma sınıfının birinci turdaki doğru pozitif değeri 34 iken onuncu turdaki doğru pozitif değeri ise 59'dur. Pituitary sınıfı için doğru pozitif değeri 65 iken 73 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 90 iken onuncu tur sonunda 102 olmuştur. Pituitary ve tumör yok sınıfını diğer sınıflara göre istemci 1 modeli ilk turdan itibaren iyi öğrenmektedir.



(a) Birinci tur

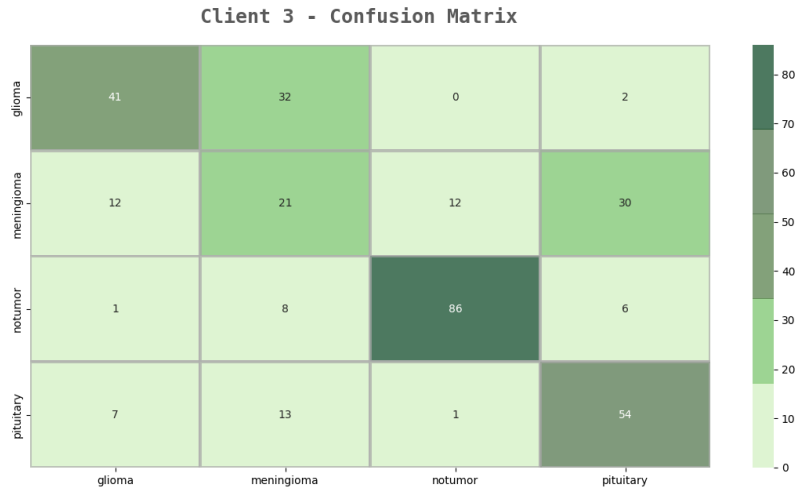


(b) Onuncu tur

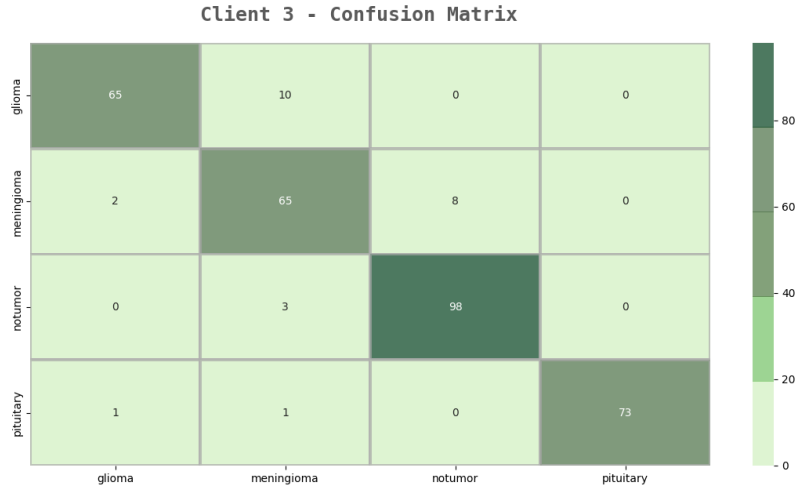
Şekil 4.8: İstemci 2 karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

İstemci 2 için birinci turda glioma sınıfının doğru pozitif değeri 43 iken son tur olan onuncu turdaki doğru pozitif değeri 61'dir. Meningioma sınıfının birinci turdaki doğru pozitif değeri 43 iken onuncu turdaki doğru pozitif değeri ise 64'dür. Pituitary sınıfı için doğru pozitif değeri 64 iken 72 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 89 iken onuncu tur sonunda 102 olmuştur. Pituitary ve tumör yok sınıfını diğer sınıflara göre istemci 2 modeli ilk turdan itibaren iyi öğrenmektedir. İstemci 2

modeli, istemci 1 modeline göre ilk turlarda glioma ve meningioma sınıflarını daha iyi tahminlemiştir.



(a) Birinci tur

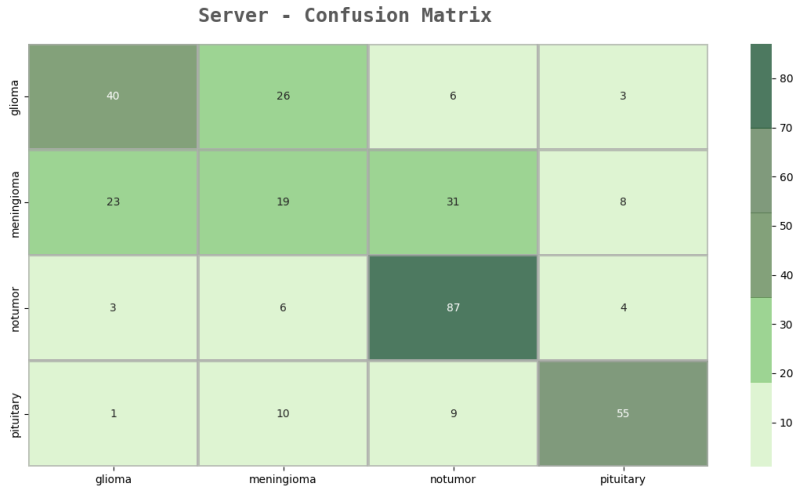


(b) Onuncu tur

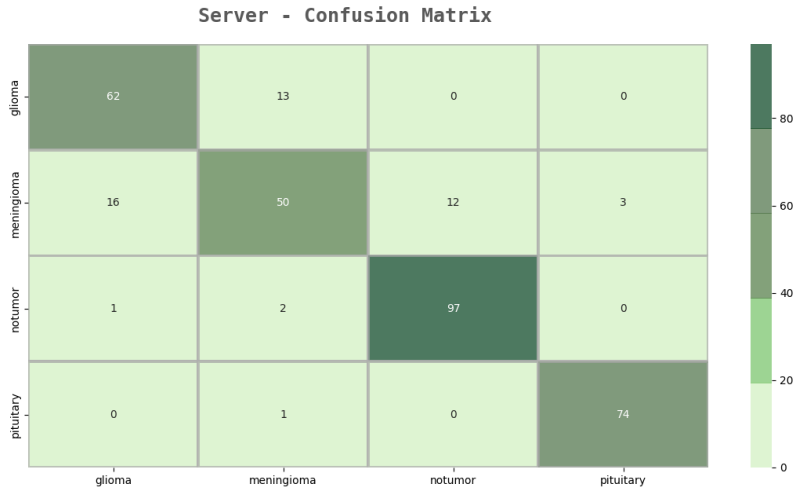
Şekil 4.9: İstemci 3 karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

İstemci 3 için birinci turda glioma sınıfının doğru pozitif değeri 41 iken son tur olan onuncu turdaki doğru pozitif değeri 65'dür. Meningioma sınıfının birinci turdaki doğru pozitif değeri 21 iken onuncu turdaki doğru pozitif değeri ise 65'dir. Pituitary sınıfı için doğru pozitif değeri 54 iken 73 olmuştur. Tümör yok sınıfı için birinci turda doğru

pozitif değeri 86 iken onuncu tur sonunda 98 olmuştur. İstemci 3 modeli, diğer istemci modellerine göre ilk turda meningioma sınıfını pituitary sınıfı ile fazla karıştırmıştır. Son turda bütün istemciler performanslarını benzer bir şekilde arttırmıştır.



(a) Birinci tur



(b) Onuncu tur

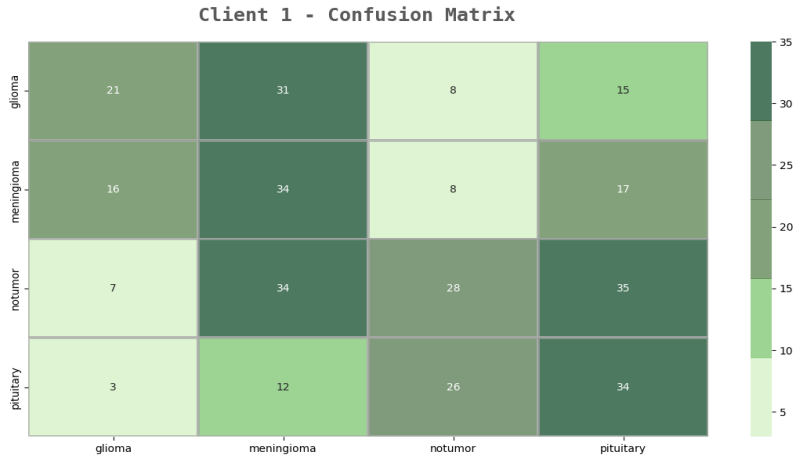
Şekil 4.10: Sunucu karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

Sunucuda bulunan global model için birinci turda glioma sınıfının doğru pozitif değeri 40 iken son tur olan onuncu turdaki doğru pozitif değeri 62'dir. Meningioma sınıfının birinci turdaki doğru pozitif değeri 19 iken onuncu turdaki doğru pozitif değeri ise

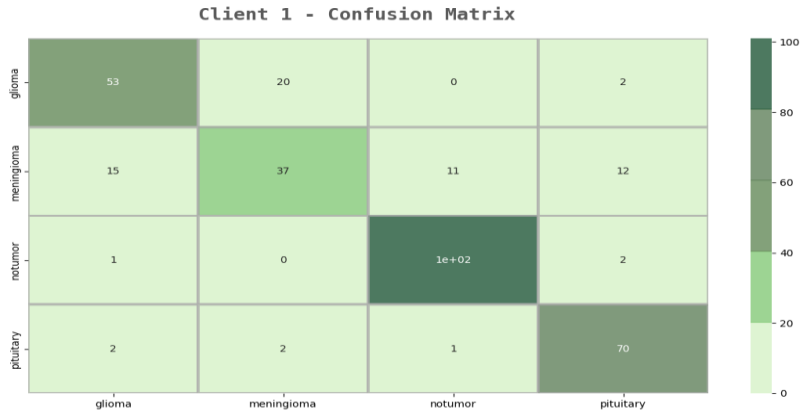
50’dir. Pituitary sınıfı için doğru pozitif değeri 55 iken 74 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 87 iken onuncu tur sonunda 97 olmuştur. Global model özellikle ilk turda meningioma sınıfını iyi genelleyemezken son turda meningioma sınıfı tahminlerini oldukça geliştirdiği görülmektedir.

4.5.2. QFedAvg strajesine göre istemci ve sunucu modelleri gelişimi

Şekil 4.11’de istemci 1, Şekil 4.12’de istemci 2, Şekil 4.13’de istemci 3 ve Şekil 4.14’de sunucu için modellerin gelişimini gösteren karmaşıklık matrisleri verilmiştir.



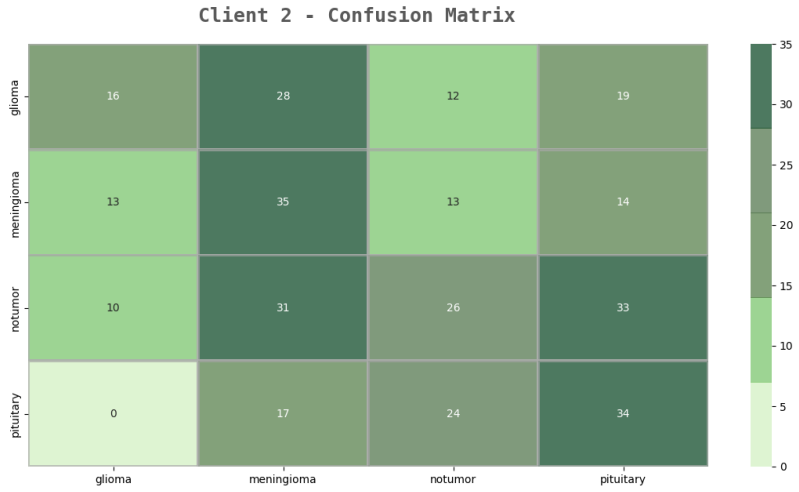
(a) Birinci tur



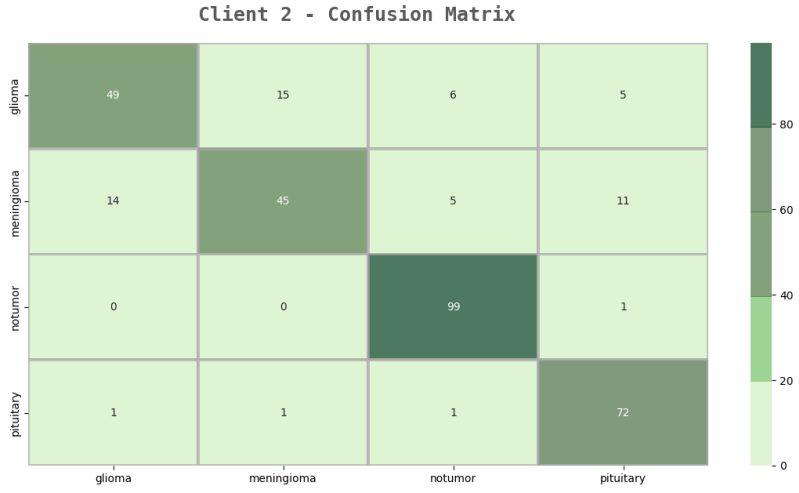
(b) Onuncu tur

Şekil 4.11: İstemci 1 karmaşıklık matrisleri (a) Birinci tur (b) Onuncu

İstemci 1 için birinci turda glioma sınıfının doğru pozitif değeri 21 iken son tur olan onuncu turdaki doğru pozitif değeri 53'dür. Meningioma sınıfının birinci turdaki doğru pozitif değeri 34 iken onuncu turdaki doğru pozitif değeri ise 37'dir. Pituitary sınıfı için doğru pozitif değeri 34 iken 70 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 28 iken onuncu tur sonunda 102 olmuştur. QFedAvg stratejisinde istemci 1 modeli ilk turda bütün sınıflar için iyi bir genellemeye hakim değildir fakat son turda özellikle tumör yok ve pituitary sınıflarını oldukça iyi genellemiştir.



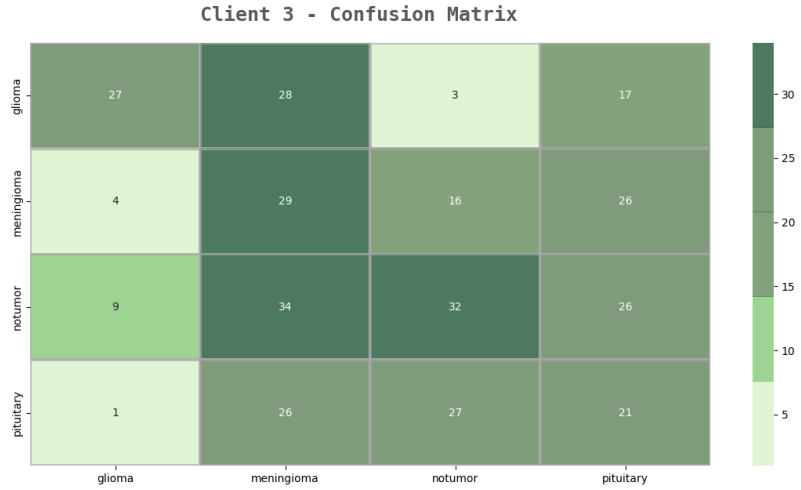
(a) Birinci tur



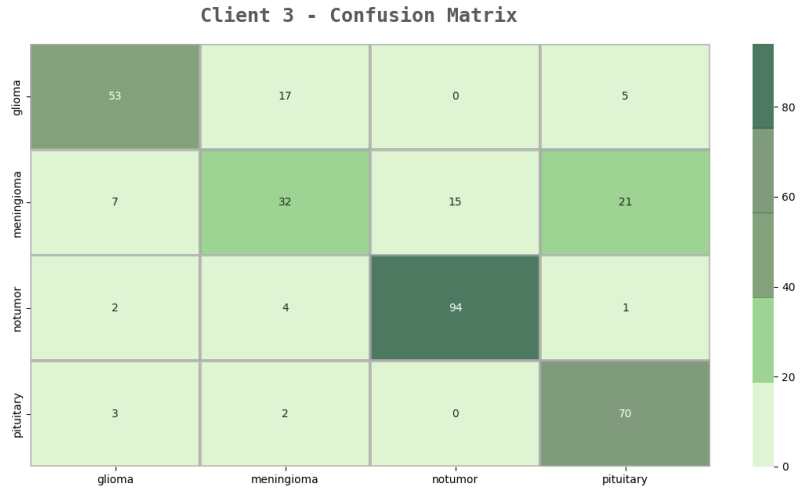
(b) Onuncu tur

Şekil 4.12: İstemci 2 karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

İstemci 2 için birinci turda glioma sınıfının doğru pozitif değeri 16 iken son tur olan onuncu turdaki doğru pozitif değeri 49'dur. Meningioma sınıfının birinci turdaki doğru pozitif değeri 35 iken onuncu turdaki doğru pozitif değeri ise 45'dir. Pituitary sınıfı için doğru pozitif değeri 34 iken 72 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 26 iken onuncu tur sonunda 99 olmuştur. QFedAvg stratejisinde istemci 2 modeli, istemci 1 ile benzer bir gelişim göstermiştir.



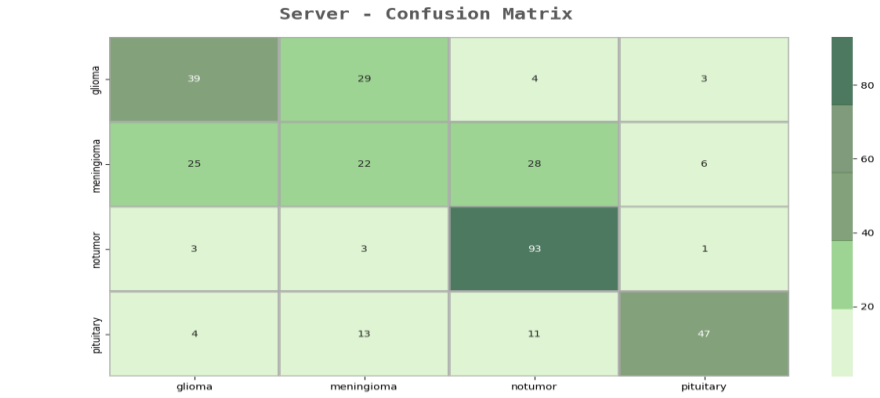
(a) Birinci tur



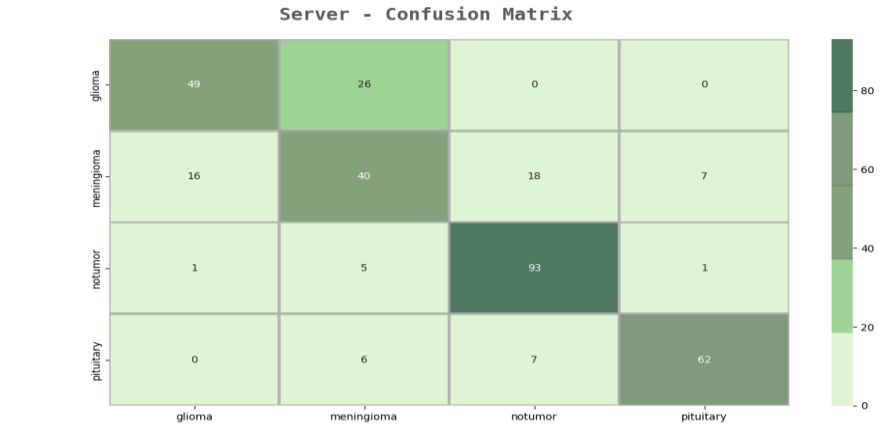
(c) Onuncu tur

Şekil 4.13: İstemci 3 karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

İstemci 3 için birinci turda glioma sınıfının doğru pozitif değeri 27 iken son tur olan onuncu turdaki doğru pozitif değeri 53'dür. Meningioma sınıfının birinci turdaki doğru pozitif değeri 29 iken onuncu turdaki doğru pozitif değeri ise 32'dir. Pituitary sınıfı için doğru pozitif değeri 21 iken 70 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 34 iken onuncu tur sonunda 94 olmuştur. İstemci 3 incelendiğinde diğer iki istemci gibi meningioma sınıfını iyi genelleyememiştir. FedAvg strateji ile geliştirilen istemci modellerinden farkları ise tümör yok ve pituitary sınıflarını ilk turda iyi tahminleyemeyip son turda oldukça geliştirmiş olmalarıdır.



(a) Birinci tur



(b) Onuncu tur

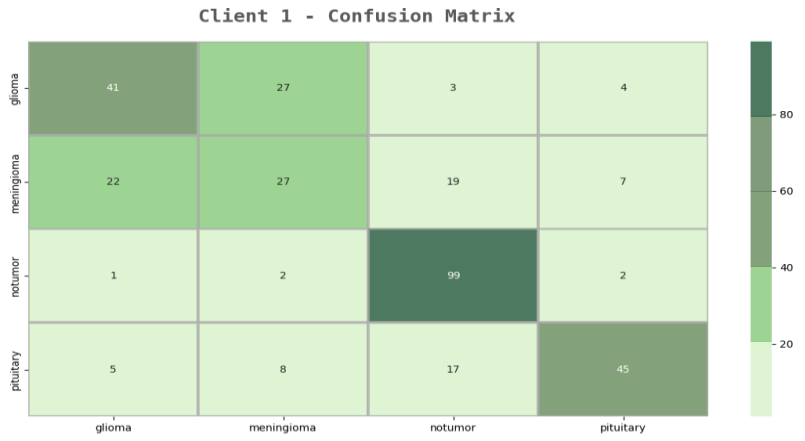
Şekil 4.14: Sunucu karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

Sunucuda bulunan global model için birinci turda glioma sınıfının doğru pozitif değeri 39 iken son tur olan onuncu turdaki doğru pozitif değeri 49'dur. Meningioma sınıfının

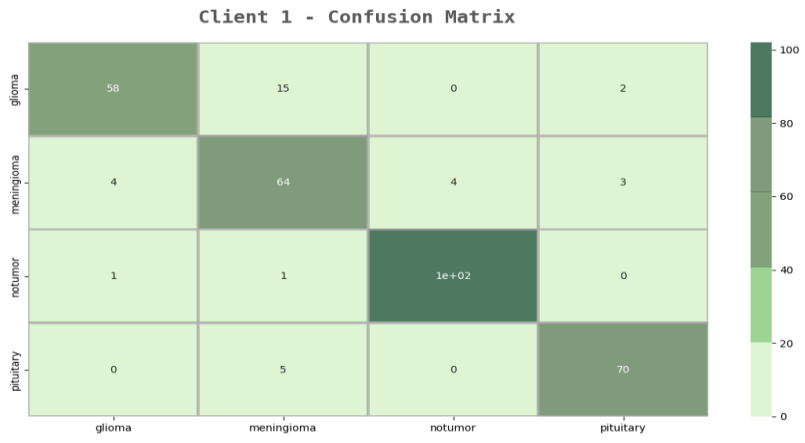
birinci turdaki doğru pozitif değeri 22 iken onuncu turdaki doğru pozitif değeri ise 40'dır. Pituitary sınıfı için doğru pozitif değeri 47 iken 62 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 93 iken onuncu tur sonunda da 93 olmuştur. Global model 10 tur sonunda QFedAvg stratejisinde en çok meningioma sınıfının başarımını geliştirmiştir.

4.5.3. Hata Toleranslı FedAvg strajesine göre istemci ve sunucu modelleri gelişimi

Şekil 4.15'de istemci 1, Şekil 4.16'da istemci 2, Şekil 4.17'de istemci 3 ve Şekil 4.18'de sunucu için modellerin gelişimini gösteren karmaşıklık matrisleri verilmiştir.



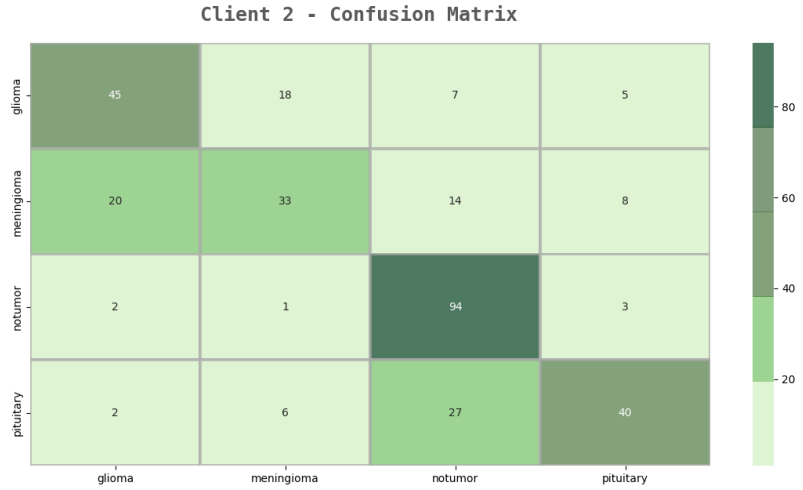
(a) Birinci tur



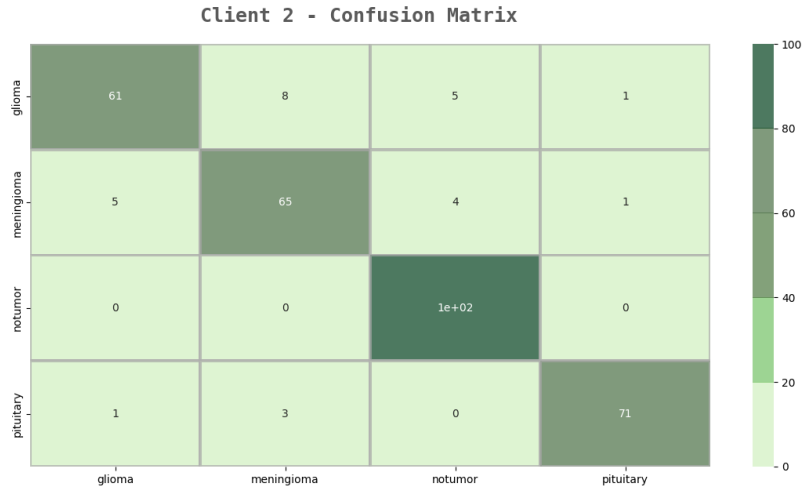
(b) Onuncu tur

Şekil 4.15: İstemci 1 karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

İstemci 1 için birinci turda glioma sınıfının doğru pozitif değeri 41 iken son tur olan onuncu turdaki doğru pozitif değeri 58'dir. Meningioma sınıfının birinci turdaki doğru pozitif değeri 27 iken onuncu turdaki doğru pozitif değeri ise 64'dür. Pituitary sınıfı için doğru pozitif değeri 45 iken 70 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 99 iken onuncu tur sonunda 102 olmuştur. En çok gelişimi on tur sonunda meningioma sınıfı göstermiştir.



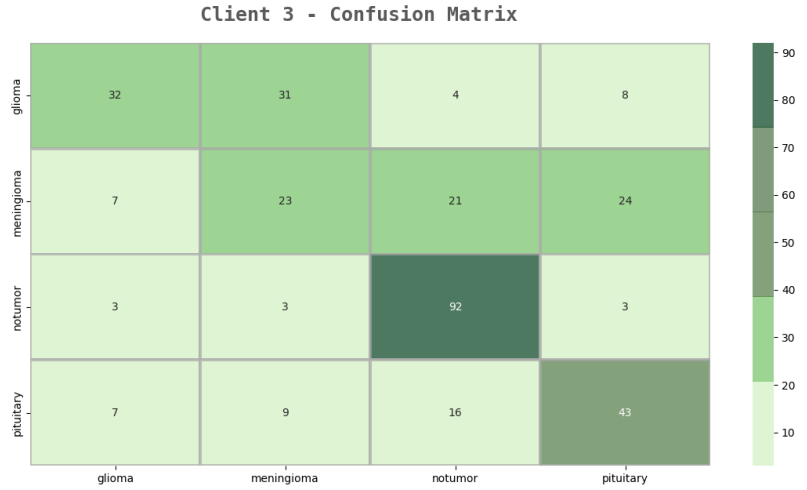
(a) Birinci tur



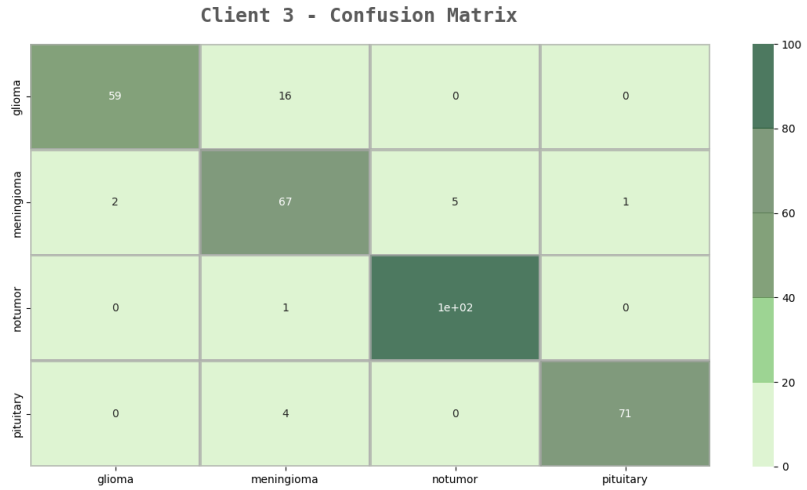
(b) Onuncu tur

Şekil 4.16: İstemci 2 karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

İstemci 2 için birinci turda glioma sınıfının doğru pozitif değeri 45 iken son tur olan onuncu turdaki doğru pozitif değeri 61'dir. Meningioma sınıfının birinci turdaki doğru pozitif değeri 33 iken onuncu turdaki doğru pozitif değeri ise 65'dir. Pituitary sınıfı için doğru pozitif değeri 40 iken 71 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 94 iken onuncu tur sonunda 102 olmuştur. İstemci 2 modeli için de en çok gelişimi on tur sonunda meningioma sınıfı göstermiştir.



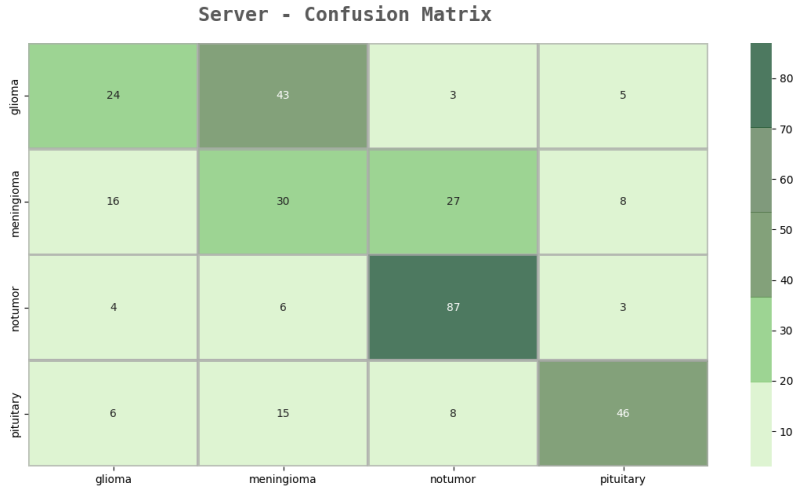
(a) Birinci tur



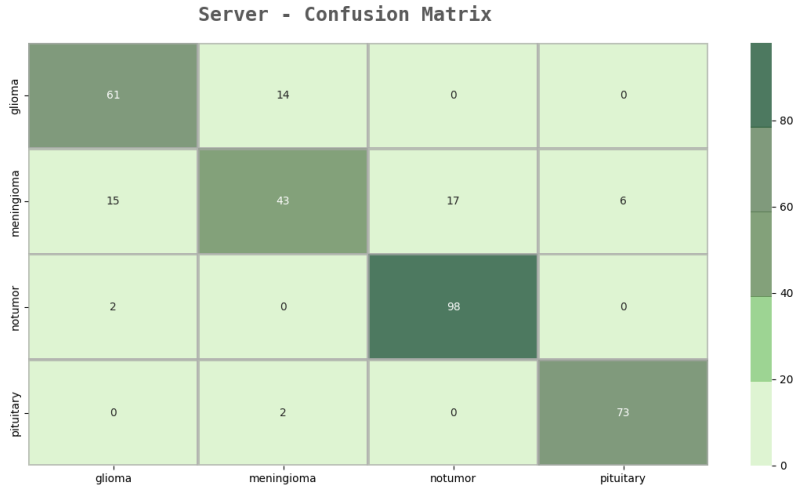
(b) Onuncu tur

Şekil 4.17: İstemci 3 karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

İstemci 3 için birinci turda glioma sınıfının doğru pozitif değeri 32 iken son tur olan onuncu turdaki doğru pozitif değeri 59'dur. Meningioma sınıfının birinci turdaki doğru pozitif değeri 23 iken onuncu turdaki doğru pozitif değeri ise 67'dür. Pituitary sınıfı için doğru pozitif değeri 43 iken 71 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 92 iken onuncu tur sonunda 102 olmuştur.



(a) Birinci tur



(b) Onuncu tur

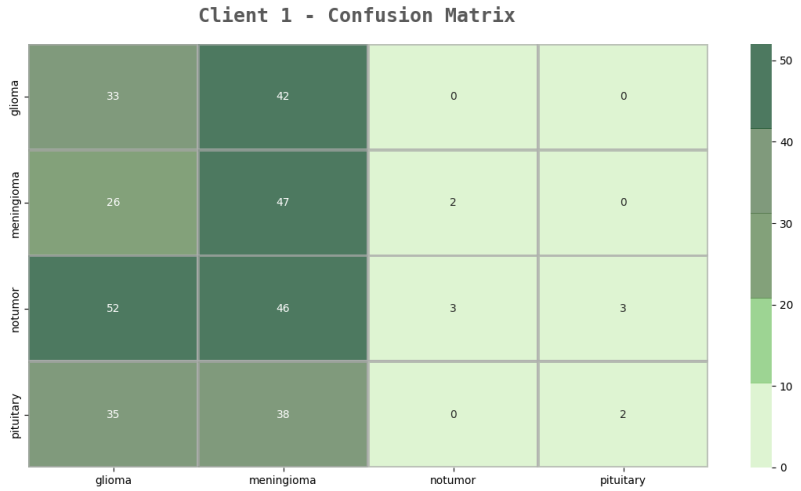
Şekil 4.18: Sunucu karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

Sunucuda bulunan global model için birinci turda glioma sınıfının doğru pozitif değeri 24 iken son tur olan onuncu turdaki doğru pozitif değeri 61'dir. Meningioma sınıfının birinci turdaki doğru pozitif değeri 30 iken onuncu turdaki doğru pozitif değeri ise 43'dür. Pituitary sınıfı için doğru pozitif değeri 46 iken 73 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 87 iken onuncu tur sonunda 98 olmuştur.

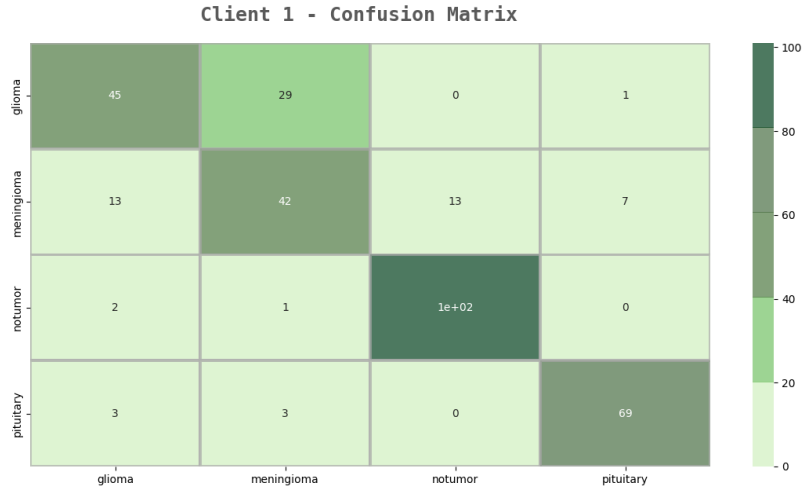
Karmaşıklık matrisleri incelendiğinde FT - FedAvg stratejisinin, standart FedAvg stratejisine yakın performans gösterdiği görülmektedir.

4.5.4. Diferansiyel gizlilik ile birlikte FedAvg strajesine göre istemci ve sunucu modelleri gelişimi

Şekil 4.19'da istemci 1, Şekil 4.20'de istemci 2, Şekil 4.21'de istemci 3 ve Şekil 4.22'de sunucu için modellerin gelişimini gösteren karmaşıklık matrisleri verilmiştir. Bu yöntem de diferansiyel gizlilik sadece istemci 1'in model ağırlıklarına eklenmiştir. Gürültü eklenen istemci 1 model ağırlıklarının, kendi yerel modeli ve diğer istemcilerin yerel modelleriyle birlikte sunucudaki global modeli nasıl etkileyeceği gözlemlenmiştir.



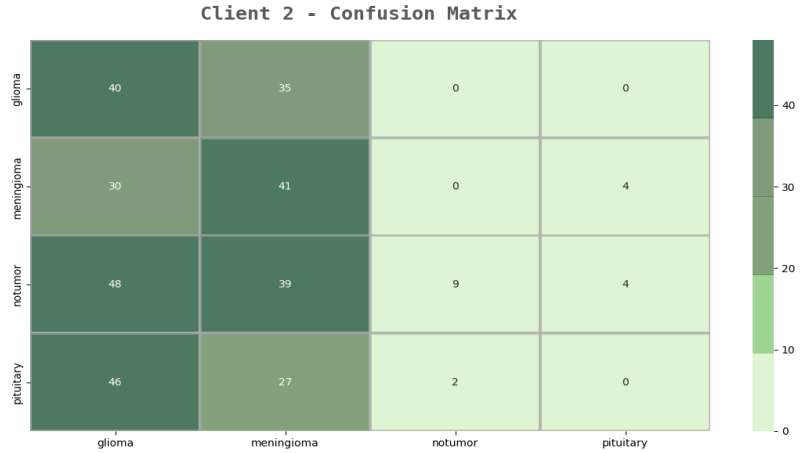
(a) Birinci tur



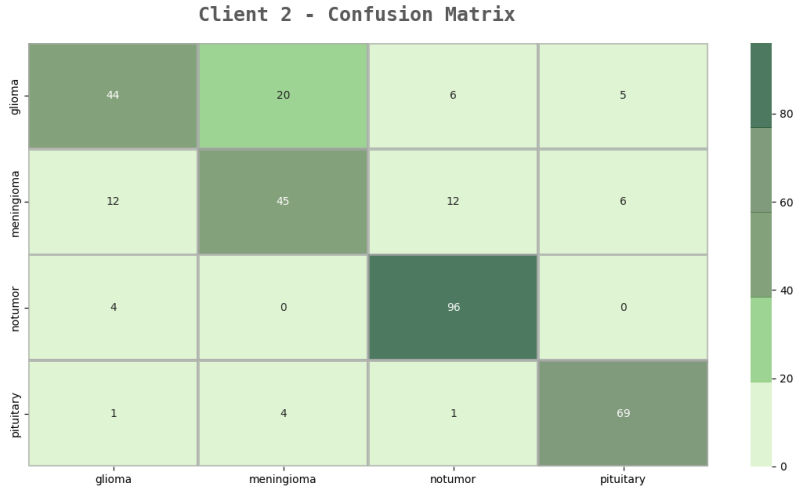
(b) Onuncu tur

Şekil 4.19: İstemci 1 karmaşıklık matrisleri (a) Birinci tur (b) Onuncu

İstemci 1 için birinci turda glioma sınıfının doğru pozitif değeri 33 iken son tur olan onuncu turdaki doğru pozitif değeri 45'dir. Meningioma sınıfının birinci turdaki doğru pozitif değeri 47 iken onuncu turdaki doğru pozitif değeri ise 42'dir. Pituitary sınıfı için doğru pozitif değeri 2 iken 69 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 3 iken onuncu tur sonunda 102 olmuştur. DPFedAvg beklenildiği gibi diğer stratejilere göre daha düşük performans sergilemektedir çünkü gürültülü model ağırlıkları optimize edilmektedir. Pituitary ve tümör yok sınıflarının gelişimi göze çarparken meningioma sınıfının onuncu turdaki performansı ilk tura göre daha düşüktür.



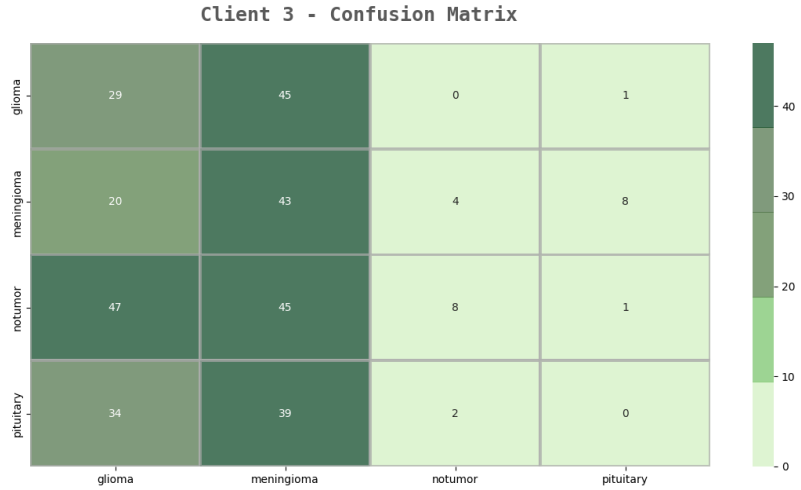
(a) Birinci tur



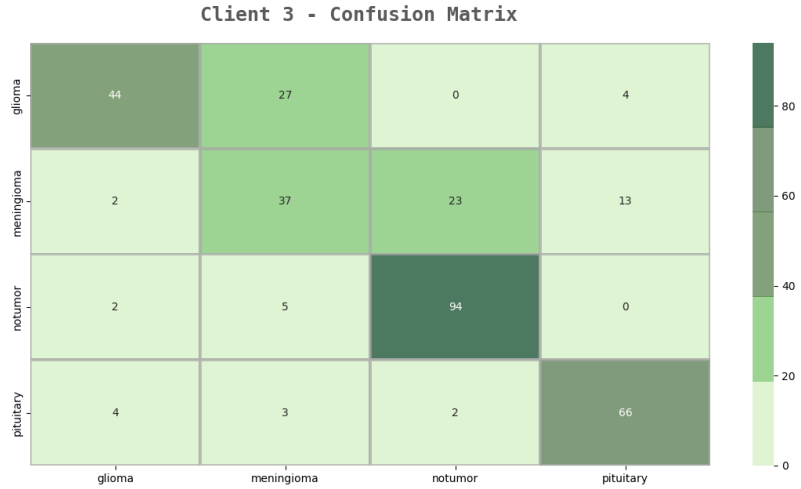
(b) Onuncu tur

Şekil 4.20: İstemci 2 karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

İstemci 2 için birinci turda glioma sınıfının doğru pozitif değeri 40 iken son tur olan onuncu turdaki doğru pozitif değeri 44'dür. Meningioma sınıfının birinci turdaki doğru pozitif değeri 41 iken onuncu turdaki doğru pozitif değeri ise 45'dir. Pituitary sınıfı için doğru pozitif değeri 0 iken 69 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 9 iken onuncu tur sonunda 96 olmuştur. DPFedAvg stratejisi en fazla pituitary ve tümör yok sınıflarını etkilemiştir. Öyle ki ilk turda istemci 2 modeli pituitary sınıfına dair hiç doğru tahmin yapmamıştır.



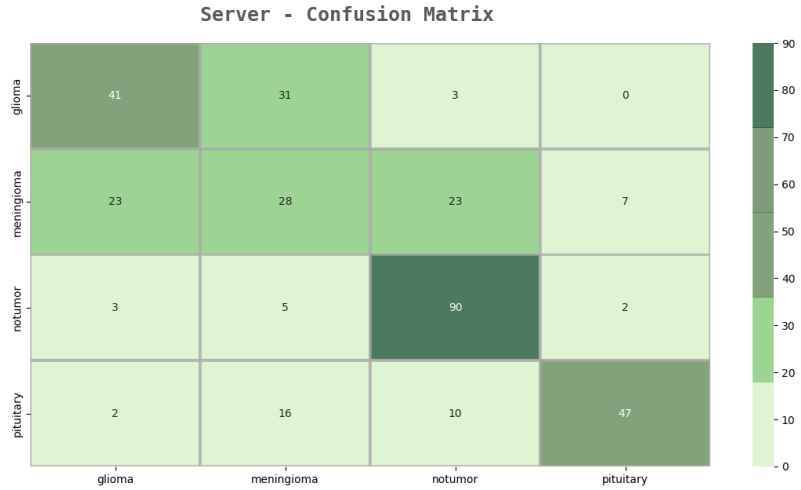
(a) Birinci tur



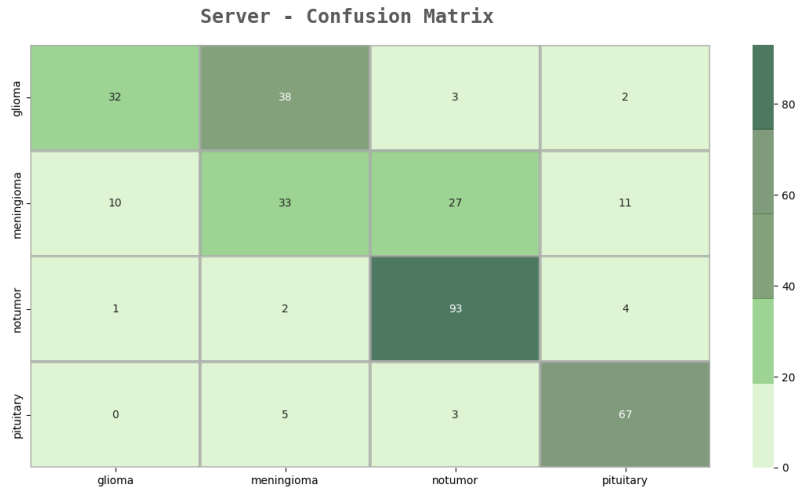
(b) Onuncu tur

Şekil 4.21: İstemci 3 karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

İstemci 3 için birinci turda glioma sınıfının doğru pozitif değeri 29 iken son tur olan onuncu turdaki doğru pozitif değeri 44'dür. Meningioma sınıfının birinci turdaki doğru pozitif değeri 43 iken onuncu turdaki doğru pozitif değeri ise 37'dir. Pituitary sınıfı için doğru pozitif değeri 0 iken 66 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 8 iken onuncu tur sonunda 94 olmuştur.



(a) Birinci tur



(b) Onuncu tur

Şekil 4.22: Sunucu karmaşıklık matrisleri (a) Birinci tur (b) Onuncu tur

Sunucuda bulunan global model için birinci turda glioma sınıfının doğru pozitif değeri 41 iken son tur olan onuncu turdaki doğru pozitif değeri 32'dir. Meningioma sınıfının birinci turdaki doğru pozitif değeri 28 iken onuncu turdaki doğru pozitif değeri ise 33'dür. Pituitary sınıfı için doğru pozitif değeri 47 iken 67 olmuştur. Tümör yok sınıfı için birinci turda doğru pozitif değeri 90 iken onuncu tur sonunda 93 olmuştur.

İstemci 1 yerel model ağırlıklarına diferansiyel gizlilik ile gürültü eklenmesinin özellikle sunucuda bulunan global modelin gelişimini beklendiği üzere olumsuz

etkilediği gözlemlenmiştir. Bu etken eklenen gürültü oranına göre ve eldeki probleme göre değişebilmektedir.

Karmaşıklık matrisleri incelendiğinde en başarılı stratejinin FedAvg olduğu görülmektedir ancak FT - FedAvg'da tercih edilebilir. Gerçek hayat senaryolarında FT - FedAvg kullanmak FedAvg'dan daha faydalı olabilmektedir çünkü FT – FedAvg hataları tolere edebilmektedir. Yakın performanslara sahip iki strateji arasındaki seçim çalışmanın senaryosuna ve donanımsal özelliklere göre seçilebilir.

4.5.5. Federe öğrenme stratejileri ile geleneksel modellerin karşılaştırılması

Çalışmada kullanılan stratejilere ek olarak geleneksel olarak CNN modeli, VGG16 ve DenseNet modeli 5 epoch eğitilmiştir. CNN modelinin mimarisi lokal ve sunucu modelinin mimarisi ile aynı yapıdadır. VGG16 modeli, 16 katmanlı derin bir sinir ağıdır. Evrişim ve tam bağlantılı katmanlardan oluşmaktadır. Evrişim katmanlarından oluşan özelliği çıkarma bölümü ve tam bağlantılı katmanlarla birleştirilen sınıflandırma bölümünü içermektedir. DenseNet modeli, yoğun bağlantıları olan "Dense bloklar" olarak adlandırılan bir yapıyı temel almaktadır. Her katman, önceki katmanların çıkışları ile birleştirilerek öğrenme ve bilgi akışı kolaylaştırılmaktadır. Bu yoğun bağlantılar, daha az parametreyle daha derin ve etkili sinir ağı modelleri sağlamaktadır. Bu modeller federe öğrenme yapısına göre bölümlemeden tüm veri kümesinden eğitim kümesi, validasyon kümesi ve test kümesi oluşturulmuştur. Eğitim kümesi 5141 örnek, validasyon kümesi 571 örnek ve test kümesi ise 1311 örnekten oluşmaktadır. Tablo 4.x'de F1 skoruna göre karşılaştırma tablosu verilmiştir.

Tablo 4.4 : Federe öğrenme stratejileri ve geleneksel modellerin F1-Skor değerleri

	FedAvg	QFedAvg	FT-FedAvg	DPFedAvg	CNN	VGG16	DenseNet
F1 Skor	0.855	0.737	0.831	0.743	0.835	0.875	0.853

CNN, VGG16 ve DenseNet modellerinin federe öğrenme stratejisi içerisinde eğitilen modellere göre daha fazla örnek ile eğitilmiştir. Dolayısıyla Tablo 4.4'deki sonuçlar bu duruma göre yorumlanmalıdır. En başarılı federe öğrenme stratejisi olan FedAvg daha az veri ile eğitilmesine rağmen işbirlikçi model yapısı sayesinde daha fazla veri ile eğitilen geleneksel model yapılarına yakın bir performans sergilemiştir. Bu sonuç federe öğrenme yapısının gizlilik sorununa olan katkısının yanı sıra az veriye sahip

istemcilerin de federe öğrenme sayesinde kabul edilebilir başarı da model geliştirebileceğini göstermiştir.

Çalışmada kullanılan stratejilerin federe öğrenme sürecini tamamlama süresi saniye cinsinden Tablo 4.5'de gösterilmiştir.

Tablo 4.5 : Stratejilere göre federe öğrenme sürecinin tamamlanma süresi

	FedAvg	QFedAvg	FT-FedAvg	DPFedAvg
İletişim Süresi	4490.27	3970.61	4982.88	5011.36

Tablo 4.5'de görüldüğü üzere en kısa süren strateji yaklaşımı Q-FedAvg olmuştur. Bu beklenen bir durumdur çünkü Q-FedAvg model ağırlıklarına sıkıştırma uygulamaktadır dolayısıyla modeller tam kapasite ile kullanılmadığı için iletişim süresi diğer stratejilere göre daha kısa sürmektedir.

Çalışmadaki asıl amacımız, verilerin paylaşılmadan cihazlar üzerinde kalmasını sağlayarak ortak bir model geliştirmek olsa da, incelenen ilk ve son turların sonuçları, tüm istemciler ve sunucular için işbirlikçi bir model geliştirildiğini açıkça göstermektedir.

BÖLÜM 5. SONUÇLAR VE ÖNERİLER

Son yıllarda, dünya çapında akademi, endüstri ve hükümetler tarafından federe öğrenmeye artan bir ilgi ve yatırım olmuştur. Federe öğrenimin etkinliğini, ölçeklenebilirliğini ve güvenliğini arttırmak için birçok araştırma makalesi, aracı ve çerçevesi geliştirilmiştir. Ayrıca, çeşitli endüstrilere ve sektörlere federe öğrenme hizmetleri ve çözümleri sağlamak için birçok şirket ve girişim ortaya çıkmıştır.

Federe öğrenmenin gelişmeye ve olgunlaşmaya devam ettikçe daha erişilebilir hale gelmesi ve böylece daha fazla kuruluşun federe öğrenmenin avantajlarından yararlanmasını sağlaması beklenmektedir. Doğru politikalar, düzenlemeler ve standartlarla Federe öğrenme, veri odaklı teknolojiler çağında yenilik, işbirliği ve değer yaratma için güçlü bir araç olabilmektedir.

Federe öğrenmenin, makine öğrenmesindeki mahremiyet endişelerini gidermeye yardımcı olsa da sihirli değnek bir çözüm olmadığını unutmamak önemlidir. Federe öğrenme yine de gizlilik, güvenlik ve etik konuların dikkatli bir şekilde değerlendirilmesini gerektirir ve kötüye kullanımı, suistimali veya istenmeyen sonuçları önlemek için uygun güvenceler ve kontroller yürürlükte olmalıdır. Bu nedenle, makine öğrenmesi, mahremiyet, güvenlik, etik ve hukuk alanlarında uzmanları içeren federe öğrenmeye yönelik çok disiplinli bir yaklaşıma sahip olmak önemlidir.

Sonuç olarak, federe öğrenme, mahremiyet ve mahremiyetten ödün vermeden makine öğrenmesi görevlerinde işbirliği yapmak için güvenli ve verimli bir yol sunan umut verici bir yaklaşımdır. Gizliliği ve güvenliği korurken verilerin daha verimli ve etkili bir şekilde kullanılmasını sağlayarak birçok sektörü ve sektörü dönüştürme potansiyeline sahiptir. Bu nedenle, sosyal ve ekonomik faydalar için tam potansiyelini ortaya çıkarmak için Federe öğrenmenin daha fazla araştırılmasına, geliştirilmesine ve benimsenmesine yatırım yapmaya ihtiyaç vardır.

Bu çalışma, Federe öğrenme ile tıbbi verilerin gizliliğini koruyarak bir model geliştirilip geliştirilemeyeceği, nispeten az veri ile eğitilen modellerin iş birliği sayesinde yüksek veri ile eğitilen modeller gibi performans verip veremeyeceği, farklı stratejilerin modelleri nasıl etkilediği ve diferansiyel mahremiyet eklemenin model performansını etkileyip etkilemediği sorularına cevap aramıştır. Deneyler, beyin tümörü veri kümesi üzerinde yapılmıştır. Araştırmanın katkısı şu şekilde özetlenebilir:

• Bu çalışma, tıbbi teşhis ve tedavide daha doğru sonuçlar için yeni bir araç sağlayan federe öğrenimin beyin tümörü verilerinde uygulanabilirliğini göstermektedir. Ayriyeten her bir istemcinin normalden az veri ile model geliştirdiği göz önünde bulundurulursa federe öğrenmenin iş birliği yapısı sayesinde az veri ile yüksek başarım elde edilebileceği de çalışmada gösterilmiştir.

• Veri gizliliği, tıbbi verilerin analizinde önemli bir husustur. Bu çalışmada kullanılan federe öğrenme yöntemi, veri gizliliğini korurken verilerin daha geniş kullanımına izin vermektedir. Bu nedenle, bu çalışma tıbbi verilerin analizi için güvenli ve etkili bir yöntem olarak kabul edilebilmektedir.

• Kurumların az sayıda ve çeşitte veri ile global modele katkıda bulunarak bu global modelin performansından faydalanabilecekleri görülmüştür.

• Federe öğrenme yöntemi dağıtık bir öğrenme yöntemi olduğundan, verilerin merkezi bir yerde toplanması ve saklanması ihtiyacını ortadan kaldırmaktadır. Dolayısıyla bu çalışma, tıbbi görüntüleme verilerinin analizini, e merkezi veri depolama altyapısına ihtiyaç duymadan ve hastaların meta verilerinin de elde edilmeden daha geniş bir kullanıma sunmaktadır.

Çalışmanın geliştirilebilirliği oldukça yüksektir. Örneğin bu çalışmada istemciler ve sunucu arası veri paylaşımı eşit olarak dağıtılmıştır. Bir sonraki adımda istemciler ve sunucular arası veri paylaşımı ve sınıflar arası dağılım dengesiz bir şekilde yapılabilir. Böylece istatistiksel heterojenlik sorunun çözümüne odaklanılabilir. Bu durumda kullanılan strateji yaklaşımları farklılık gösterecektir. İstatistiksel heterojenlik için oluşturulan FedProx gibi optimize ediciler bu sorunun çözümü için tercih edilebilir.

Bir başka ele alınabilecek durum ise haberleşme mimarisini değiştirmek olabilir. Bu çalışmada bir sunucu ve üç istemci vardır. Dolayısıyla merkezi haberleşme mimarisi kullanılmıştır. Gerçek hayat problemleri her zaman bir sunucuya ihtiyaç duymaz veya

bir sunucu temin edilemeyebilir. Bu durumda merkezi olmayan haberleşme mimarisi tercih edilebilmektedir. Merkezi olmayan haberleşme mimarisi ile sunucu olmaksızın tüm istemciler model ağırlıklarını kendi aralarında paylaşarak ortak model geliştirecektir. Merkezi olmayan haberleşme ile ortak model geliştirmenin zorlukları da ayriyeten ele alınmış olacaktır. Örneğin, bir istemci diğer istemcilere göre cihaz heterojenliğinden dolayı yavaş kalırsa oluşacak iletişim sıkıntısının çözümlenmesi, bir istemcinin yerel verilerinden dolayı olabilecek düşük model başarımının diğer istemcileri ve ortak modelin gelişimini etkilememesi adına çözüm üretilmesi gibi zorluklar ele alınmalıdır.

Bu çalışmada üç istemci ve bir sunucu simüle edilmiştir. Daha iyi bir model genellemesi için istemci sayısı ve veri adeti ileriki çalışmalarda arttırılabilir.

Çalışmada özelleştirilmiş bir CNN modeli kullanılmıştır fakat hazır, eğitilmiş bir çok derin öğrenme modeli probleme göre kullanılabilmektedir. Buna transfer öğrenimi denmektedir. Bu hazır eğitilmiş modeller de istemcilere entegre edilerek veya probleme göre daha komplike modeller geliştirilerek daha karmaşık modellerin ağırlıklarının paylaşılması ve ortak model geliştirilmesi sağlanabilir. Tabii karmaşık modellere sahip istemcilerin iletişim verimliliğini arttırmak, maliyeti azaltmak için çeşitli sıkıştırma metotları uygulanabilir ve farklı sıkıştırma metotlarının performansa etkisi gözlemlenebilir.

Sonuç olarak, bu alandaki araştırmaların önemi, giderek artan miktarda veriye sahip olunan günümüzde, veri güvenliği ve mahremiyeti sorunlarının yanı sıra, öğrenme modellerinin dağıtılmış olarak eğitimi açısından büyük bir önem taşımaktadır.

Bir sonraki adım, daha fazla çalışma ve deneysel testler yaparak, bu konuda daha iyi bir anlayışa sahip olmak olacaktır. Ayrıca, işbirlikçilerin özellikleri, model mimarileri ve katılımcı sayısı gibi faktörlerin performans üzerindeki etkisini daha iyi anlamak için yukarıda bahsedilen farklı senaryoları simüle etmek gerekebilir.

Bu tezin, akademik araştırmalarda ve tıp endüstrisinde birçok uygulama alanı bulunmaktadır. Bu uygulama ile tıbbi verilerin güvenliği sağlanırken aynı zamanda AI modellerinin etkin bir şekilde kullanımı artmaktadır. Bu sayede sadece beyin tümörü sınıflandırma değil aynı zamanda bir çok hastalık kategorisinde gerek sınıflandırma gerek segmentasyon çalışmaları yapılabilir.

KAYNAKLAR

Adnan, M. S. G., Kalra, S., Cresswell, J. C., Taylor, G. P., & Tizhoosh, H. R. (2022). Federated learning and differential privacy for medical image analysis. *Scientific Reports*, *12*(1). https://doi.org/10.1038/s41598-022-05539-7

Agarap, A. F. (2018, March 22). *Deep Learning using Rectified Linear Units (ReLU)*. arXiv.org. https://arxiv.org/abs/1803.08375

Alistarh, D. (2016, October 7). *QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding*. arXiv.org. https://arxiv.org/abs/1610.02132

Bartoletti, I. (2019). AI in Healthcare: Ethical and Privacy Challenges. *Lecture Notes in Computer Science* (pp. 7–10). Springer Science+Business Media. https://doi.org/10.1007/978-3-030-21642-9_2

Beutel, D. J. (2020). *Flower: A Friendly Federated Learning Research Framework*. arXiv.org. https://arxiv.org/abs/2007.14390

Biratu, E. S., Schwenker, F., Ayano, Y. M., & Debelee, T. G. (2021). A Survey of Brain Tumor Segmentation and Classification Algorithms. *Journal of Imaging*, *7*(9), 179. https://doi.org/10.3390/jimaging7090179

Brinker, T. J., Hekler, A., Utikal, J., Grabe, N., Schadendorf, D., Klode, J., Berking, C., Steeb, T., Enk, A., & Von Kalle, C. (2018). Skin Cancer Classification Using Convolutional Neural Networks: Systematic Review. *Journal of Medical Internet Research*, *20*(10), e11936. https://doi.org/10.2196/11936

Chellapandi, V. P. (2023, March 19). *On the Convergence of Decentralized Federated Learning Under Imperfect Information Sharing*. arXiv.org. https://arxiv.org/abs/2303.10695

Chen, S., Xue, D., Chuai, G., Yang, Q., & Liu, Q. (2020). FL-QSAR: a federated learning based QSAR prototype for collaborative drug discovery. *bioRxiv (Cold Spring Harbor Laboratory)*. https://doi.org/10.1101/2020.02.27.950592

Cheung, Y., & Yu, F. (2020). Federated-PCA on Vertical-Partitioned Data. *Federated-PCA on Vertical-Partitioned Data*. https://doi.org/10.36227/techrxiv.12331649.v1

Das, A., & Patterson, S. (2021). *Multi-Tier Federated Learning for Vertically Partitioned Data*. https://doi.org/10.1109/icassp39728.2021.9415026

Dongqi, C., Fan, T., Kang, Y., Fan, L., Xu, M., Wang, S., & Yang, Q. (2022). Accelerating Vertical Federated Learning. *IEEE Transactions on Big Data*, 1–10. https://doi.org/10.1109/tbdata.2022.3192898

El-Wahab, B. S. A., Nasr, M. E., Khamis, S., & Ashour, A. S. (2023). BTC-fCNN: Fast Convolution Neural Network for Multi-class Brain Tumor Classification. *Health Information Science and Systems*, *11*(1). https://doi.org/10.1007/s13755-022-00203-w

Filatov, D. (2022, July 27). *Brain Tumor Diagnosis and Classification via Pre-Trained Convolutional Neural Networks*. arXiv.org. https://arxiv.org/abs/2208.00768

Ghanbari, H., Mahdianpari, M., Homayouni, S., & Mohammadimanesh, F. (2021). A Meta-Analysis of Convolutional Neural Networks for Remote Sensing Applications. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *14*, 3602–3613. https://doi.org/10.1109/jstars.2021.3065569

Ghosh, A., Kole, A. (2021), *A Comparative Study of Enhanced Machine Learning Algorithms for Brain Tumor Detection and Classification.* TechRxiv. Preprint. https://doi.org/10.36227/techrxiv.16863136.v1

Ghosh, A. (2020, June 7). *An Efficient Framework for Clustered Federated Learning*. arXiv.org. https://arxiv.org/abs/2006.04088

Ghosh, S., & Rudy, Y. (2009). Application of L1-Norm Regularization to Epicardial Potential Solution of the Inverse Electrocardiography Problem. *Annals of Biomedical Engineering*, *37*(5), 902–912. https://doi.org/10.1007/s10439-009-9665-6

Glasgow, M. (2021, November 5). *Sharp Bounds for Federated Averaging (Local SGD) and Continuous Perspective*. arXiv.org. https://arxiv.org/abs/2111.03741

Gómez-Guzmán, M. A., Jiménez-Beristáin, L., García-Guerrero, E. E., López-Bonilla, O. R., Tamayo-Perez, U. J., Esqueda-Elizondo, J. J., Vizcaino, K. P., & Inzunza-González, E. (2023). Classifying Brain Tumors on Magnetic Resonance Imaging by Using Convolutional Neural Networks. *Electronics*, *12*(4), 955. https://doi.org/10.3390/electronics12040955

Hard, A. (2018, November 8). *Federated Learning for Mobile Keyboard Prediction*. arXiv.org. https://arxiv.org/abs/1811.03604

Hong, P. -Y. Wang and W. -B. Zhao, "Homomorphic Encryption Scheme Based on Elliptic Curve Cryptography for Privacy Protection of Cloud Computing," 2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), New York, NY, USA, 2016, pp. 152-157, doi: 10.1109/BigDataSecurity-HPSC-IDS.2016.51.

Hu, S., Li, Y., Liu, X., Li, Q., Wu, Z., & He, B. (2020). The OARF Benchmark Suite: Characterization and Implications for Federated Learning Systems. *ACM Transactions on Intelligent Systems and Technology*, *13*(4), 1–32. https://doi.org/10.1145/3510540

Hu, S., Zhang, Z., & Mo, K. (2023). Homomorphic Encryption and its Application to Blockchain. *Frontiers in Computing and Intelligent Systems*, *3*(1), 110–112. https://doi.org/10.54097/fcis.v3i1.6343

Huang, W. (2020, December 18). *Fairness and Accuracy in Federated Learning*. arXiv.org. https://arxiv.org/abs/2012.10069

Islam, M., Reza, M. T., Kaosar, M., & Parvez, M. Z. (2022). Effectiveness of federated learning and CNN ensemble architectures for identifying brain tumors using MRI images. Neural Processing Letters. https://doi.org/10.1007/s11063-022-11014-1

Kadykov, V., Levina, A., & Voznesenskiy, A. S. (2021). Homomorphic Encryption within Lattice-Based Encryption System. *Procedia Computer Science*, *186*, 309–315. https://doi.org/10.1016/j.procs.2021.04.149

Kandhro, I. A., Jumani, S. Z., Ali, F., Shaikh, Z. U., Arain, M. A., & Shaikh, A. A. (2020). Performance Analysis of Hyperparameters on a Sentiment Analysis Model. *Engineering, Technology & Applied Science Research*, *10*(4), 6016–6020. https://doi.org/10.48084/etasr.3549

Kumari, D., & Bhat, S. (2022). A Review on Brain Tumor Detection Using Convolutional Neural Network. *Zenodo (CERN European Organization for Nuclear Research)*. https://doi.org/10.5281/zenodo.7048369

Lai, P. (2022, January 28). *Bit-aware Randomized Response for Local Differential Privacy in Federated Learning*. OpenReview. https://openreview.net/forum?id=ZUXZKjfptc9

Li, Q. (2023, February 3). *Vertical Federated Learning: Taxonomies, Threats, and Prospects*. arXiv.org. https://arxiv.org/abs/2302.01550

Li, Q., Diao, Y., Chen, Q., & He, B. (2021). *Federated Learning on Non-IID Data Silos: An Experimental Study.* arXiv (Cornell University). https://doi.org/10.48550/arxiv.2102.02079

Li, T. (2019, May 25). *Fair Resource Allocation in Federated Learning*. arXiv.org. https://arxiv.org/abs/1905.10497

Mahesh, A., Banerjee, D., Saha, A., Prusty, M. R., & Balasundaram, A. (2023). CE-EEN-B0: Contour Extraction Based Extended EfficientNet-B0 for Brain Tumor Classification Using MRI Images. *Computers, Materials & Continua*, *74*(3), 5967–5982. https://doi.org/10.32604/cmc.2023.033920

Mahmud, M. H., Huang, J. Z., Salloum, S., Emara, T. Z., & Sadatdiynov, K. (2020). A survey of data partitioning and sampling methods to support big data analysis. *A*

Survey of Data Partitioning and Sampling Methods to Support Big Data Analysis, *3*(2), 85–101. https://doi.org/10.26599/bdma.2019.9020015

Malla, P. P., Sahu, S., & Alutaibi, A. I. (2023). Classification of Tumor in Brain MR Images Using Deep Convolutional Neural Network and Global Average Pooling. *Processes*, *11*(3), 679. https://doi.org/10.3390/pr11030679

Mandle, A. K., Sahu, S. P., & Gupta, G. P. (2022). CNN-Based Deep Learning Technique for the Brain Tumor Identification and Classification in MRI Images. *International Journal of Software Science and Computational Intelligence*, *14*(1), 1–20. https://doi.org/10.4018/ijssci.304438

Manne, R., Kantheti, S., & Kantheti, S. (2020). Classification of Skin cancer using deep learning, Convolutional Neural Networks - Opportunities and vulnerabilities- A systematic Review. *International Journal of Modern Trends in Science and Technology*, *6*(11), 101–108. https://doi.org/10.46501/ijmtst061118

McMahan, H. B. (2016, February 17). *Communication-Efficient Learning of Deep Networks from Decentralized Data*. arXiv.org. https://arxiv.org/abs/1602.05629

Meshram, S., & Hasamnis, M. A. (2012). Huffman Encoding using VLSI. *International Journal of Electrical and Electronics Engineering*. 1, 143-145, https://doi.org/10.47893/ijeee.2012.1028

Mittal, S., & Ramachandran, R. (2022). Understanding integer-based fully homomorphic encryption. *ResearchGate*. https://doi.org/10.1063/5.0080604

Morell, J. V., & Alba, E. (2022). Dynamic and adaptive fault-tolerant asynchronous federated learning using volunteer edge devices. *Future Generation Computer Systems*, *133*, 53–67. https://doi.org/10.1016/j.future.2022.02.024

Naehrig, M., Lauter, K. E., & Vaikuntanathan, V. (2011). *Can homomorphic encryption be practical?* https://doi.org/10.1145/2046660.2046682

Niknam, S. (2019, July 30). *Federated Learning for Wireless Communications: Motivation, Opportunities and Challenges*. arXiv.org. https://arxiv.org/abs/1908.06847

R, P., & Pattabiraman, V. (2022). Identification of Subtype Blood Cells Using Deep Learning Techniques. In *Advances in information security, privacy, and ethics book series* (pp. 270–285). IGI Global. https://doi.org/10.4018/978-1-6684-5250-9.ch014

Rasheed, Z., Ma, Y., Ullah, I., Shloul, T. A., Tufail, A. B., Ghadi, Y. Y., Khan, M. Z., & Mohamed, H. G. (2023). Automated Classification of Brain Tumors from Magnetic Resonance Imaging Using Deep Learning. *Brain Sciences*, *13*(4), 602. https://doi.org/10.3390/brainsci13040602

Raza, A., Ayub, H., Khan, J., Ahmad, I., Salama, A., Daradkeh, Y. I., Javeed, D., Rehman, A. U., & Hamam, H. (2022). A Hybrid Deep Learning-Based Approach for Brain Tumor Classification. *Electronics*, *11*(7), 1146. https://doi.org/10.3390/electronics11071146

Sadoon, T. a. U., & Ali, M. K. (2021). Deep learning model for glioma, meningioma and pituitary classification. *International Journal of Advances in Applied Sciences*, *10*(1), 88. https://doi.org/10.11591/ijaas.v10.i1.pp88-98

Saeed, M., Halepoto, I. A., Khaskheli, S., & Bushra, M. (2023). Optimization and efficiency analysis of deep learning based brain tumor detection. *Mehran University Research Journal of Engineering and Technology*, *42*(2), 188. https://doi.org/10.22581/muet1982.2302.19

Sarkar, A., Maniruzzaman, M., Alahe, M. A., & Ahmad, M. (2023). An Effective and Novel Approach for Brain Tumor Classification Using AlexNet CNN Feature Extractor and Multiple Eminent Machine Learning Classifiers in MRIs. *Journal of Sensors*, *2023*, 1–19. https://doi.org/10.1155/2023/1224619

Sattler, F., Wiedemann, S., Müller, K., & Samek, W. (2020). Robust and Communication-Efficient Federated Learning From Non-i.i.d. Data. *IEEE Transactions on Neural Networks and Learning Systems*, *31*(9), 3400–3413. https://doi.org/10.1109/tnnls.2019.2944481

Scheibner, J., Raisaro, J. L., Troncoso-Pastoriza, J. R., Vayena, E., Fellay, J., Vayena, E., & Hubaux, J. (2021). Revolutionizing Medical Data Sharing Using Advanced Privacy-Enhancing Technologies: Technical, Legal, and Ethical Synthesis. *Journal of Medical Internet Research*, *23*(2), e25120. https://doi.org/10.2196/25120

Shen, S., Zhu, T., Wu, D., Wang, W., & Zhou, W. (2020). From distributed machine learning to federated learning: In the view of data privacy and security. *Concurrency and Computation: Practice and Experience*, *34*(16). https://doi.org/10.1002/cpe.6002

Shi, P., & Huang, H. (2022). Lightweight MobileNetV2 offline handwritten Chinese character recognition based on attention mechanism. In *International Symposium on Robotics, Artificial Intelligence, and Information Engineering (RAIIE 2022)*. https://doi.org/10.1117/12.2659091

Singh, A. (2019, September 18). *Detailed comparison of communication efficiency of split learning and federated learning*. arXiv.org. https://arxiv.org/abs/1909.09145

Śliwa, M., & Pańczyk, B. (2021). Performance comparison of programming interfaces on the example of REST API, GraphQL and gRPC. *Journal of Computer Sciences Institute*, *21*, 356–361. https://doi.org/10.35784/jcsi.2744

Sultan, H. H., Salem, N. M., & Al-Atabany, W. (2019). Multi-Classification of Brain Tumor Images Using Deep Neural Network. *IEEE Access*, *7*, 69215–69225. https://doi.org/10.1109/access.2019.2919122

Tian, Y., Zhang, Y., & Zhang, H. (2023). Recent Advances in Stochastic Gradient Descent in Deep Learning. *Mathematics*, *11*(3), 682. https://doi.org/10.3390/math11030682

Tiwari, P. C., Pant, B., Elarabawy, M., Abd-Elnaby, M., Mohd, N., Dhiman, G., & Sharma, S. (2022). CNN Based Multiclass Brain Tumor Detection Using Medical Imaging. *Computational Intelligence and Neuroscience*, *2022*, 1–8. https://doi.org/10.1155/2022/1830010

Viet, K. L. D., Ha, K., Quoc, T. N., & Hoang, V. T. (2023). MRI Brain Tumor Classification based on Federated Deep Learning. *MRI Brain Tumor Classification Based on Federated Deep Learning*. https://doi.org/10.1109/zinc58345.2023.10174015

Wang, J. (2021, June 4). *Local Adaptivity in Federated Learning: Convergence and Consistency*. arXiv.org. https://arxiv.org/abs/2106.02305

Wang, Y., Li, Y., Song, Y. S., & Rong, X. (2020). The Influence of the Activation Function in a Convolution Neural Network Model of Facial Expression Recognition. *Applied Sciences*, *10*(5), 1897. https://doi.org/10.3390/app10051897

Wu, W., & Yang, X. (2022). A federated learning differential privacy algorithm for non-Gaussian heterogeneous data. *Authorea (Authorea)*. https://doi.org/10.22541/au.166618858.84732188/v1

Xing, H. (2020, February 28). *Decentralized Federated Learning via SGD over Wireless D2D Networks*. arXiv.org. https://arxiv.org/abs/2002.12507

Xu, X., Lin, L., Sun, S., & Wu, S. (2023). A review of the application of three-dimensional convolutional neural networks for the diagnosis of Alzheimer's disease using neuroimaging. *Reviews in the Neurosciences*. https://doi.org/10.1515/revneuro-2022-0122

Yamashita, R., Nishio, M., G, R. K., DO, & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. *Insights Into Imaging*, *9*(4), 611–629. https://doi.org/10.1007/s13244-018-0639-9

Yang, C., Wang, Q., Xu, M., Chen, Z., Bian, K., Liu, Y., & Liu, X. (2021). *Characterizing Impacts of Heterogeneity in Federated Learning upon Large-Scale Smartphone Data*. https://doi.org/10.1145/3442381.3449851

Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W., & Gao, Y. (2021). A survey on federated learning. *Knowledge Based Systems*, *216*, 106775. https://doi.org/10.1016/j.knosys.2021.106775

Zhang, Y., Wang, Y., Hong, S., & Gui, G. (2019). Blind Channel Identification Aided Generalized Automatic Modulation Recognition Based on Deep Learning. *IEEE Access*, *7*, 110722–110729. https://doi.org/10.1109/access.2019.2934354

Url-1 <*https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset*>, erişim tarihi: 02.07.2021

Url-2 <*https://figshare.com/articles/dataset/brain_tumor_dataset/1512427*>, erişim tarihi: 02.04.2017

Url-3 <*https://www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri*>, erişim tarihi: 07.06.2020

Url-4 <*kaggle.com/datasets/ahmedhamada0/brain-tumor-detection?select=no*>, erişim tarihi: 02.07.2021