FEDERE MAKİNE ÖĞRENMESİ: KAVRAMLAR, ZORLUKLAR, YÖNTEMLER

FEDERATED MACHINE LEARNING: CONCEPTS, CHALLENGES, AND METHODS

MAKBULE MELISA KILIÇ

PROF. DR. EMRE AKTAŞ
TEZ DANIŞMANI

Hacettepe Üniversitesi

Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin

Elektrik ve Elektronik Mühendisliği Anabilim Dalı için Öngördüğü

YÜKSEK LİSANS TEZİ olarak hazırlanmıştır.

2023

ÖZET

FEDERE MAKİNE ÖĞRENMESİ: KAVRAMLAR, ZORLUKLAR, YÖNTEMLER

Makbule Melisa KILIÇ

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Danışmanı: Prof. Dr. Emre AKTAŞ

Haziran 2023, 145 sayfa

MC MAHAN ve ekibinin öncülük ettiği Federe öğrenmesi, ilk defa 2016 yılında Google'da ele alınmış olup, o tarihten beri bu konu ile ilgili çalışmalar artarak devam etmektedir. Ele alınan istemci-sunucu modelinde, istemciler veri içeren cihazlardır, bu istemciler sisteme eklenirler ve katılımcı olarak da adlandırılırlar. Hedef istemcilerdeki veriyi kullanarak makine öğrenmesinin eğitimini yapmaktır. Daha sonrasında eğitilmiş model yine istemcilerde kullanılacaktır. Bunun için bir yöntem, katılımcıların verilerini merkezi sunucuya aktarmaları, sunucuda bütün verinin kullanılarak eğitimin gerçekleştirilmesi, daha sonra eğitilmiş modelin katılımcılarla geri paylaşılmasıdır. Bu yöntem geleneksel merkezi makine öğrenmesidir. Federe öğrenmede ise katılımcılar ham veriyi merkezi sunucuya göndermezler. Bu yöntemde, sunucu ve katılımcılar birleşik olarak çalışarak verilerin cihazlardan çıkmamasına olanak tanınır. Gelişen teknoloji ile, verinin merkezi sunucu yerine kenar katılımcılarda olduğu uygulamalar artmakta, aynı zamanda kenar katılımcıdaki verinin büyüklüğü artmaktadır. Kenardaki büyük verinin merkeze aktarılmasının maliyeti ve ayrıca buradaki kişisel verilerin paylaşılmamasının hassasiyeti, federe öğrenmenin önemini gelecekteki sistemler için artırmaktadır ve bu araştırma konusunu çok incelenen bir alan haline getirmektedir. Federe öğrenmede

istemciler, sunucudan gelen modelleri, kendi verilerini kendi cihazlarında kullanarak eğitirler. Oluşturulan yeni modeller sunucuda birleştirilerek tekrar cihazlara dağıtılır.

Bu tez çalışmasında, federe makine öğrenmesinin temelleri incelenmiş ve anlatılmıştır. Federe öğrenmenin temel zorlukları olan sistem heterojenliği, istatistiksel heterojenlik ve veri düzensizliğinin ele alınmış ve etkileri incelenmiştir. Bu zorlukların üstesinden gelmek üzere literatürde yeni birleştirme yöntemleri geliştirilmesi güncel bir konudur. Bu tezde bu yöntemlerden FEDSGD, FEDAVGM ve FEDAVG incelenmiştir.

Federe öğrenmeyi ve parametre değişikliklerinin sistem üzerindeki etkilerini incelemek için deneyler düzenlenmiştir. Yapılan deneylerde ayrıca federe öğrenme zorluklarından olan sistem heterojenliği, istatistiksel heterojenlik ve veri düzensizliği kavramlarının doğruluk, kayıp ve hesaplama süresi performansı üzerindeki etkisi FEDAVGM yöntemi için incelenmiştir. Model parametrelerinin her yinelemede değişim hızını kontrol etmek için farklı yöntemler vardır. Bunun için iki farklı yöntem olan FEDAVGM ve FEDAVG yöntemleri deneylerde karşılaştırılmıştır.

Sonuç olarak öğrenme parametre seçiminin sistem performansı üzerinde belirgin etkilerinin olabileceği görülmüştür. Verilen parametre değerlerinin duruma en uygun şekilde seçilmediği durumlarda aşırı öğrenme yatkınlığı oluşabilmektedir. Sistem heterojenliğin öğrenme süresini nasıl uzattığı, veri düzensizliğinin cihazlarda farklılığa sebep olup hesaplama süresini uzatabildiği ve bunun çeşitli sorunlara yol açabileceği, istatistiksel heterojenliğin öğrenmeyi azaltıp dalgalanmaları arttırdığı deneylerde gözlemlenmiştir. Belirli durumlarda FEDAVGM, momentum ile sistemin geçmiş gradyan hesaplamalarına dayanarak sonuç verdiği için daha başarılı olduğu görülmüştür.

Anahtar Kelimeler: Federe öğrenme, Konvolüsyon Sinir Ağları, Ortalama Federe Öğrenme, Sistem Heterojenliği, İstatistiksel Heterojenlik, Veri Düzensizliği

ABSTRACT

FEDERATED MACHINE LEARNING: CONCEPTS, CHALLENGES, AND METHODS

Makbule Melisa KILIÇ

MSc, Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Emre AKTAŞ
June 2019, 145 pages

The Federated Learning (FL) research, led by MC Mahan and his team, was first introduced at Google in 2016. Since then, this field has been increasing interest and ongoing research. In the client-server model used, the clients are devices that hold data and are referred to as participants. The goal is to train machine learning models using the data on the participating clients. The trained model is then used again on the clients. One approach is to have the participants send their data to a central server, where the training is performed using the entire data, and then share the trained model with the participants. This approach follows the traditional centralized machine learning paradigm. In federated learning, however, the participants do not send their raw data to the central server. Instead, the server and participants collaborate to ensure the data remains on the devices. With advancing technology, applications where data resides on edge participants rather than a central server, are increasing, and the size of data on edge participants is also growing. The cost of transferring large amounts of data from the edge to the center and the privacy concerns associated with sharing personal data highlight the importance of federated learning for future systems and make it a widely studied research area. In federated learning, the clients train the received models from the server using their data on their

own devices. The newly generated models are then aggregated on the server and redistributed to the devices.

This thesis examines and explains the fundamentals of federated machine learning. The critical challenges of federated learning, such as system heterogeneity, statistical heterogeneity, and data sparsity, are addressed, and their impacts are studied. Developing new aggregation methods to overcome these challenges is a current topic in the literature. In this thesis, three of these methods, FEDSGD, FEDAVGM, and FEDAVG, are investigated.

Experiments are conducted to examine federated learning and the effects of parameter changes on the system. In these experiments, the impact of system heterogeneity, statistical heterogeneity, and data sparsity on accuracy, loss, and computation time performance is explicitly analyzed for the FEDAVGM method. Different methods exist to control the rate of parameter changes in each iteration. For this purpose, two different methods, FEDAVGM and FEDAVG, are compared in the experiments.

The results show that selecting learning parameters can significantly affect the system's performance. Overfitting tendencies can arise when the parameter values are not chosen appropriately for the situation. It is observed through the experiments that system heterogeneity prolongs the learning time, data sparsity introduces variations across devices and increases computation time, and statistical heterogeneity reduces learning and increases oscillations. In specific cases, FEDAVGM has shown better performance, relying on past gradient calculations with momentum.

Keywords: Federated learning, Convolutional Neural Networks, FEDAVG, FEDSGD, System Heterogeneity, Statistical Heterogeneity, Data Heterogeny

TEŞEKKÜR

Lisansüstü ve lisans eğitimim boyunca bilgi ve tecrübelerinden yararlandığım, tez boyunca desteğini her zaman hissettiğim, tecrübesine güvendiğim, beraber çalışmaktan çok mutlu olduğum, değerli hocam Prof. Dr. Emre Aktaş'a,

Hayatım boyunca her koşulda bana desteğini vermiş, vermeye devam eden, bu süreçte de kaçtığım sorumlulukları bana hiç hissettirmeden karşılıksız seven canım aileme,

Bu dönemde kahrımı çeken, bütün sızlanmalarımla beni seven, canımın içi arkadaşlarıma,

Ve bu süreçte her zaman yanımda olup bana sonsuz destek veren, ihtiyacım olup olmamasından bağımsız her orada olan, her zaman hayatımda olmasını dileyeceğim, eski iş arkadaşım, meslektaşım, en yakın arkadaşım, erkek arkadaşım Ataberk Akgül'e,

Sonsuz Teşekkürler…

Melisa KILIÇ
Haziran 2023, Ankara

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

SİMGELER VE KISALTMALAR

Simgeler

β Geçmiş gradyanların işleme ne kadar katılacağını belirleyen sembol. Başlangıç değeri 0.9'dur.

t Zaman adımı

w_t Mevcut ağırlık

α Öğrenme katsayısı

$\frac{\partial L}{\partial w_t}$ Mevcut gradyan, kayıp fonksiyonunun w parametresine göre türevi

$\mathcal{D}$ Sunucunun katılımcılardan topladığı bütün veri seti

$\mathcal{D}_k$ k'ıncı katılımcının verisi

D_k k'ıncı katılımcının veri boyutu

$K.$ Katılımcı sayısı

n Bütün katılımcılarda bulunan toplam örnek sayısı,

n_k k'ıncı katılımcının örnek sayısını

η Öğrenme katsayısı

∇w_k k. katılımcıdan gelen ağırlık güncellenmesi

J Çoklu sınıflandırmadaki sınıf sayısını ifade eder.

e^{x_i} Her giriş değerine uygulanır. Giriş çok büyükse büyük bir çıktı, giriş çok küçükse küçük bir çıktı olur.

x_i Herhangi bir gerçek sayı değerini alabilir.

Kısaltmalar

FÖ. Federe Öğrenme

SSA Sıradan Sinir Ağları

ESA Evrişimsel Sinir Ağları

Gİ Gradyan İniş

SGİ Stokastik Gradyan İniş

1. GİRİŞ

1.1. Motivasyon ve Problem Tanımı

Yapay zekâ, insan zekasını taklit eden sistemler veya makineler anlamına gelmektedir. Makine öğrenimi (ML), kural tabanlı olmayarak, deneyimden öğrenme ve bu öğrenimini sürekli geliştirmeye el verişli olan yapay zekanın bir alt kümesidir. Günümüzde büyük veri ile karşı karşıyayız bu veri çokluğunu anlamlandırabilmek, değerli hale getirebilmek için makine öğrenme algoritmaları güçlü bir araç olarak düşünülebilir. Örneğin, dijital ikiz gerçek dünyadaki bir nesnenin dijital ortamda her özelliğiyle incelenebilir olmasıdır. Elde edilen gerçek veri, dijital ikiz eş zamanlı çalışarak, insanlar çalışmalarını gerçek ortam yerine dijital ortamda gerçekleştirebilirler. Böylelikle bir fabrikada çıkan sorunları, üretim verimini, süreç yönetimini , gerçekleşebilecek hata tespitini sağlayabilirler. Geleneksel makine öğrenmesi, merkez ve merkeze bağlı cihazlardan oluşur. Merkeze bağlı cihazlardan veri merkezle paylaşılır. Bütün veri merkezde toplanır. Bu durum merkeze bağlı cihazlar için sürekli meşgul olma, batarya kullanım yoğunluğu açısından olumludur. Bir grup cihazın katılımı ile modeli genelleştirebilir, genelleştirilen model uyumlu cihazlarda kullanılabilir hale gelebilir. Ancak, büyük veri için veri transferi, işlem gücü zorluğu, gizlilik geleneksel öğrenmenin yetersiz kaldığı alanlardır.

Eğitimin tek bir sunucuya odaklanılmasından, sistemin dağıtık olarak ele alınması gündeme gelmiştir. Dağıtık makine öğrenmesinde büyük miktardaki veri birden fazla kaynakta dağıtık bir şekilde eğitilmektedir. Her cihaz, kendi verileriyle, modelini eğitir.

Mc Mahan ve ekibinin öncülük ettiği federe öğrenmesi, ilk defa 2016 yılında, Google'da ele alınmış olup araştırmalar artarak günümüze kadar uzanmıştır. Federe öğrenmesinin gelişme konseptinde yer alan makalelerin başında [30], [8], [35] referansları gelmektedir. Bu makalelerde, merkezi eğitim verileri olmadan işbirlikçi makine öğrenmesini sağlayan federe öğrenme konusu ele alınmaktadır. Dağıtık makine öğrenmesinin özelleşmiş bir şekli olan federe öğrenme sunucu ve istemcilerden oluşur. İstemci yerel olarak modeli eğiten, modelin gelişmesine katkıda bulunan cihazlardır. Katılımcı ve müşteri olarak da adlandırılırlar. Federe öğrenmede, her cihaz, kendi verileriyle, modelin bir kısmını yerel

olarak eğitir. Daha sonra yerel veri ile eğitilen model merkezi bir sunucuda birleştirilir. Merkezi sunucu bütün katılımcıların iş birliğiyle birleştirdiği geliştirdiği modeli cihazlarla paylaşır. Federe öğrenmede paylaşılan veri değil model olup, veri gizliliğini korur. Federe öğrenmenin güvenlik, gizlilik ve veri bütünlüğü gibi önemli avantajları mevcuttur. Derin ağların iletişim açısından verimli bir şekilde öğrenilmesini dağınık veriler ile sağlamaktadır. Sistemin verimli olarak düşünülmesinin ana sebebi, Model parametrelerinin güncellenmesi sırasında sınırlı miktarda iletişim gerektirmesidir.

Geleneksel yöntem olan merkezi makine öğrenmesi, veri merkezli yaklaşımı, iletişim maliyeti, güvenilirlik, çok büyük veri boyutlarının tek bir sunucuda toplanması gibi konularda yetersiz kalabilmektedir. Dağıtık öğrenme, güvenirlik konusunda başarı sağlarken, iş birliği konusunda yetersiz kalmıştır. Geleneksel yöntemin eksikliklerinden yola çıkan, özel dağıtılmış bir makine öğrenmesi olan federe öğrenme, tanım olarak, istemcilerin, sunucudan aldıkları modeller ile kendi verilerini sunuya göndermeden, modellerini eğittikleri ve eğitilen modellerin sunucuda toplanarak, belirlenen yöntemlerle birleştirildikleri bir makine öğrenme yöntemidir. Bu sayede, her yinelemede bütün istemcilerin modellerinin birleşimi, diğer istemcilere yayılır. Bu sistem ile istemci sadece kendi verisi ile eğitimini tamamlamak zorunda kalmadan diğer istemcilerle iş birliği içinde öğrenimini gerçekleştirmektedir. Modellerin istemcilere gönderilmesi, model en uygun değere yakınsayana kadar devam etmektedir.

Federe öğrenmesi, derin öğrenme modellerinin eğitiminde kullanılabilecek bir yöntem olup derin öğrenme modellerinin veri gizliliği ve güvenliği gibi konularda karşılaştığı sorunlara çözüm sunmaya çalışmaktadır [1]. Makine öğrenmesindeki veri gizliliğinin ihlali problemini çözmeyi amaçlayan federe öğrenmesi, modelin merkezi bir sunucu denetimine tabi tutulmasıyla ortaya çıkmıştır. Sistem, istemcilerin her birinin kendi verilerini kendi cihazlarında tutmasın ve verinin iletiminin gerçekleştirilmemesini hedeflemektedir. Bu yaklaşımda, verinin bulunduğu kaynağın ve verinin işlemesinin yakın yerlerde olması gerektiği savunulmaktadır. Gelişen teknolojiyle birlikte, verinin boyut olarak artışı ve farklılaşması sebebiyle, veri gizliliği riskinin önemini kaçınılmaz bir şekilde arttırmıştır. Veri iletiminin olmaması, istemcilerin eğitim için kendi modellerini paylaşması, veri gizliliğine önemli ölçüde katkıda bulunur. İhtiyaç duyulan

mahremiyet için önerilen yöntemlerden biri olan federe öğrenme bu konuda bu sorunun çözümü olarak görülebilir.

Federe öğrenmesinin uygulama alanlarının artması beklenmektedir. Otomotivden sağlığa, endüstriyel yönetimden, nesnelerin interneti (IoT) alanına, telekomünikasyondan finansa gibi birçok alanda çalışmalar devam etmektedir [4]. Özellikle tıp alanında federe öğrenme, verilerin gizliliğinin ön planda tutulma politikası ve veri yoğunluğu sebebiyle, ön plana çıkmaktadır. Dijital ortam, bütün hastaların hastalık raporları, sağlık kayıtları, kişisel bilgileri, tahlilleri içermektedir. Hastanelerin, diğer hastanelerle iş birliği ile, verilerini paylaşmadan, merkezi sunucu denetiminde, kendi modellerini eğitmeleri, makine öğrenmesiyle teşhis teknolojisinin ilerlemesine katkıda bulunacaktır. Yapay zekanın tıp alanına yayılmasıyla, hassasiyeti artan gizlilik ihlalleri konusunda federe öğrenme ile çözüm bulunabilir. Federe öğrenme sisteminin gelişmesiyle, tıp alanında veri yönetimi ve teşhis doğruluğu konusunda çok fazla ilerleme gösterilebilir [36][37].

Yeni gelişen sistemlerde, bazı zorluklar olması da kaçınılmazdır. Federe öğrenmenin endüstriyel alanlarda uygulanmasıyla gelen bazı zorluklar olacaktır. Bu zorluklar yeni birleştirme yöntemleri ile aşılabilir, çözümler bulunabilir. Yeni birleştirme yöntemleri ile gelişmeye devam ettikçe endüstride kullanım alanlarının artması beklenmektedir [37][28].

Sistem heterojenliği, istatistiksel heterojenlik, iletişim, güvenlik ve gizlilik konularının federe öğrenme zorluklarının arasında yer aldığı söylenebilir [4]. Bu zorlukların kaynaklanma yerleri müşteriler, sunucu, veri, dağıtım, model, birleştirme gibi çok çeşitli alanda olabilir [3]. Bu sebeple güncel çalışmalarda federe öğrenme algoritmalarını geliştirmeyi ve zorlukların şekline göre algoritmaları sınıflandırmayı hedeflemektedir. Algoritmaların güncel zorluk ya da performans düşüklüklerine destek olması yönünde çalışma yapılarak zorlukların ve iletişim güçlüklerinin azaltılması hedeflenmektedir [38][28].

Tez kapsamında, geleneksel yöntemin zorluklarından, yeni teknolojinin getirisinden oluşan ihtiyaçla ve dağıtık teknolojinin iş birliğine olan ihtiyacıyla birlikte bu sorunlara çözüm arayan bir sistem olan federe öğrenme incelenmiştir. Sistemin işleyiş yöntemi açıkça ortaya konmuştur. Devamında federe öğrenme yöntemlerinden FEDSGD, FEDAVG ve FEDAVGM yöntemlerine değinilmiş, federe öğrenme yöntemlerinin sınıflandırılmasına yakın zamanda yayınlanan bir çalışma olan [28] numaralı referanstaki gibi yer verilmiştir. Federe öğrenme yönteminin avantajları ve dezavantajları da tez kapsamında incelenmiştir.

1.2. Tezin Kapsamı

Federe öğrenme kavramlarının ele alındığı bu tezde, federe öğrenmenin ne olduğu ve nasıl çalıştığı açıklanmıştır. Federe öğrenme kavramının anlaşılması için geleneksel ve dağıtık makine öğrenmesi karşılaştırılmıştır. Federe makine öğrenmesini uygulayacağımız sistemde, cihazların kendi verileriyle, modelini eğitmesi için kullanılan EVRİŞİMSEL sinir ağı açıklanmıştır. ESA'nın çalışma yöntemi kapsamında en uygun şekle sokma (optimizasyon), kontrollü iniş, aktivasyon fonksiyonları, öğrenme katsayısı, ileri geri yayılım, paket boyutu gibi temel kavramlar da ele alınmıştır. Federe öğrenme sistemi ayrıntılı olarak ele alındıktan sonra, FÖ zorluklarından olan iletişim maliyeti, veri düzensizliği, sistem heterojenliğini ve istatistiksel heterojenlik; yöntemlerinden olan FEDSGD FEDAVG,FEDAVGM ele alınmıştır. Deneylerde, ESA temel kavramlarının FÖ sistem performansına etkileri incelenmiştir. FEDAVG,FEDAVGM yöntemleri karşılaştırılmıştır. Deneyleri uygulama sırasında, MNIST veri seti ile çalışmalar gerçekleştirilmiştir. MATLAB aracı ile özel bir fonksiyon kullanılmadan sistemin mantığından yola çıkılarak algoritma oluşturulmuştur.

Federe öğrenmede, sistemin gelişmesinin sağlanması için zorluklarla ilgili daha çok çalışma yapılması gerektiği düşünülmektedir. Zorlukların üzerine çalışmalar arttıkça, federe öğrenmenin gelişmesi sağlanabilir. Günümüzde federe sistemin geliştirilmesi gereken noktalarla ilgili olarak çalışmaların arttırılması, federe öğrenmedeki yöntemlerin çeşitliliğini arttırmaktadır. Zorluklar için oluşturulan çözümlerin, sistem üzerinde

ilerlemesi gözlemlendikçe, federe öğrenme kullanımının yaygınlaşacağı ön görülmektedir.

1.3. Organizasyon

Tezin organizasyonu aşağıda anlatılan şekilde organize edilmiştir:

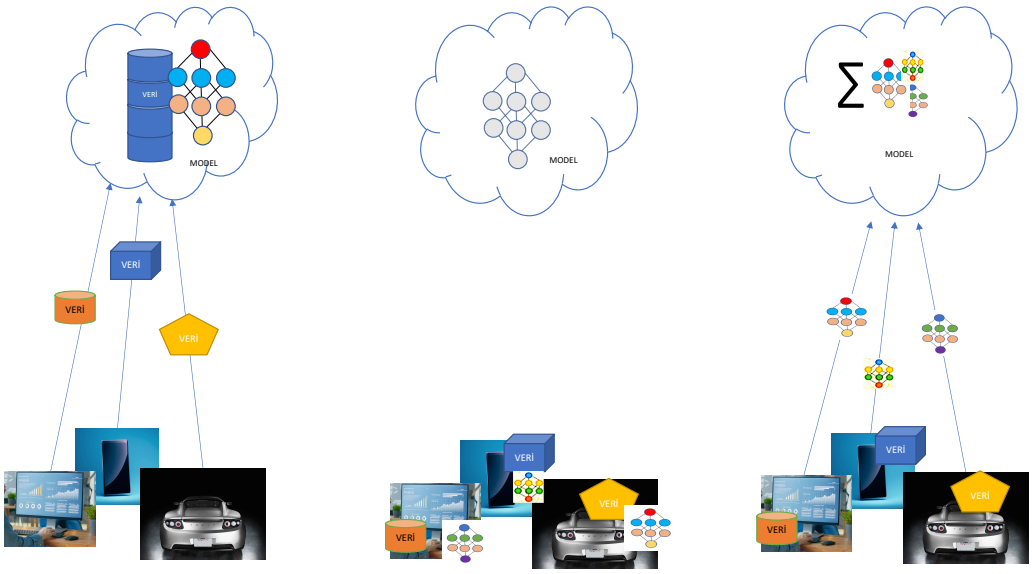
Bölüm 1'de, problemin tanımı, kapsamı ve önemi üzerinde durulmuştur. İlk olarak nasıl ortaya çıktığı, bu yaklaşımla tezde nelerin üzerinde durulacağından bahsedilmiştir. Organizasyon bölümünde de tezin başlıklarının içerikleri tanımlanmıştır. Bölüm 2'de, federe öğrenme yönteminin işleyişinin nasıl olduğu, FÖ avantajları ve dezavantajları ve FÖ'de FEDSGD, FEDAVGM,FEDAVG yöntemlerinden bahsedilmiştir. Federe öğrenme sisteminde istemcilerin yerel olarak eğitimlerini gerçekleştirdikleri ESA ve ESA kavramlarından bahsedilmiştir. Bölüm 3'te, yaptığımızın deneylerin sonuçları doğruluk, kayıp grafiği olarak eklenilmiş ve ilgili açıklamalar detaylıca anlatılmıştır. İkinci kısımda, kod için oluşturduğumuz mimari anlatılmıştır. Federe öğrenme mimarisi için her istemcinin kendi içinde modeli eğitme yöntemi olan EVRİŞİMSEL sinir ağından ayrıntılı olarak bahsedilmiştir. Kodun devamı olan birleşik öğrenme kısmının mimarisi anlatılmıştır. Bölüm 4'te, 3. Bölümde yapılan çalışmaların genel yorumları ve karşılaştırmalarına yer verilmektedir.

2. GENEL BİLGİLER

2.1. Merkezi, Dağıtılmış Yerine ve Federe Makine Öğrenmesi

Makine öğrenmesi, büyük miktardaki veriyi işlemek, veriyi analiz etmek, kullanılabilir hale getirmek amacıyla kullanılan bir araçtır. Veri artışı ile birlikte makine öğrenmesinin kullanımı da yaygınlaşmış, yaygınlaşmaya devam etmektedir. Makine öğrenme, veri kullanımıyla, uygulama alanlarının genişletilmesine, var olanların iyileştirilmesine olanak sağlar. İnternet uygularının iyileştirilip geliştirilmesine, sağlık sektöründe, teşhis konulmasından kullanılan araçlara, otomotivden, eğlence sektörüne, bilime kadar birçok alanda gelişmiş ve geliştirilmeye açık durumdadır. Bu bölümde Merkezi, dağıtılmış yerinde ve federe makine öğrenmesi kavramlarına değinilmiştir. Kıyaslamalar yapılarak, sistemler arasındaki farklar ortaya konulmuştur.

Şekil 2.1'de, soldaki çizim, merkezi makine sisteminin temsili bir ifadesidir. Görüldüğü üzere, veri cihazlardan merkezi bir sunucuya toplanır. Modelin eğitimi toplanmış olan verilerle gerçekleştirilir. Şekil 2.1'de, ortadaki çizim, dağıtık makine öğrenme sisteminin temsili bir ifadesidir. Görüldüğü üzere, veri cihazlarda kalır. Merkezi sunucu modeli cihazlara dağıtır. Cihazlar kendi verisi ile modelini eğitir. Şekil 2.1'de, sağdaki çizim, federe makine öğrenme sisteminin temsili bir ifadesidir. Görüldüğü üzere, veri cihazlarda kalır. Merkezi sunucu modeli cihazların eğitimi için dağıtır. Cihazlar, modeli kendi verisi ile geliştirir. Yerel olarak geliştirilen model, sunucuya gönderilir. Sunucu, modelleri birleştirerek, ortak gelişmiş bir model geliştirmiş olur. Cihazlar en uygun değere yakınsayana kadar bu yineleme gerçekleştirilir [26].



Şekil 2.1. Merkezi /dağıtılmış yerinde/ federe makine öğrenme

2.1.1. Merkezi Makine Öğrenme

Merkezi makine öğrenmede, katılımcılardan gelen tedarik edilen veriler sürekli olarak bir merkeze (sunucu) doğru aktarılır. Merkezde verilerin eğitimi sağlanarak, yeni özellikler oluşturulur, performans iyileştirilmesi yapılmak üzere model eğitilir. Eğitimin sadece merkezde olması hesaplama açısından verim sağlamaktadır [25].

Gelişen teknolojiyle birlikte veri yoğunluğu, boyutu ve hassasiyeti bir hayli artmıştır. Bu durum merkezi öğrenmede bazı eksikliklerin ortaya çıkmasına zemin hazırlamıştır. Verilerin, model eğitimi için katılımcılardan merkeze iletilmesi, artan veri yoğunluğuyla sistemde gecikmelere yol açmakta ve iletim sıkıntıları yaşanmasına sebep olmaktadır. Aynı zamanda, verilerin, model eğitimi için katılımcılardan merkeze iletilmesi, özel verilerin merkezle paylaşılmasına, bu durumun da kişisel gizliliği ihlal etmeye sebep olmaktadır [26]. Sistemin dezavantajlarından yola çıkılarak, merkezi olmayan sistemlere karşı çalışmalar yapılmıştır.

Merkezi öğrenmede, $\mathcal{D}$ sunucunun katılımcılardan topladığı bütün veri seti olup, $\mathcal{D}_k$. k'ıncı katılımcının verisi ve D_k k'ıncı katılımcının veri boyutu olarak ifade edilir. Buna ek olarak, k'ıncı katılımcının yerel veri kümesi şu şekilde ifade edilebilir [27]:

Katılımcı indisi kümesi $\mathcal{K} = \{1, \dots, K\}$ ile gösterilirse, bütün veri seti

$$\mathcal{D} = \bigcup_{k \in K} \mathcal{D}_k. \tag{1}$$

ve toplam veri büyüklüğü

$$D = \bigcup_{k \in K} D_k. \tag{2}$$

ile gösterilir.

$x_k^{(i)}$ ve $y_k^{(i)}$ $\mathcal{D}_k$ 'nın i. elemanının giriş ve çıkış değerlerini ifade etmektedir. Aşağıdaki denklemde bu ifade edilmiştir [27].

$$\mathcal{D}_k^{(i)} = (x_k^{(i)}, y_k^{(i)}) \tag{3}$$

Her bir kullanıcı, kendi yerel veri kümesini kullanarak yerel tahminler yapar. Her bir veri örneği $x_k^{(i)}$için yerel model parametreleri $\boldsymbol{\theta}$ kullanılır. Gerçek hedef değeri $y_k^{(i)}$ olarak gösterilmiştir. $x_k^{(i)}$ değerinin yerel model parametreleri $\boldsymbol{\theta}$ kullanarak, ürettiği tahmin fonksiyonu, $i = 1, \dots, \mathrm{D}_k$ olmak üzere, aşağıda matematiksel olarak şu şekilde gösterilmiştir:

$$f(x_k^{(i)} | \boldsymbol{\theta}) = y_k^{(i)} \tag{4}$$

Her bir kullanıcının yerel modelinin kaybı , yerel tahminler ile gerçek değerler arasındaki farkı ölçen bir fonksiyon olarak hesaplanır. Bu fonksiyon $\mathcal{F}_k(\boldsymbol{\theta})$ ile gösterilir.

Burada genel kayıp fonksiyonu $F(\boldsymbol{\theta})$ ile gösterilmiştir. Aşağıda gösterilen eşitlikte katılımcıların kayıp fonksiyonlarının ağırlıklı ortalaması alınarak genel kayıp fonksiyonu elde edilmiştir:

$$F(\boldsymbol{\theta}) = \sum_{k=1}^{K} \frac{D_k}{D} \cdot \mathcal{F}_k(\boldsymbol{\theta}) \tag{5}$$

(5) numaralı eşitlikle, her cihazın ağırlıklı ortalaması alındıktan sonra (6) numaralı eşitlikte kayıp fonksiyonu minimum değerine getirilir. Aşağıdaki eşitlik, F($\boldsymbol{\theta}$) fonksiyonunun en küçük değerine ulaşan değerini temsil etmektedir. $\boldsymbol{\theta}^*$ optimal parametre değerlerini ifade eder [27].

$$\boldsymbol{\theta}^* = arg\min_{\boldsymbol{\theta}} F(\boldsymbol{\theta}) \tag{6}$$

(7) ve (8) numaralı eşitliklerde regresyon ve sınıflandırma için kayıp fonksiyonları yer almaktadır. Regresyon için kayıp fonksiyonu aşağıdaki gibi ifade edilmektedir:

$$\mathcal{F}_k(\theta) = \frac{1}{D_k} \sum_{i=1}^{D_k} ||f(x_k^{(i)}|\boldsymbol{\theta}) - y_k^{(i)}||^2 \tag{7}$$

Sınıflandırma için kayıp fonksiyonu modelin tahminini ve gerçek değerleri arasındaki farka bakar. Aşağıdaki eşitlikte, $y_k^{(i)} = 0$ ya da 1 değerini alır ve kayıp fonksiyonu aşağıdaki gibi ifade edilmektedir [27]:

$$\mathcal{F}_k(\theta) = \frac{1}{D_k} \sum_{i=1}^{D_k} [y_k^{(i)} \cdot lnf(x_k^{(i)}|\boldsymbol{\theta}) + \left(1 - y_k^{(i)}\right) \cdot ln\,(1 - f(x_k^{(i)}|\boldsymbol{\theta}))] \tag{8}$$

Federe öğrenme ve dağıtılmış makine öğrenimi merkezi olmayan sistemler olarak sunucu ve katılımcı ilişkisiyle çalışırlar. Merkezi öğrenimdeki veri aktarımının aksi şekilde çalışan federe öğrenme ve dağıtılmış makine öğreniminin çalışmasıyla ilgili olarak açıklamalarda bulunulacaktır.

2.1.2. Dağıtılmış Yerinde Makine Öğrenme

Dağıtılmış makine öğrenimi, büyük boyutlu veri ile işlem yapılmasına olanak veren, katılımcının veriyi paylaşması gerekmeksizin, gerçekleştirilen bir makine öğrenimidir[25]. Merkezi öğrenmeden farklı olarak katılımcı yardımıyla eğitimin gerçekleşmesi merkezi öğrenmedeki veri merkezine olan bağlılığı gidermiştir. Daha verimli ve hızlı bir yapı gelişmiştir. Katılımcılar verilerini iletmek yerine yerel olarak eğitimlerini yapmak için sunucudan modeli edinirler. Daha sonra katılımcılar kendilerindeki verilerle modeli kullanarak eğitimlerini sağlarlar. Modeli kendi verilerine göre bağımsız olarak kişiselleştirmiş olurlar. Veri, yerel katılımcıdan hiç ayrılmadığı için gizlilik ihlalinin olasılığı düşürülmüş olur. Fakat model sadece tek bir katılımcının verisiyle, diğer katılımcılardan bağımsız geliştirilmiş olur ve diğer katılımcıların etkisi yardımıyla gelişmeye devam etme süreci durmuş olur [26].

2.1.3. Federe Makine Öğrenme

2.1.2 bölümünde anlatılan dağıtılmış yerinde makine öğrenmesiyle çok benzer olan federe öğrenmesinin dağıtılmış yerinde makine öğrenmesinden farkına değinilerek federe öğrenme sisteminin avantajları ortaya konulabilir. Dağıtılmış yerinde makine öğrenmesinde model, merkez sunucudan katılımcılara gönderilir ve her bir katılımcı uygun bir değere yakınsayana kadar kendi eğitimini tamamlar. Diğer katılımcılarla herhangi bir bağlantısı olmaz ve onların modelinden etkilenmez. Her bir katılımcı bağımsız olarak modelinin eğitimini yerel veri ile tamamlar. Federe öğrenmede ise diğer katılımcılarla ortak bir model oluşturarak modeli geliştirmeyi amaçlarlar. Veri, dağıtılmış yerinde makine öğrenmesindeki gibi katılımcıdan ayrılmaz, model sunucudan alınır ve yerel olarak kendi belirledikleri epoch saylarıyla eğitilirler. Bundan sonra sunucu tarafından toplanılan yerel modeller seçilen yönteme göre birleştirilerek global bir model oluşur. Daha sonra bu model katılımcıların eğitimini iyileştirme amacıyla katılımcılara tekrar dağıtılır [34]. Yineleme işlemi model yakınsayana kadar tekrar edilir.

2.2. Evrişimsel Sinir Ağları

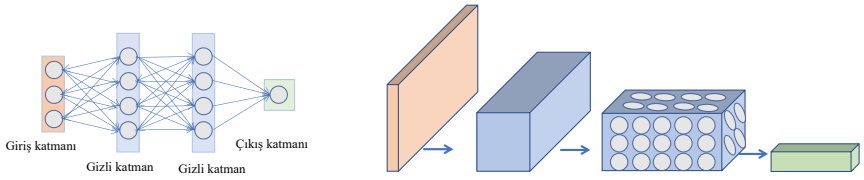
Tez kapsamında çalışılan ve nümerik deneyleri yapılan federe öğrenme yapılarında kullanılan sinirsel ağ mimarisi ESA olmuştur. Federe öğrenmenin çalışma yapısını ve sistem parametrelerine hassasiyetini açıklayabilmek için ESA ile ilgili bazı temel bilgilerin verilmesi gerekli olmuştur. Bu bölümde bu temel bilgiler özetlenmiştir.

Sinir ağlarının bir türü olan evrişimsel sinir ağları, 1988 yılında Yann LeCun ile ortaya çıkan ESA'nın ilk basamağı atılmış oldu. İlk ESA, el yazısı posta kodlarındaki rakamları, geri yayılım yoluyla başarılı bir şekilde sınıflandırmıştır. Bilgisayar görü çalışmalarında kullanılan derin öğrenme mimarisi olan ESA, aynı insan beynindeki görsel korteks gibi çalışır. Görsel korteks, beyinde bulunan bir bölge olup, görsel bilgilerin işlendiği ve algılandığı yerdir. Girdi olarak verilen bir resmin ne olduğunu istatistiksel olarak çıktı almamızı sağlar.

Evrişimsel Sinir Ağları, sıradan sinir ağlarındaki gibi "weight" (ağırlık) ve "bias" (saptırma) terimlerinden oluşurlar. Ağırlık ve saptırma terimleri, makine öğrenmesi ve derin öğrenme modellerinde kullanılan modelin karmaşıklığını ve esnekliğini sağlamak için kullanılan ayarlanabilir parametrelerdir. Ağırlıklar, modelin her bir giriş özelliğinin önemini belirlerken, saptırma, bir modelin öğrenme sürecinde çıktıyı kaydırmak veya saptırmak için kullanılmaktadır. Temel olarak sistemin bir girdisi vardır ve iç çarpım gerçekleştirerek bir çıktı elde etmeyi amaçlamaktadır. Öğrenme sürecinde geri yayılım kullanır, geri yayılım hata fonksiyonunun ağırlıklara göre türevlerini hesaplayarak çıktıdan girdiye doğru geriye çalışır ve hata fonksiyonunun minimize edilmesinde görev almaktadır.

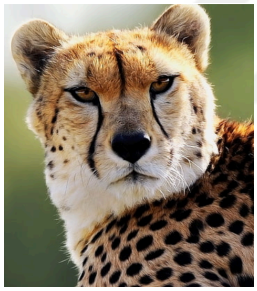
ESA, sıradan sinir ağlarından farklı olarak daha az parametreye ihtiyaç duyar ve bu sayede daha hızlı hale gelir. Ayrıca ESA, görsellerin orijinal boyutlarını koruyarak verimli bir şekilde çalışabilirken, sıradan sinir ağları (SSA) orijinal görsel yerine ölçeklenmiş görseller kullanmak durumunda kalırlar. ESA'larını SSA'dan ayıran temel özelliklerden biri de ESA'nın mimarisinde bulunan evrişimsel ve ortaklama katmanının çıktısının 2 boyutlu olmasıdır [10].

Şekil 2.2'de soldaki görsel bir girdi, iki gizli ve bir çıktı olarak 3 katmandan oluşan, ileri beslemeli sıradan sinir ağıdır. Şekil 2.2'de sağdaki görsel, 3 boyutlu bir giriş katmanından oluşan, evrişimsel sinir ağıdır. Kırmızı renk ile girdi olarak verilen, görsel ifade edilir. Genişlik ve yüksekliğin görselin boyutlarını, derinlik ise kanal sayısını (kırmızı, yeşil, mavi) içermektedir.



Şekil 2.2. Sıradan yapay sinir ağları/ evrişimsel sinir ağı

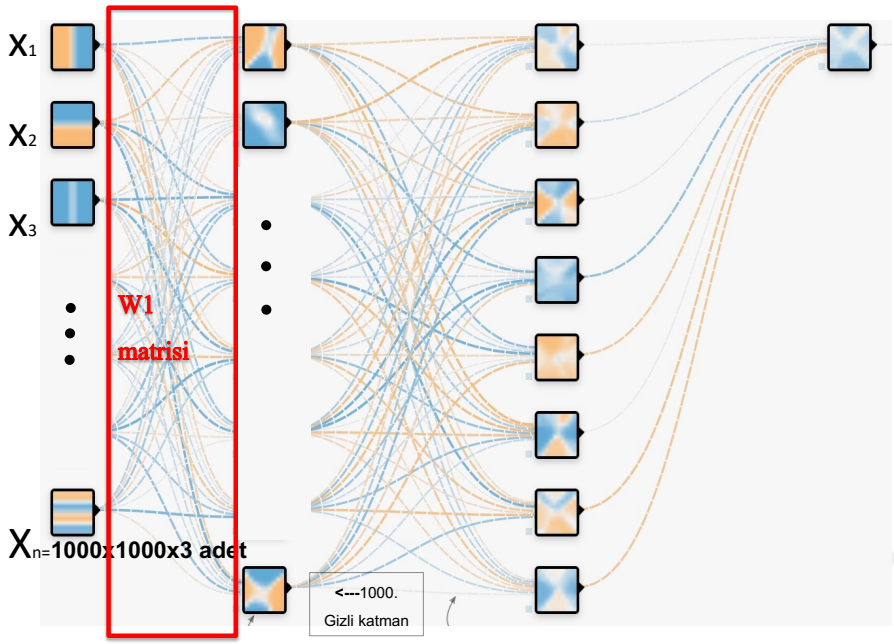
Görüntü hesaplama işleminin SSA ile hesaplamanın zorluklarından biri de görüntü boyutlarının çok büyük olabilmesidir. Şekil 2.3. numaralı görselde yüksek kaliteli bir çita görüntüsü vardır:



Şekil 2.3. ESA için örnek bir girdi

Düşük kaliteli fotoğraflar bir şekilde hesaplanabilir olsa da yukarıdaki şekil 2.3'teki görüntüdeki çitanın boyutları 1000x1000x3 olabilir. Bu 3 milyon girdi özelliğinin olduğu anlamına gelir.

Şekil 2.4'te, bahsedilen X_n 3 milyon boyutlu bir giriş olduğundan bahsedebiliriz. Gizli katman ise 1000 adet olduğunu varsayarsak w1 matrisi (1000,3M) $x \in \mathbb{R}^{3M}$ boyutlu yani 3 milyar olmaktadır. Bu kadar çok parametre ile ezberlemeyi önlemek için yeterli veriyi kullanmak zor olduğu gibi hesaplama ve yeterli bellek yönetimi de zor olacaktır. Bu sebeple görüntü işlemelerinde konvolüsyon yapmak daha mantıklıdır [12]. İlerleyen paragraflarda bu konudan bahsediyor olacağız.



Şekil 2.4. Görsel için oluşturulan SSA yapısı

2.2.1. Evrişimsel Sinir Ağlarında Kullanılan Kavramlar

Bu bölümde ESA yapısı oluşturulurken kullandığımız kavramlara yer verilmiştir. Bu kavramlardan "Deneysel Çalışmalar" bölümünde deney sonuçlarına yorum yaparken ve deney sırasında yararlanılmıştır.

2.2.1.1. Evrişimsel Katman

Evrişimsel katmanı genişlik ve yükseklik olarak girdi matrisinden daha küçük olur. Derinlik olarak ise, evrişimsel katmanın filtre sayısı, girdi matrisinin derinliği ile örtüşür. Evrişimsel filtre kadar özellik haritası oluşur. Özellik haritası, her bir filtre tarafından yakalanan özelliklerin çıktısını temsil eder. Her bir filtre, giriş matrisinin genişlik ve yüksekliği boyunca hareket ettirilerek, filtreyle o anki konumdaki alanın çarpımı sağlanır. Bu işlem ile özellik haritası olan matris elde edilir. Bir filtre ile bir özelliği tespit edebilirken birden fazla özellik için daha fazla filtre kullanılabilir. Bu özellikler de sınıflandırma işlemi için girdi sağlayabilir. Bu katmanın çıktısı başka bir katmanın girdisi olabilir. Örneğin, insanların yüzlerindeki mimikler incelenip, kızgın yüzlere sahip olan insanları seçilmek istenirse, özellik haritasında kaşların, dudakların yapısına bakılarak

öğrenme gerçekleştirilir. Ayırt etmeyen özellikler için, örneği kulak, inceleme yapılmazken, farklılık gösteren bölgelerin özellik haritaları araştırılır.

Adım sayısı (stride): Evrişimsel katman anlatılırken, filtrenin giriş matrisinin genişlik ve yüksekliği boyunca hareket ettirildiğinden bahsetmiştik. Hareket ettirilirken, matrisin üzerinden kaç birim kaydırılacağına karar veren sayı adım sayısı olarak adlandırılır.

Dış boşluk sayısı (padding): Filtre ve giriş matrisinin kaydırma işlemi sonucu elde edilen matris boyutu giriş matrisinden daha küçük boyutlu olur. Görüntünün bazı özellikleri kaybolabilir. Bu durumu önlemek amacıyla giriş matrisinin çevresine ekstra sıfır ekleme işlemine dış boşluk denilir. Çevresine eklediğimiz sıfır sayısı da dış boşluk sayısı olarak adlandırılır [10] [11].

2.2.1.2. Ortaklama Katman (Pooling Layer)

Ortaklama katmanı (pooling layer), evrişimsel sinir ağlarında kullanılan bir katmandır. Bu katman, özellik haritasındaki öznitelikleri özetlemek ve boyutunu azaltmak amacıyla kullanılır. Matrisleri toplama şeklimize göre havuzlama yöntemi seçilir. Bütün terimler toplanılırsa “sum ortaklama”, bütün matrislerin ortalamaları alınırsa “mean ortaklama” ya da en büyük değerler alınırsa da “max ortaklama” yapılmış olunur.Böylece, ağın daha genel ve özgül olmayan özellikleri öğrenmesi sağlanırken hesaplama yükü azaltılır ve aşırı uyum (overfitting) riski azaltılır.

2.2.1.3. Düzleştirme Katman (Flattening)

Düzleştirme Katmanı (flattening), evrişimsel sinir ağlarında kullanılan bir katmandır. Bu katmandan sonraki hesaplamalar için bundan önce olan hesaplanmış çok boyutlu matrisin, 1 boyutlu olarak dönüştürülme işini kapsamaktadır [13]. Düzleştirme katmanı, tam bağlantılı katmana girdi oluşturur.

2.2.1.4. Tam Bağlantılı Katman (Fully Connected Layer)

Bu katmandan önce düzleştirme katmanıyla matris tek boyutlu vektör haline dönüştürülmüştür. Düzleştirilmiş vektörün tam bağlantılı katmana girdi sağlamasıyla, bu katmanda matematiksel işlemlerden geçer. Sınıflandırma işi bu katmanda olur. Tam bağlantılı katmanın nihai çıktı sayısı, sınıflandırma yapılan sayı ile aynı olur. Bu katmanın sonunu aktivasyon fonksiyonunu takip eder [15]. Tam bağlantılı katman ile önceki katmanın tüm özelliklerini son katmana aktarırız ve çıkışı oluşturmasında etkili olduğunu gözlemleyebiliriz.

2.2.1.5. Paket Boyutu

Sinir ağlarında , veri ile modelin öğrenmesinde gerçekleştirilen her yinelemede ileri ve geri yayılım uygulanır. Bu konuyla ilgili ayrıntı 2.2.1.9. numaralı bölümde verilmiştir. Hesaplama yoğunluğunun fazla olması sebebiyle verinin baştan sona tek seferde hesaplanması zor olur. Bu sebeple her yinelemede veri seti küçük paketlere ayrılır. Örneğin, öğrenime katılmış olan cihazda 2000 adet verimiz olsun. Paket boyutu 100 olursa, her bir epoch için tüm veri setinin üzerinden geçebilmek için 20 adet yineleme yapılır. Birinci yineleme, birinci paketin (100 adet veri) model üzerinden geçirilmesi anlamına gelir. İkinci yineleme, ikinci paketin (100 adet veri) model üzerinden geçirilmesi anlamına gelir.

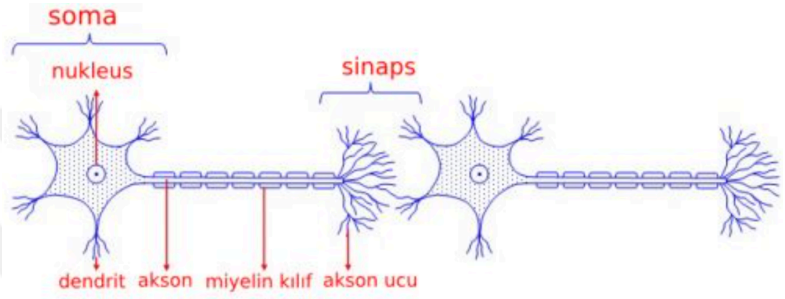
Eğer paket boyutu veri setiyle aynı boyuta sahip olursa bu paket gradyan iniş adını alır. Her yinelemede sistem çok büyük boyutlu veriyle baş etmek durumundadır. Paket gradyan iniş, veri seti boyutu çok büyük değilse kullanılabilir. Fakat, büyük veri boyutları için yineleme süreleri çok uzayabilir.

Eğer ~~mini~~ paket boyutu 1 ise bu paket stokastik gradyan iniş adını alır. Her mini paketteki maliyet aynı olmayacağı için grafiğin doğrusal değil de gürültülü olması muhtemeldir. Yüksek gürültü de öğrenme oranıyla ayarlanabilir. Fakat, vektorizasyon bütün hızımızı kaybedilmesine sebep olur. Örneğin, cihazda 2000 adet verimiz olsun. Paket boyutu 1 olursa, 1. epoch sayısında tüm veri setinin üzerinden geçebilmek için 2000 adet yineleme yapılır.

Bu sebeple paket boyutu seçimi var olan veri sayısına ve bellek kullanımını etkileyeceği için aynı zamanda CPU/GPU bilgisine bağlıdır [24].

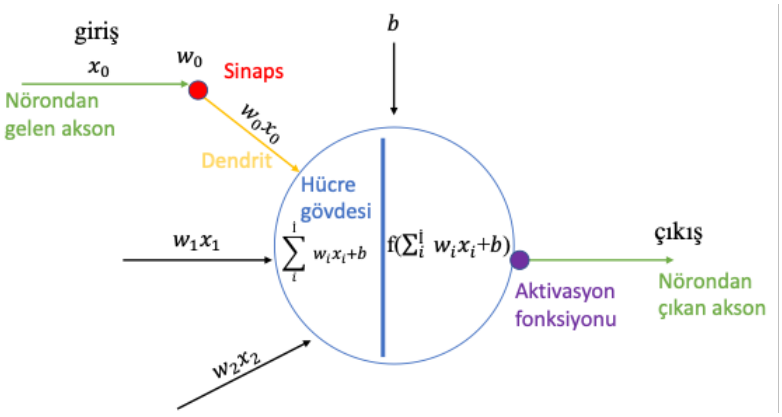
2.2.1.6. Aktivasyon Fonksiyonu

Şekil 2.5.'teki görsel, temel hesaplama birimi olan nöronun yapısını içermektedir. Bir nöronun çalışma yapısı; dendritlerden giriş sinyalleriyle başlayıp, sinyalin hepsi hücrenin gövdesine taşınır. Toplanılan sinyal bir eşiğin üstündeyse akson kısmına iletim sağlanır. Sinyalin aksonlardan iletilmesinden sonra sinaps yardımıyla diğer nöronun dendritine iletilir.



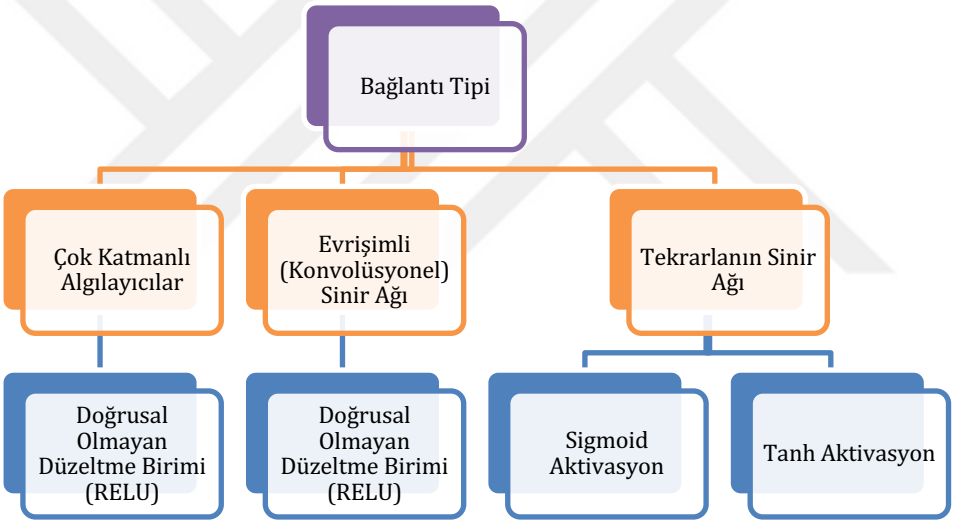
Şekil 2.5. İnsanlardaki sinir ağı yapısı

Şekil 2.6.'daki görsel, aktivasyon fonksiyonun matematiksel gösterimini içermektedir [48]. Sinyal (Xo), aksonlar yardımıyla ilerlerken, sinapstaki etkisinin gücüyle (Wo) diğer nöronun dendrit kısmıyla etkileşime girerler (Xo*Wo). Buradaki sinaps gücü öğrenilebilir bir parametre olup diğer nöron üzerinde bir etkisi olmaktadır. Matematiksel görselde aksona iletimin sağlanması aktivasyon fonksiyonu ile sağlanmaktadır. Nöronlarda belirli bir eşiğin üstüne çıkması kontrol edildikten sonra aksona iletilir. Matematiksel olarak ifade edildiğinde eşiğin üstüne çıkma kontrolü aktivasyon fonksiyonu yardımıyla yapılır. Sonuç olarak aktivasyon fonksiyonu, gerçek değerleri alıp belirli bir aralık değerine getirme görevini yerine getirir [20].



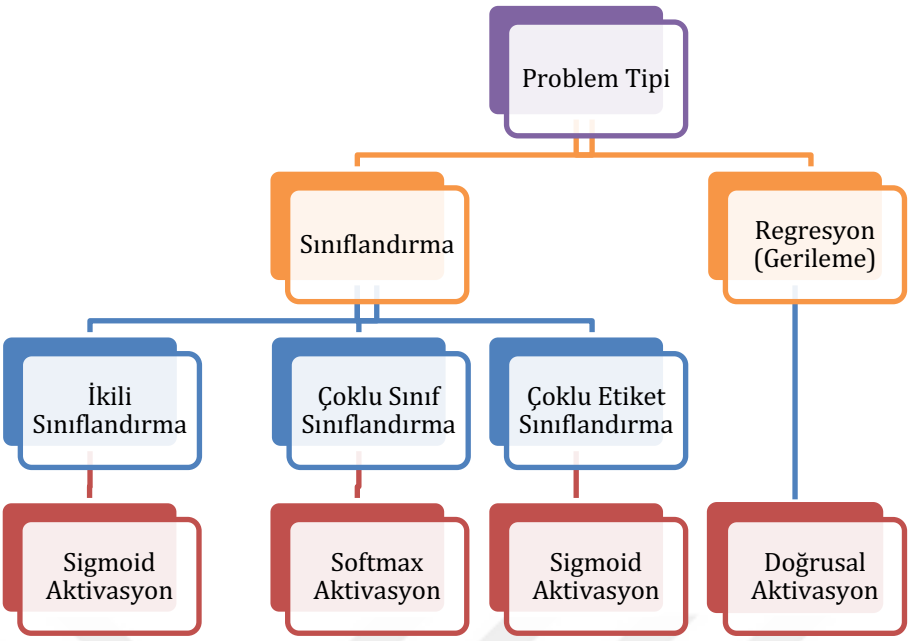
Şekil 2.6. Yapay sinir ağlarındaki sinaptik iletim

Şekil 2.7 gizli katmanlar için aktivasyon fonksiyonlarını tipik olarak nasıl seçildiği özetlemektedir [21].



Şekil 2.7. Aktivasyon fonksiyonunu bağlantı tipine göre sınıflandırma

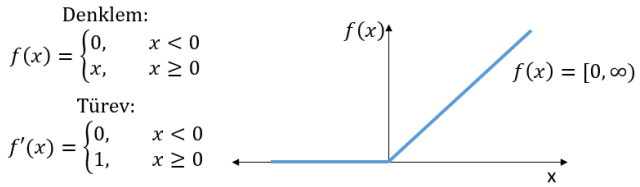
Şekil 2.8 çıkış katmanı için aktivasyon fonksiyonlarının nasıl seçildiği özetlemektedir [21]:



Şekil 2.8. Aktivasyon fonksiyonunu problem tipine göre sınıflandırma

Doğrultulmuş Doğrusal Birim (Relu): Gizli katman, bir katmandan girdi alıp başka bir katmana girdi sağlayan katmandır. Sinir ağlarındaki doğrultulmuş doğrusal birim (Relu) genellikle gizli katmanda aktivasyon fonksiyonu olarak kullanılır. Aktivasyon fonksiyonları, sinir ağındaki her bir düğümün (nöron) çıkışını belirlemektedir. Doğrultulmuş doğrusal birimin, diğer fonksiyonlara kıyasla daha basit bir yapısı olmasının yanı sıra stokastik gradyan inişinin yakınsamasında daha hızlıdır [20]. Türev işlemi vardır ve bu geri yayılıma izin verir. Geri yayılım algoritması, hatayı geriye doğru yayarak ağırlıkları güncellerken türevlerden yararlanır.

Şekil 2.9'da Relu fonksiyonun sağda grafiği, solda ise matematiksel denklemi ve türevi yer almaktadır:



Şekil 2.9. Doğrultulmuş doğrusal birim denklem ve grafik gösterimi

Softmax: Son katmanda uygulanılan aktivasyon fonksiyonu seçilirken amaca göre seçim yapılır. Örneğin çoklu sınıflandırma için model eğitiliyorsa, çıkış katmanında softmax

yönteminin kullanılması uygundur. Çıktıları 0 ve 1 arasında bir değer olur ve çıktıların toplamı da 1 değerine eşit olur. Bu sayede çıktı değerleri oranlanarak olasılık değerine dönüştürerek karar vermemizde etkili olur.

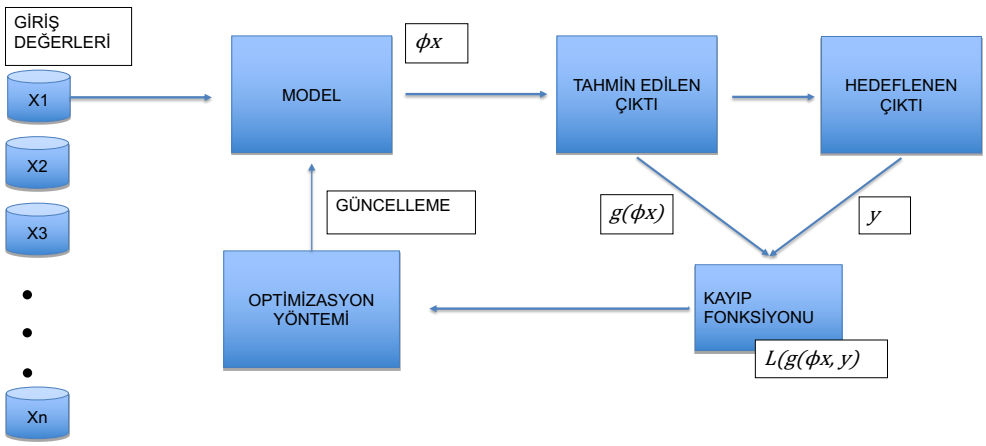
Softmax aktivasyonu için matematiksel denklem aşağıda verilmiştir [40]. Paydadaki terim Normalleştirme terimi olup çıktı dağılımının (0,1) arasında olmasını sağlar. Softmax aktivasyonu sonucunda x_i değeri ise [0,1] aralığında bir değere sabitlenir. i=1,…,J olmak üzere eşitlik (9)'da softmax matematiksel olarak gösterilmiştir:

$$f(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{J}(e^{x_j})} \tag{9}$$

2.2.1.7. Optimizasyon Metodu

Bu bölümde, modelin çıktısı ve gerçek veriyle karşılaştırılan fark sonucu ortaya çıkan kaybın nasıl minimize edilip, modele geri dönüş vererek bu kaybın nasıl azaltılabileceği üstünde durulmuştur.

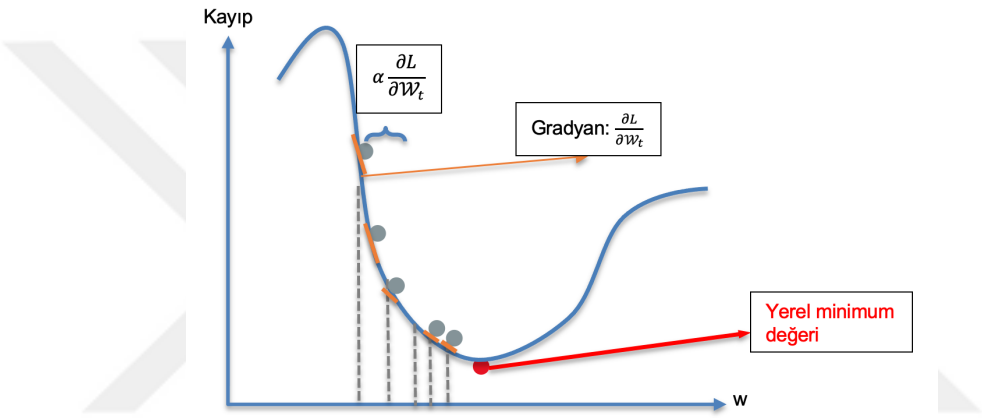
Şekil 2.10'da modelin çıktısı ve gerçek veriyle karşılaştırılan fark sonucu ortaya çıkan kaybın minimize edilip, modele geri dönüş vererek bu optimizasyon metodunun nasıl çalıştığının daha kolay anlaşılması için bir akış görseli vardır.



Şekil 2.10. Optimizasyon yönteminin sistemdeki akış görselleştirmesi

Gradyan iniş (Gİ) makine öğrenme algoritmalarında hata oranın minimize etmeye çalışan optimizasyon yöntemlerinin genel adıdır [18]. Gİ, yerel ve global minimum değerini arar. Şekil 2.11.'daki grafik kayıp-ağırlık grafiğidir. Resimde kırmızı olarak işaretlenen nokta yerel minimum değeri olup kayıp fonksiyonun en düşük değere ulaştığı noktadır. Ağırlık değiştikçe, bir kayıp değeri elde edilir. Gİ ile kırmızı noktayı bulmayı hedefleyerek kaybın minimum olduğu noktaya bulmaya çalışmaktadır.

Gİ ve SGİ arasındaki farka değinirsek, Gİ, bütün öğrenme verisiyle işleme girip ağırlıkları güncellerken, SGİ tek bir veriyle işleme girip ağırlıklarını günceller [13].



Şekil 2.11. Stokastik gradyan inişin görselleştirmesi

En yüksek artış oranı kayıp fonksiyonunun gradyanı yardımıyla öğrenilir. Öğrenme oranıyla çarpılarak eski ağırlık değeri negatif yönde güncellenip, yeni ağırlık değeri (öğrenilebilir parametre) elde edilir. Gİ matematiksel gösterimi, öğrenilebilir parametrenin güncellenmesi aşağıdaki gibi ifade edilmiştir [13]:

$$W_{t+1} = W_t - \alpha \frac{\partial L}{\partial W_t} \tag{10}$$

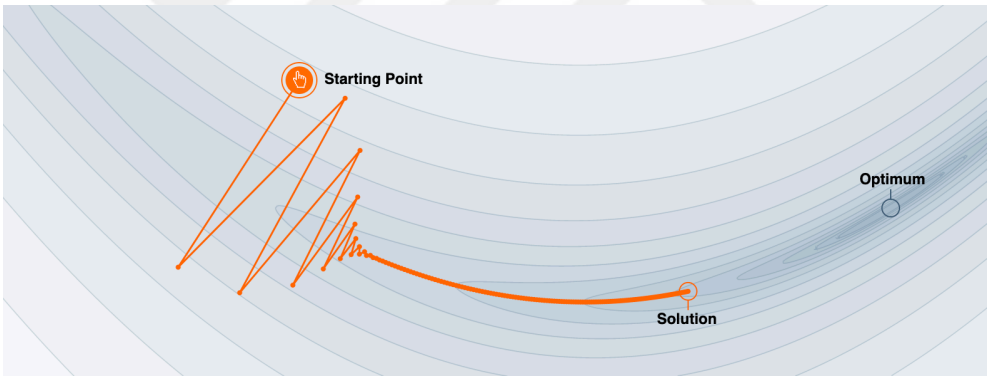
2.2.1.8. Kontrollü İniş ve Öğrenme Katsayısı

Öğrenme Katsayısı ağın eğitilmesi sırasında kullanılan optimizasyon metodu ve geri yayılım, oluşan hatanın en uygun değere getirilmesi için kullanılan parametrelerdir. Optimizasyon ve geri yayılım ile ilgili 2.2.1.7 ve 2.2.1.9 numaralı bölümlerden ayrıntılı

olarak bilgi edinilebilir. Öğrenme katsayısı eğitim sırasında mini paketlerden gelen bilgilerle, eğitim yaptığımızın modelin ağırlıklarının hangi oranda güncelleneceği hakkında sisteme katkıda bulunan bir parametredir [45].

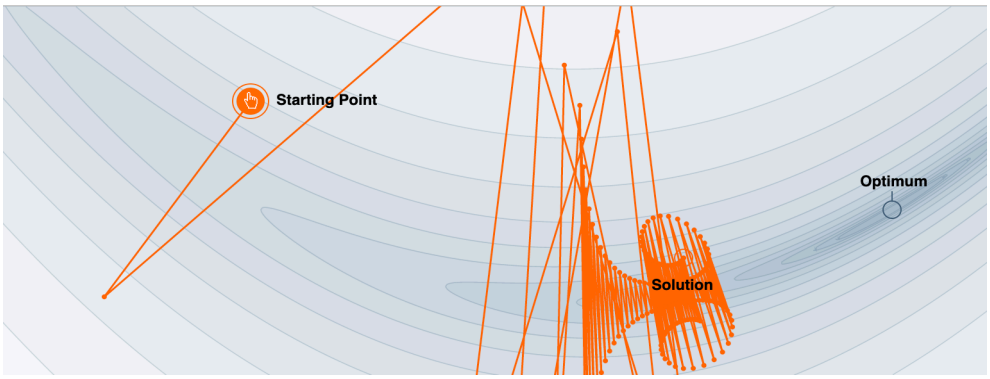
SGİ ile optimizasyon yapıldıktan sonra geri yayılım ile hata azaltılmaya çalışır. Hatayı çıktı değerinde geriye doğru türev alınarak farkın bulunması sağlanır, daha sonra bulunan fark ile öğrenme katsayısı çarpılarak farkın ne kadar etki etmesi gerektiğine öğrenme katsayısı ile karar verilir. Buradaki öğrenme hızının küçük olması, adımları küçük atmamıza ve ilerlemenin yavaş ve emin adımlarla olmasına etki ederken, öğrenme katsayısının büyük olması ise uygun çözümden uzaklaşmamıza, çok fazla salınım yapmamıza, sonucunda da uygun değere ulaşamamamıza sebep olabilir [46].

Momentum: Kontrollü iniş yöntemi (momentum) uygun değer (optimum nokta) aranırken salınımı azaltmak, hedefe gitme hızını arttırmak için kullanılabilir [17] [18].



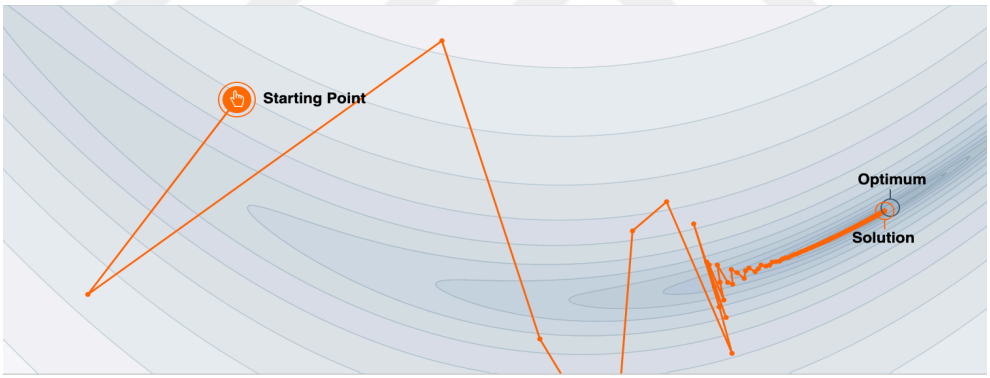
Şekil 2.12. Düşük öğrenme katsayısı görselleştirmesi

Şekil 2.12 numaralı görsel, öğrenme parametresinin düşük olduğu gradyan inişi sembolize eder. Çok fazla adım ile minimum noktasına salınarak gitmeye çalışır. Çok fazla salınım olması optimum noktasına ulaşmayı geciktirir. Bu durum bizi öğrenme parametresini büyütmeye yöneltir. Şekil 2.13 numaralı görselde, görüldüğü üzere öğrenme parametresinin artmasıyla sapmalar artmıştır. Dikey yöndeki bu sapmalar optimuma ulaşmayı yavaşlatmaktadır [19].



Şekil 2.13. Yüksek öğrenme katsayısı görselleştirmesi

Bu sorunlara karşı önerilen yöntem kontrollü iniş ile Gİ uygulamaktır. Momentum ile gradyan iniş adımlarının sertliği azalır, dikey yönde salınımı azaltıp yatay yönde optimuma ulaşmayı hızlandırır. Momentum ile gradyan inişe kısa süreli bir hafıza verdiğimizi düşünebiliriz. Geçmiş gradyanların katılma oranını da β parametresiyle sağlanır. Yüksek öğrenme parametresiyle oluşturduğumuz görsele β parametresi eklenince optimuma yaklaşma şekli şekil 2.13'te verilmiştir:



Şekil 2.14. Yüsek öğrenme katsayısı ve eklenmiş momentum görselleştirmesi

$V_{t=0} = 0$ ve $0 < \beta < 1$ olmak üzere aşağıdaki eşitliklerde (33) ve (34) matematiksel olarak kontrollü inişin açıklamasına olarak yer verilmiştir:

$$w_{t+1} = w_t - \alpha V_t \tag{11}$$

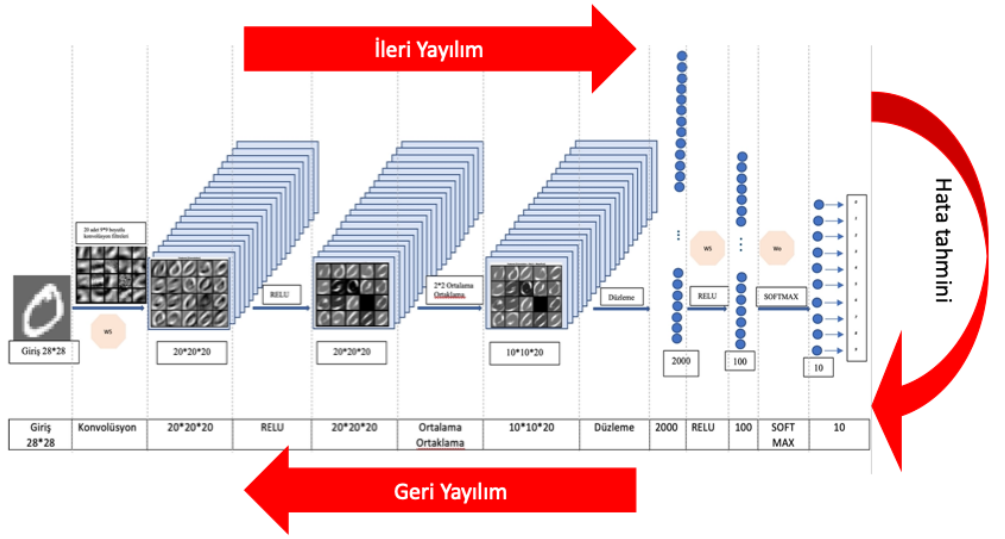
$$V_t = \beta V_{t-1} - (1-\beta)\frac{\partial L}{\partial w_t} \tag{12}$$

2.2.1.9. İleri/Geri Yayılım

Günümüzde ileri geri yayılımın geldiği noktaların zemini 1970'li yıllarda David Rumelhart ile temeli oluşturuldu fakat tam olarak beklenilen ilgi gösterilmedi. Hinton, Rumelhart ve Williams'ın [39] makalesinde geri yayılım konusu ele alınmış ve önemi o günden günümüze kadar gelmiştir.

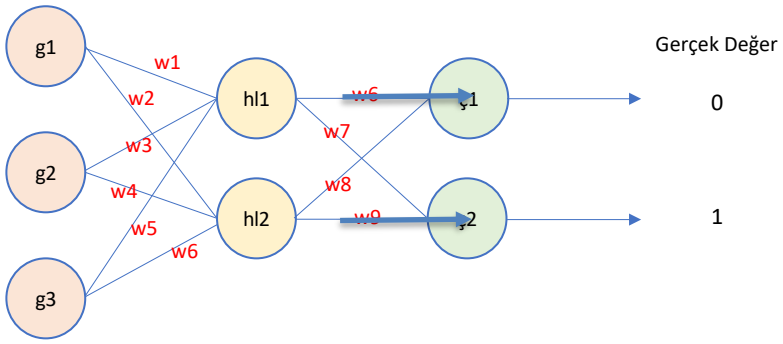
Geri yayılım algoritması, sinir ağlarında eğitim sırasında eğitilen çıktı ile gerçek çıktı değeri arasındaki çıkan farkı minimum değere getirmek için uygulanan bir tekniktir. Geri yayılım tekniğindeki genel mantık ağırlık değerlerinin her yinelemede geri yayılım yoluyla güncellenerek hatayı minimum değerine getirmeyi amaçlayarak eğitimin doğruluk grafiğini en yüksek değere yakınsarken, kayıp grafiğini en küçük değere yakınsamayı hedeflemektedir [39].

Şekil 2.15 yapılan deneylerin ESA mimarisi olup, ileri/geri yayılımın yönlerini göstermek amaçlı eklenmiştir:



Şekil 2.15. Konvolüsyon Sinir Ağlarında İleri/Geri Yayılım

Şekil 2.16'da basit bir sinir ağı görseli mevcuttur. 3 adet öznitelik girişi olan sinir ağın 2 adet gizli katmanı ve 2 adet sınıflandırma yapan çıktısı mevcuttur. Aşağıdaki örnek üzerinden ileri geri yayılım olayını inceleyebiliriz.



Şekil 2.16. Yapay sinir ağlarında katman ve ağırlıkların görselleştirilmesi

İleri yayılım: Girdi değerleri veri setinden alınıp öznitelik değerleri olmaktadırlar. Ağırlık değerleri için modelde ilk başta 0-1 arasında herhangi bir değer atanabilir. Çıkış değerini oluşturmak için girdi ve katman değerleri çarpılarak aktivasyon fonksiyonu uygulanır. Daha sonra sonuç çıktı katmanı ve ara katmandaki ağırlık değerleriyle çarpılarak çıkış değeri elde edilmeye çalışılır. Eğer daha fazla ara katman varsa ara katmanların ağırlık değerleri ve bir önceki katmandan gelen değer çarpılarak çıkış katmanına doğru ulaşılmaya çalışılır. Aşağıda, anlatılanlar matematiksel olarak ifade edilmiştir. [41]

Eşitlik (13) ve (14)'te giriş öznitelik değerleri başlangıç olarak belirlenen ağırlıklarla çarpılarak gizli katman (hl1-hl2) değerleri elde edilmiştir:

$$hl1 = g1 \cdot w1 + g2 \cdot w3 + g3 \cdot w5 \tag{13}$$

$$hl2 = g1 \cdot w2 + g2 \cdot w4 + g3 \cdot w6 \tag{14}$$

Ağırlık ve girdi vektörünün çıktısı olan hl1 ve hl2 değerleri eşitlik (15)'te fonksiyonu yazılmış olan softmax aktivasyonuna girdi olur:

$$f(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{J}(x_j)} \quad (15)$$

Ağırlık ve girdi vektörünün çıktısı olan hl1 ve hl2 değerleri yukarıda eşitlik (15)'te fonksiyonu yazılmış olan softmax aktivasyon fonksiyonuna girdi olduktan sonraki çıktılar hl1o ve hl2o olarak aşağıdaki eşitlik (16) ve (17)'de belirtilmiştir:

$$hl1o = \frac{e^{hl_1}}{\sum_{j=1}^{2}(x_j)} \quad (16)$$

$$hl2o = \frac{e^{hl_2}}{\sum_{j=1}^{2}(x_j)} \quad (17)$$

Eşitlik (16) ve (17)'de giriş öznitelik değerleri başlangıç olarak belirlenen ağırlıklarla çarpılarak gizli katman(hlo1-hlo2) değerleri elde edildikten sonra gizli katman ve çıkış değerleri arasında bulunan ağırlık değerleriyle çarpılarak aktivasyon fonksiyonuna girmemiş olan çıkış değerleri elde edilmiştir:

$$ç1o = hlo1 * w6 + hlo2 * w8 \quad (18)$$

$$ç2o = hlo1 * w7 + hlo2 * w9 \quad (19)$$

Softmax aktivasyon fonksiyonuna, yukarıda eşitlik (18) ve (19)'un çıktıları uygulanarak (ç1o ve ç2o değerleri) ç2 ve ç1 değerleri yani eşitlik (20) ve (21) hesaplanır:

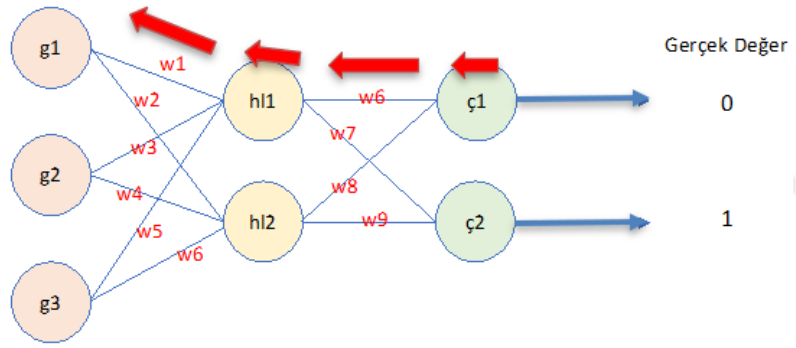
$$ç2 = \frac{e^{ç2o}}{\sum_{j=1}^{2}(x_j)} \quad (20)$$

$$ç1 = \frac{e^{ç1o}}{\sum_{j=1}^{2}(x_j)} \tag{21}$$

Elde edilen çıktı değerleri ile gerçek çıktı değerleri arasında sayısal bir farklılık olacaktır. Bu sayısal farklığın karesinin toplamı bize hata fonksiyonunu vermektedir:

$$Hata\ fonksiyonu = \sum \frac{1}{2} \cdot (gerçek\ değer - çıktı)^2 \tag{22}$$

Elde edilen hatayı minimize etmek amacıyla ileri yayılımdan sonra ağırlıkları değerlerini daha doğru değerler elde etmek için geri yayılım yaparak çıkış katmanında oluşan hata en küçük değere yakınsatmaya çalışılır.



Şekil 2.17. Sinir ağlarında geri yayılım

2.2.1.10. Epoch Sayısı

Bir makine öğrenme modelinin eğitim sürecinde, tüm veri setinin, epoch sayısı kadar tamamlanmasıdır. Veri seti, modelin üzerinde öğrenme işlemlerinin gerçekleştirildiği veri kümesidir. Bir epoch, eğitim veri setindeki tüm örneklerin model tarafından bir kere işlenmesini ifade eder. Eğitim süreci genellikle birden fazla epoch üzerinde gerçekleştirilir. Bu durum, modelin eğitim setini birden fazla görmesine olanak tanıyarak performansının artmasını sağlayabilir.

2.3. Federe Öğrenmenin Temelleri

Bu bölümün altındaki üç bölümde anlatım akışı şu şekilde olacaktır. Bölüm 2.3.1'de, federe öğrenmenin nasıl çalıştığı matematiksel eşitliklerle anlatılmıştır. Federe öğrenmenin akışı görsel ve akış şemasıyla desteklenmiştir. Bölüm 2.3.2 ise federe öğrenmede karşılaşılan zorluklara yer verilmiştir. Bölüm 2.3.3 ise sunucuda birleştirilmesi sırasında uygulanılan yöntemlerin sınıflandırılması ve bu yöntemlerden olan FEDSGD, FEDAVGM ve FEDAVG yöntemleri yer alacaktır.

2.3.1. Federe Öğrenmenin Çalışma Temelleri

Federe öğrenme, bir çeşit özelleşmiş dağıtık makine öğrenmesidir. Bölüm 2.1'de merkezi, yerinde dağıtık ve federe öğrenmenin farklarından bahsedilmiştir. Bu bölümde çalışma şekli ayrıntılı olarak ele alınmıştır.

Federe öğrenmede veri cihazdan ayrılmaz. Federe öğrenmesi merkezi bir sunucu denetiminde katılımcılarla gerçekleştirilir. Katılımcılar ilk modeli sunucudan elde eder. Kendi verileri ile sunucudan gelen modelin eğitimini sağlarlar. Bu tez kapsamında ESA kullanılarak katılımcıların öğrenmesi sağlanmıştır.

Cihazların kendi verileri ile eğittiği model sunucu tarafından birleştirilir. Birleştirilme sırasında özel yöntemler kullanılır. Birleştirme yöntemlerinin tipik bir sınıflandırılmasına bölüm 2.3.3'te yer verilmiştir. Sunucu, bütün cihazlardan topladığı, yerel model parametrelerini kullandığı birleştirme yöntemine göre birleştirir. Model güncelleştirilerek, yeni oluşturulan model, sistemin daha iyi gelişmesini daha iyi öğrenmesini sağlar. Yeni oluşturulan model, katılımcılara tekrardan dağıtılır. Katılımcılar daha gelişmiş bir model üzerinden verilerini kullanarak modeli geliştirir. Bu eğitim süreci, doğruluk oranı optimum değere ulaşana kadar, değer uygun bir değere yakınsayana kadar yinelemeli olarak devam eder.

Federe öğrenmede "1 yineleme" terimi, federe öğrenme sürecinin ilk turunu ifade etmektedir. Bir yineleme şu adımları içerir: Katılımcılar kendi modelini güncelleştirir.

Güncellenen model sunucuda birleştirilir. Sunucu, katılımcılardan gelen modelleri birleştirerek var olan modeli günceller. Güncellenen model tekrardan katılımcılara dağıtılır. Bu süreç eğitimin ilk turudur. Model optimum değere yakınsayana kadar bu yineleme devam eder [1] [9].

Katılımcılar modelin bir parçası olup, modelin güncellenmesine katkı sağlamaktadırlar. Verilerin boyut ve sayı olarak büyüyerek geliştiği yeni teknolojilerde federe öğrenme ile modelin doğruluğunun artması beklenmektedir. Çünkü veri sayısı ve çeşitliliği arttıkça model kendini sürekli güncelleyecektir. Bu durum modellerin kullanılabilirliğini ve güvenilirliğini arttırabilir. Federe makine öğrenmesi kişisel verilerin, katılımcıdan ayrılmadan eğitimi gerçekleştirmesi ile verilerin gizliliğini büyük oranda sağlamaktadır. Bu durum ile veri gizliliğinde federe öğrenme ön plana çıkmaktadır [8].

$\mathcal{D}_k$ k'ıncı katılımcının verisi, D_k k'nıncı katılımcının veri boyutunu ifade eder [27] Katılımcı indisi kümesi $\mathcal{K} = \{1, \dots, K\}$ ile gösterilirse, toplam veri büyüklüğü

$$\mathrm{D} = \sum_{\mathcal{K}} \mathrm{D}_k \tag{23}$$

ile gösterilir.

Federe öğrenmede, eşitlik (24)'te, katılımcı kendi yerel verisiyle, sunucudan gelen modeli güncelleştirir. Katılımcı k, en uygun yerel model parametresinin değerini "$\boldsymbol{\theta}_k$" bulabilmek için $\mathcal{F}_k(\boldsymbol{\theta})$ kayıp fonksiyonunu aşağıdaki eşitlikteki gibi minimize eder [27].

$$\boldsymbol{\theta}_k = \arg\min_{\boldsymbol{\theta}} \mathcal{F}_k(\boldsymbol{\theta}) \tag{24}$$

Katılımcılar, $\boldsymbol{\theta}_k$ model parametrelerini sunucuyla paylaşır. Aşağıdaki eşitlikte gösterildiği üzere, model parametreleri birleştirilir. Toplama işlemi yapılırken, her bir yerel katılımcının minimize edilmiş model parametresi ve veri büyüklüğü çarpılarak toplanır. Sonuç toplam veri büyüklüğüne bölünür. Sonuç olarak sunucu, model parametrelerini, model parametrelerinin ağırlıklı ortalamasını alarak global bir model oluşturmak üzere birleştirir [27].

$$\boldsymbol{\theta} = \frac{\sum_{k \in K} \mathrm{D}_k \boldsymbol{\theta}_k}{D} \tag{25}$$

Katılımcılar, yukarıdaki eşitlikte (24) verilen, yerel model parametrelerini minimize etmek için gradyan inişi kullanırlar. η öğrenme katsayısı, t yineleme sayısı olmak üzere, $t = \{1, \dots, T\}$,T toplam yineleme sayısıdır. $\boldsymbol{\theta}_k^{(t)}$, t. yinelemedeki model parametresi olup, aşağıda verilen eşitlik (26)'teki fonksiyonunun k. katılımcının $\mathcal{D}_k$ ve $\theta_k^{(t)}$ değerlerine bağlı gradyanını temsil eder [27].

$$g\left(\theta_k^{(t)}\right) = \nabla \mathcal{F}_k(\theta_k^{(t)}) \tag{26}$$

Ortak bir modelin oluşturulması için sunucu ve katılımcılar model parametrelerini T yineleme sayısı kadar birbirlerine gönderim sağlayarak, gradyan inişi ile en iyi model elde etmeye çalışılır. Model yakınsayana kadar aşağıdaki eşitlikler (27) ve (28) uygulanmaya devam eder [27]:

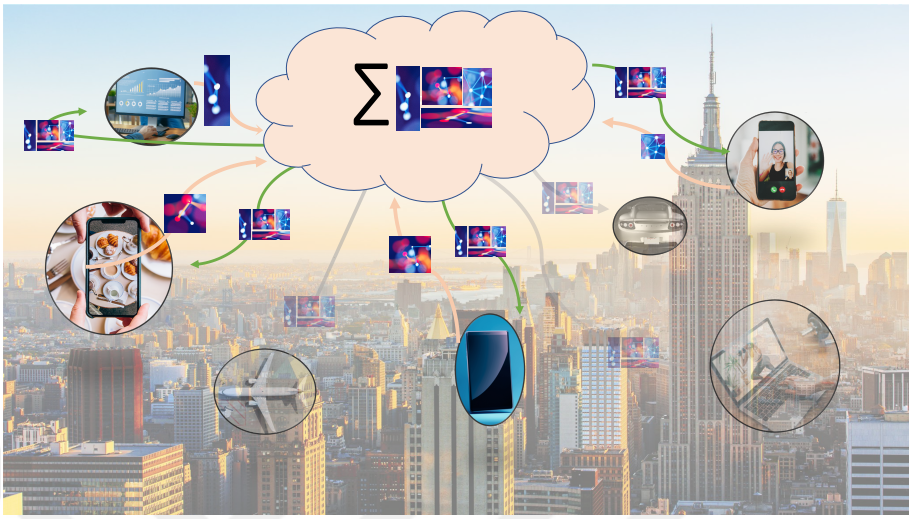
Katılımcılarda, parametre güncellemesi eşitlik (27)'deki gibi yapılmaktadır:

$$\boldsymbol{\theta}_k^{(t+1)} = \boldsymbol{\theta}_k^{(t)} - \eta g(\boldsymbol{\theta}_k^{(t)}) \tag{27}$$

Sunucularda , parametre güncellemesi eşitlik (28)'deki gibi yapılmaktadır:

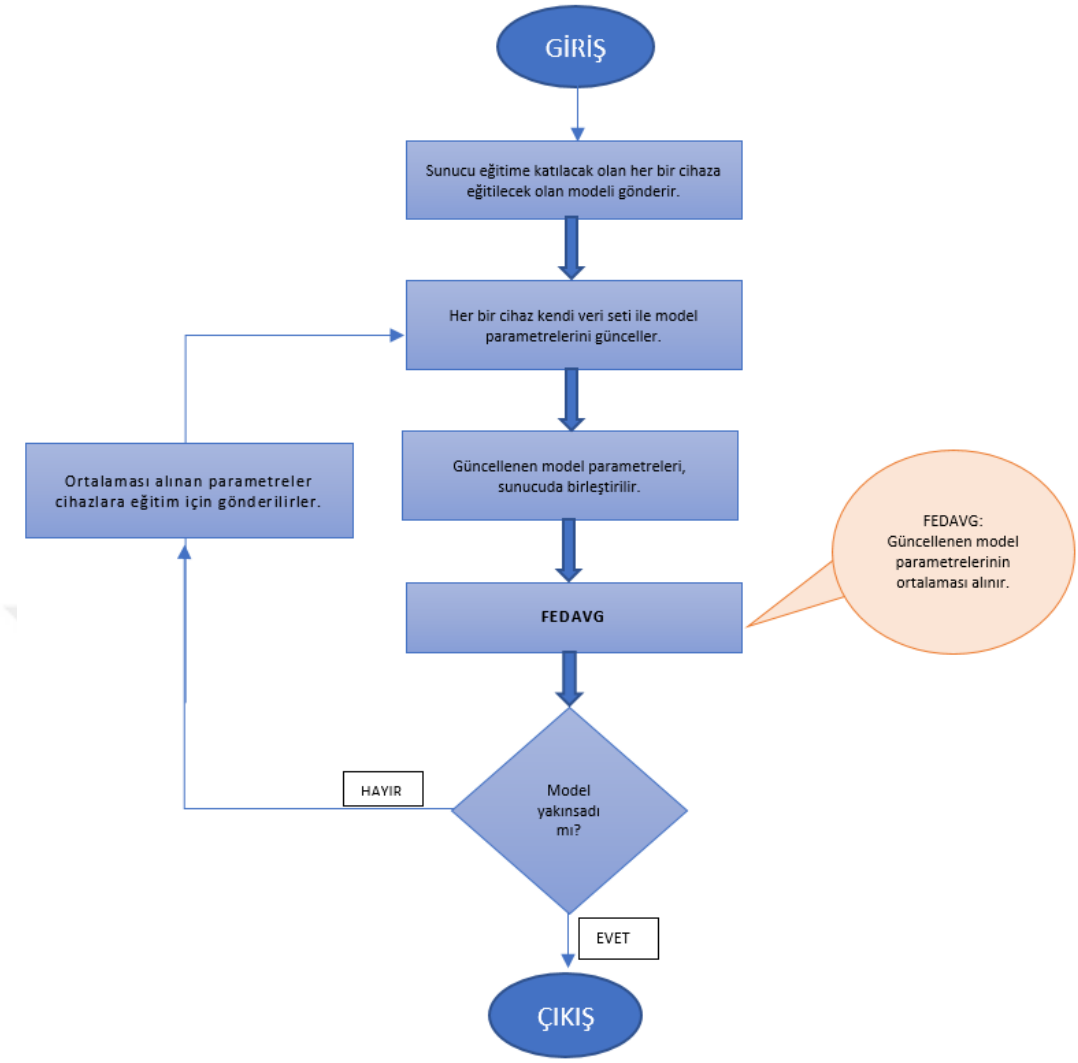
$$\boldsymbol{\theta}^{(t+1)} = \frac{\sum_{k \in K} D_k \, \boldsymbol{\theta}_k^{(t+1)}}{D} \tag{28}$$

Şekil 2.18'den görüldüğü üzere, federe öğrenmenin yapısını içerir. Katılımcı ve sunucudaan oluşan sistemde, sunuca birleştirilme işi gerçekleşir. Sunucu ve katılımcı arasında da sürekli bir iletişim mevcuttur.



Şekil 2.18. Federe öğrenme yapısı

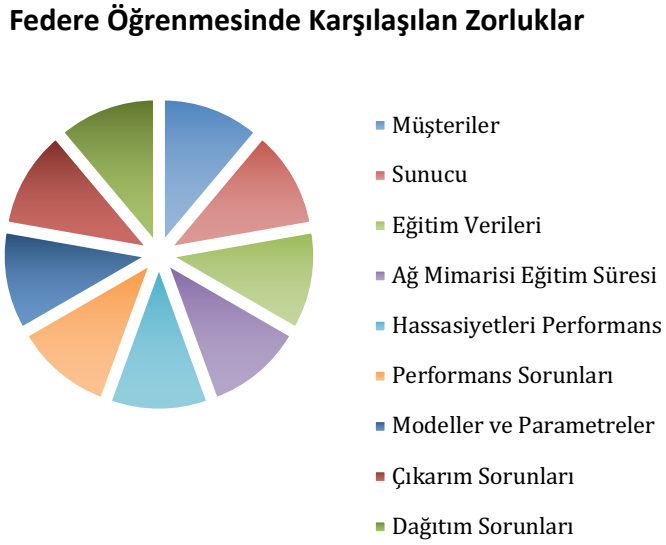
Şekil 2.19 federe öğrenmenin nasıl gerçekleştiği FEDAVG metoduyla akış diyagramı şeklinde gösterilmiştir. FEDAVG metodu bölüm 2.3.3.2'de incelenmiştir.



Şekil 2.19. Federe öğrenme akış şeması

2.3.2. Federe Öğrenmedeki Zorluklar

Federe öğrenmesinin yaygınlaşmasıyla birlikte ortaya çıkan zorluklar araştırılmış ve araştırılmaya devam etmektedir. Şekil 2.20 federe öğrenmede karşılaşılan tipik zorlukların hangi neden kaynaklı olduğu, eşit dağılım gösterdiği kabul edilerek göstermektedir [3][4]:



Şekil 2.20. Federe Öğrenmede Karşılaşılan Zorlukların Kaynakları

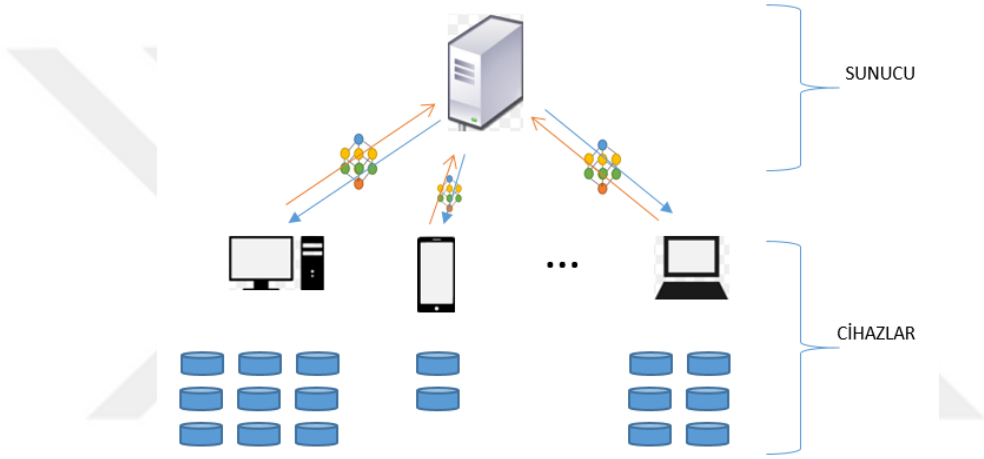
Bu zorluklardan olan veri düzensizliği, iletişimdeki maliyet, sistemsel heterojenlik, istatistiksel heterojenlik, gizlilik ilerleyen bölümlerde incelenmiştir:

2.3.2.1. Veri Düzensizliği

Cihazlar verinin eşit yayılmaması, federe öğrenmede, günümüz teknolojisindeki önemli zorluklardan biridir. Her cihazın farklı büyüklükte ve sayıda verisi olması çok olasıdır. Gerçek hayatta veri eşit dağılmamıştır. Her cihazın kapasitesi ve sahip olduğu veri sayısı farklıdır. Bu da her cihazın öğrenmesini farklı etkileyebilir. Federe öğrenmede veri düzensizliğinin olduğu bir sistemde, bazı cihazların aşırı uyum sağlamasına ya da tam öğrenememesine neden olabilir. Bu dengeleme sorununa sebep olabilir.

Veri düzensizliği iletişim yüküne de sebep olabilir. Veri boyutu büyük olan cihazlarda, iletişim kanalları, büyük model parametrelerini iletmek durumunda kalabilir. İletişim kanalları yetersiz olması durumunda ise büyük veriye sahip cihaz için öğrenme sorunlarına sebep olabilir. Büyük boyutlu veriye sahip olan cihazın bağlantısının kopması gibi durumlar oluşabilir. Bu durumda, katılımcı sayısının azalmasından ötürü, veri miktarı az olan cihaz için de öğrenme sorunlarına sebep olur.

Şekil 2.21'de her cihaz eşit miktarda veri içermemektedir. 2023 yılında kullanılan bir telefonun içindeki veri, aynı yıl üretilen bir bilgisayarın verisinin yarısına sahip olabilir. Örneğin, ESK (Elektronik Sağlık Kaydı) araştırmacıların araştırma yaparak tıbbi anlamda geri dönüşler verdikleri hastaların elektronik ortamda bilgilerinin depolandığı bir sistemdir. Her hastaneden yeteri kadar kayıt alınamayacak olabilir. Köylerden gelen veri yoğunluğuyla büyük şehirdeki veri yoğunluğunun aynı olmamasıyla beraber tek bir hastaneden ESK kaydının da alınması araştırmacılar için de yetersiz bir giriş kaynağı olabilir. Federe öğrenme bu gibi durumlarda yetersiz kalıp sonuç vermesi zor olabilir [29].



[6]

Şekil 2.21. Federe Öğrenmede Verideki Düzensizlik

2.3.2.2. İletişimdeki Maliyet

Federe öğrenmede veri cihazlarda kaldığı için, cihazların kendi verilerini kullanarak eğittiği model sunucuyla paylaşılır. Her yinelemede, model değişiminin olması ve yinelemenin model yakınsayana kadar devam etmesi, iletişim maliyetini arttırmaktadır. Modellerin daha verimli olarak güncellenmesi iletişimdeki maliyeti düşürmeyi sağlayabilir. Geleneksel makine öğrenmesinde verileri tek bir merkeze toplamanın iletişim maliyeti çok fazla iken, federe öğrenmede çok fazla cihazla sürekli olarak iletişim kurma maliyeti, sunucuda toplanma sırasında birleştirilme maliyeti, bağlantı kopma durumunda, bağlantısı kopan cihaz için tekrardan öğrenime sokma maliyeti mevcuttur. Örneğin, federe öğrenmede, cihaz sayısının fazla olması sistemin iyi eğitilmesine büyük katkı sağlarken, cihaz sayısının artışıyla, iletişim yoğunluğunun artışına sebep olur. Bu

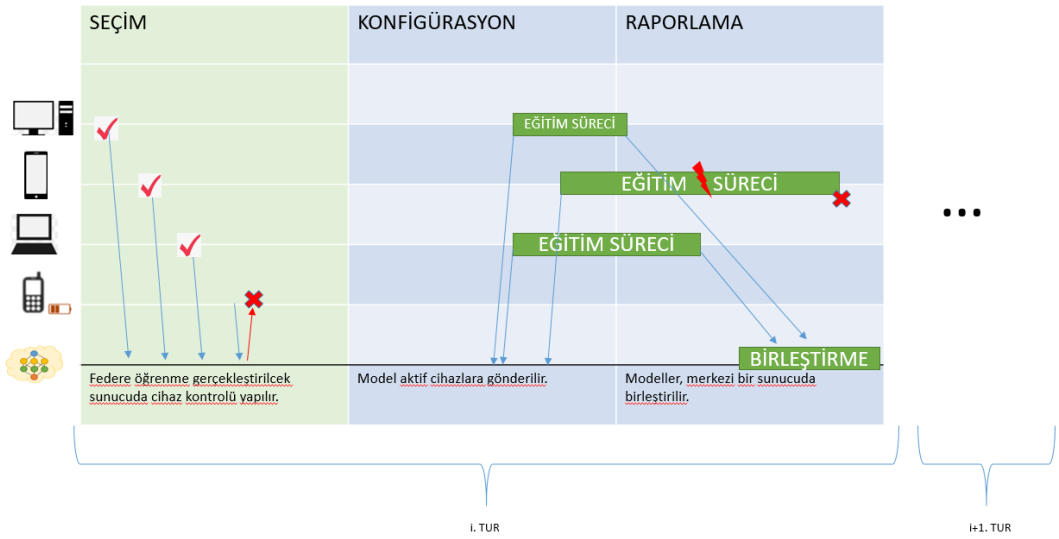
durum da maliyetteki artışa sebep olabilir [5]. Birçok katılımcının katıldığı bir sistemde katılımcıların sürekli olarak iletişim kurarak model parametrelerini güncellemesinin sıkıntı olmasıyla birlikte çoklu iletişim turu büyük bir bant genişliği gereksinimi oluşturmaktadır bu da iletişimin sıkıntıya düşmesine sebep olabilir [29].

2.3.2.3. Sistem Heterojenlik

FÖ yönteminde, bağlanılan cihazların özellikleri uyum ve tutarlılık açısından önem taşımaktadır. Her cihazın ağ bağlantısı donanım, yazılım, veri kapasitesi farklı olabilir. Bununla birlikte FÖ yönteminde öğrenmeye katılan cihaz sayısı arttıkça sistemin öğrenmesi de hızlanır. Fakat cihaz sayısının artışı sistem heterojenliğinin artışına sebep olabilir. Cihazların bağlanmama sorunları, bir turda katılıp diğer turda katılamama sorunları gibi iletişim problemleri çekilebilir. Bu durum da sistemin doğruluk yüzdesini düşürebilir.

Şekil 2.22'de, federe öğrenmedeki sistem heterojenliğinin nasıl oluştuğuyla ilgilidir. Cihazlar teknolojik, donanım, yazılım, bağlantı gibi birçok konuda farklılık gösterebilir. Bu farklılık federe öğrenme sürecinde bağlantı kopmalarına sebep olmaktadır. Görselde görüldüğü üzere, telefon yetersiz bataryadan kaynaklı olarak eğitimini tamamlayamamıştır. Federe öğrenme, sistem heterojenliği konusunda sistemin sürekli değişken olmasına uyum sağlayabilmelidir. Telefonun öğrenmesinin kesilmesi, diğer cihazların doğruluk oranlarını azaltmayacak şekilde tolere edebilmeyi öğrenmelidir.

Federe öğrenmede her bir turdaki katılan cihaz sayısının sürekli değiştiği gibi her bir cihazın hesaplama ve enerji kullanımı farklı olmaktadır. Örneğin tıp alanında kullanılan cihazlarda enerji kısıtlamaları yaygındır. Bununla birlikte daha basit hesaplamalar gerektirir. Federe öğrenmenin de bu alanda gelişmesi, her alanın ihtiyaçlarına göre hesaplama yeteneğine sahip olması gerektiği öngörülmektedir [29].

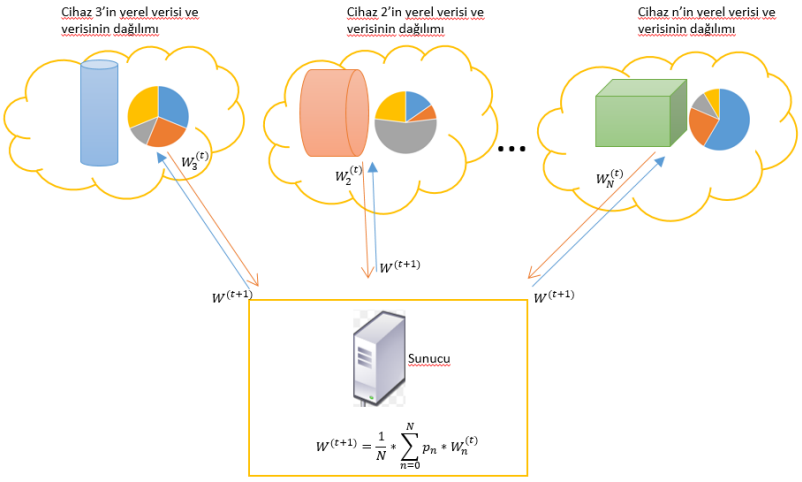


Şekil 2.22. Federe öğrenmede sistemlerdeki düzensizlik

2.3.2.4. İstatistiksel Heterojenlik

İstatistiksel heterojenlik, bir iletişim ağı üzerinde yer alan katılımcıların veri iletimini eşit dağılmayan bir şekilde gerçekleştirmesi durumunu ifade eder. [3]. FÖ sisteminde, merkezi sunucu, bütün cihazlardan gelen modelleri toplar. Merkezi sunucu içinde birleştirip var olan global modeli günceller. Cihazlardaki öğrenmeyi sağlamak için modeli tekrardan dağıtır. Bu süreçte cihazlardan modeli toplama işlemi yapılırken ortaya zorluklar çıkabilir. Bunun temelinde ise cihazların veriyi işleme süresinin ve veri boyutunun aynı olmaması olabilir. Cihazların sistemsel olarak değişiklik göstermesi farklı bir zorluk iken, istatistiksel heterojenlikteki zorluk cihazlardan model toplanırken ve dağıtılırken her cihaza eşit zamanın harcanamayacak olmasıdır. Bu sebeple veri boyutu fazla olan cihazı tur içindeki bekleme süresi diğer cihazlara göre fazla olabilir. Bu durumda modelin birleşmesi gecikebilir. Sistemde gecikmeler olup karmaşıklık artabilir [4] [5] [7].

Şekil 2.23'te her cihaz için farklı veri dağılımı ve farklı verilerin olduğu görülmektedir. Bu durumda modelin sunucudan katılımcıya, katılımcıdan sunucuya model iletimi sırasında federe öğrenmede istatistiksel düzensizlik meydana gelebilir.



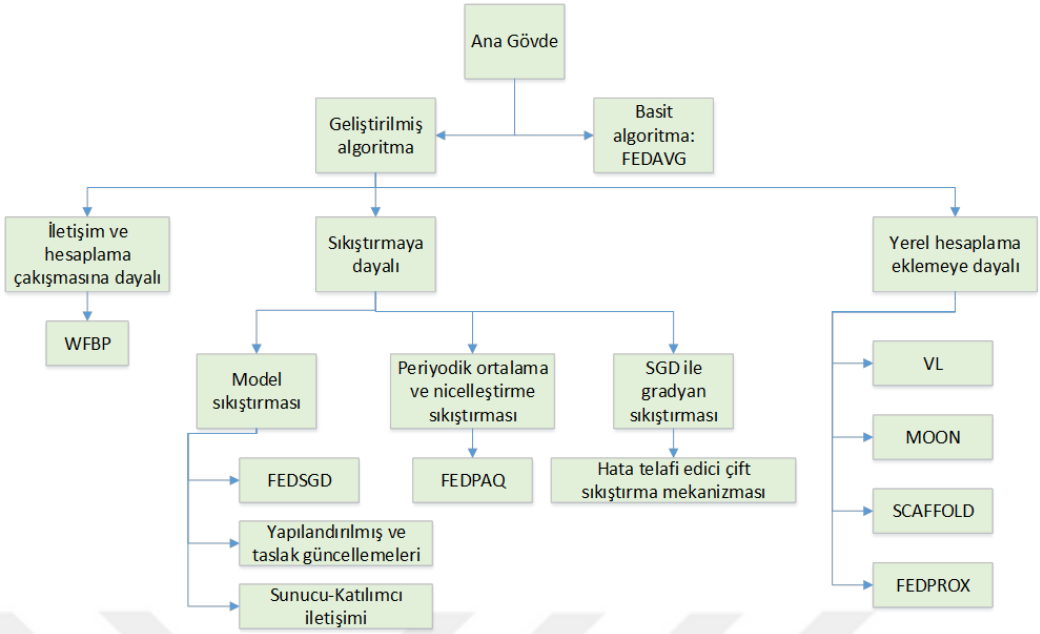
Şekil 2.23. Federe öğrenmede istatistiksel düzensizlik

2.3.2.5. Gizlilik

FÖ yönteminin hedefi kullanıcı gizliliğini sağlamaktır. Bunun için, sistemde veri alışverişi yerine model alışverişi sağlanmaktadır. Federe öğrenme, merkezi makine öğrenimindeki gizlilik endişesini azaltma hedefi ile çalışmalarını sürmesine rağmen, katılımcıların model parametrelerinin sızma riski vardır. Model güncellemelerinin iletimi sırasında model güncellemeleri ele geçirilebilir. Modelin yanıltılması için model değiştirilebilir. Federe öğrenmede, gizliliğin sağlanması için çalışmalar sürdürülmektedir. Sistemde gizliliğin verimi değiştirmek için yapılan çalışmaların bir maliyetinin olur. Sistemin ihtiyaç durumuna göre değerlendirme yapılıp, en uygun yolun bulunması yönünde aksiyon alınır.

2.3.3. Federe Öğrenme Algoritmaları

Federe öğrenmede yapılan araştırmaların artışı, federe öğrenmedeki geliştirilebilir özelliklerin gündeme gelmesini sağlar. Federe öğrenmede, 2.3.2 bölümünde bahsedilen zorluklara çözüm bulabilmek için federe öğrenmeyi optimize eden algoritmalar üzerinde çalışmalar yapılmaktadır. Çalışmalar, yeni birleştirme yöntemleriyle, zorlukları, verimli bir şekilde çözmeyi planlamaktadır. Literatürde en yaygın olarak adı geçen algoritmaların bulunduğu tipik bir sınıflandırma Şekil 2.24'te gösterilmiştir [28]:



Şekil 2.24. Federe öğrenmeyi optimize eden algoritmalar

Şekil 2.24'te de yer alan FEDAVGM, FEDAVG ve FEDSGD yöntemleri aşağıda anlatılmıştır.

2.3.3.1. FEDSGD

Bölüm 2.3.3.2'de bahsedilen FEDAVG algoritması, FEDSGD algoritmasını kapsamaktadır. Şekil 2.24'te yer alan "Federe öğrenmeyi optimize eden algoritmalar" sınıflandırmasında görüldüğü üzere,FEDSGD geliştirilmiş algoritmalardan, model sıkıştırmasına dayanan yöntemlerde incelenmektedir. Model sıkıştırması, modeli boyutunu azaltmak için kullanılır. Amacı hafıza ve depolama verimi sağlamak, daha hızlı bir sistem elde etmek ve dağıtımın kolaylaştırmak olarak açıklanabilir.

FEDSGD algoritması, Rothchild ve ekip arkadaşları tarafından önerilmiştir. Sunucu ve istemcilerin birlikte çalışmasıyla, öğrenimi FEDAVG'den daha hızlı gerçekleştiren bir yöntem olarak sunmuşlardır. [43]. Merkezi olmayan bir sistem olan FEDSGD yönteminde, her katılımcının ayrı verisi vardır. Yerel eğitim sırasında, katılımcının kendi verisi için, tek bir yerel tur içerir. Yerel tur sırasında, kendi verilerinden tek bir stokastik gradyan inişi ile hesaplamasını yapmaktadır [42]. Katılımcının, kendi veri sayısı, paket boyutu sayısına eşittir. Her katılımcı, yerel veri seti üzerinde gradyanı hesapladıktan sonra bu gradyanı merkezi sunucuya iletir. Merkezi, sunucu mevcut olan modeli

katılımcılardan gelen model ile güncelleyerek tekrardan katılımcılara dağıtır. Bu yineleme öğrenme işlemi tamamlanana kadar gerçekleşir.

FEDAVG yönteminde, enerji kısıtlamaları, düşük bant genişliği, iletişim trafiği öğrenmeyi kötü etkiler. FEDSGD yönteminde, sıkıştırılmış gradyanlar kullanıldığı için veri trafiği azalırken, iletişim verimliliği artmaktadır. Bu yöntemde FEDAVG yönteminin aksine model parametreleri tam gönderilmez. Model boyutunu azaltması veri iletişimindeki yükü hafifletir. Fakat, kayıp ve doğruluk değerlerinin yakınsamasından ödün verilmiş olur. Özellikle veri düzensizliğinde ve seyrek veri setlerinde doğruluk düşüşü gözlemlenebilir [43]. Seyrek veri seti, bir veri setindeki örneklerin çoğunlukla sıfır veya çok düşük değerlere sahip olduğu durumu ifade eder.

FEDSGD, öğrenmeye katılımı düşük olan bir müşterinin eğitime katılmasını, model güncellemelerini sıkıştırarak yakınsamasını sağlayabilir. Hesaplama açısından verimlidir fakat çok fazla yineleme gerektirir. Doğruluk oranının düşük ve kayıp oranının yüksek kalması bir sorun olarak düşünülebilir [28].

Aşağıda FEDSGD yönteminin matematiksel ifadesi yer almaktadır:

Her cihaz kendi verisini kullanarak, sunucudan gelen model ile, modelin eğitimini gerçekleştirir. Güncel model w_t olarak isimlendirilir. k'ıncı katılımcının, yerel veri kümesi ile eğittiği modelin, kayıp fonksiyonunun ($\mathcal{F}_k$), gradyanını temsil eden eşitlik (29)'da verilmiştir:

$$g_k = \nabla \mathcal{F}_k(w_t) \tag{29}$$

Yerel katılımcılar güncellediği gradyanı sunucuyla paylaşır. Gradyanların ağırlıklı ortalamasını alarak katılımcıların hepsinden elde edilen global gradyan eşitlik (30)'da verilmiştir:

$$\sum_{k \in K} \frac{\mathrm{n}_k}{\mathrm{n}} * \ g_k = \nabla f(w_t) \tag{30}$$

Sunucu, gradyanların ağırlıklı ortalamasını alarak katılımcıların hepsinden elde edilen global gradyan ile, aşağıda eşitlik (31)'de verildiği gibi model güncellemesini gerçekleştirir [30].

$$w_{t+1} = w_t - \eta \,.\nabla f(w_t) \tag{31}$$

2.3.3.2. FEDAVG (ORTALAMA FEDERE ÖĞRENME):

FEDAVG, Şekil 2.24'te görüldüğü üzere, ana gövdenin hemen altında basit algoritma olara geçer. Federe öğrenme için kullanılan bir optimizasyon algoritmasıdır. FEDAVG uygulaması temel ve uygulanması basit bir algoritma olmasıyla dolayısıyla, FÖ'de optimizasyon algoritması olarak tercih edilen bir algoritma denilebilir [33][4].

FÖ'de, dağıtılmış cihazlar, kendi yerel verileri ile model üzerinde güncelleme yapar ve ardından yapılan yerel eğitim sonuçlarını merkezi bir sunucuya iletir. Merkezi sunucu, toplanan yerel model güncellemelerini kullanarak birleştirilmiş global modeli oluşturur. Federe öğrenme geliştirilirken farklı yöntemler uygulanarak sistemin en verimli hali bulunmaya çalışılmaktadır. SGİ ise derin öğrenmede sık sık uygulanan ve doğruluğun yüksek olduğu optimizasyon yöntemidir [22][30]. FEDAVG (ortalama federe öğrenmesi) , eğitim işlemi sırasında yerel cihazlarda SGİ algoritmasını kullanarak federe öğrenmede yüksek doğruluk oranlı sunar [31][32].

FEDAVG yöntemindeki temel, bütün yerel katılımcılarda SGİ algoritması uygulaması ile kayıp fonksiyonunu minimum değere getirmeyi amaçlar. Sunucuda yerel modelleri toplayarak birleştirir [32].

Algoritmalar sürekli olarak gelişmekte ve yenilenmektedir. Federe öğrenmede her seferinde model sunucudan indirilir, eğitim gerçekleştikten sonra da tekrardan sunucuya yükleme yapılarak öğrenme işlemi devam etmektedir. FEDAVG yöntemi bu yükleme ve indirme işleminde tam model parametrelerini iletmeye çalışmaktadır. Bu durum, iletişimde büyük bir trafiğe sebep olmaktadır. FEDAVG algoritması büyük miktarda veriler ile model eğitimi zorlaşır [43].

FEDSGD yönteminden farklı olarak sistem yerel cihazın birden çok SGİ yapmasına olanak verir ve yerel veriyi minik paketler halinde eğitim sağlayarak, modelin sunucudan birleştirilmesinden önce çoklu hesaplama yapılabilir.

Matematiksel olarak yazmak istenirse, her katılımcı için eşitlik (32) uygulanır. Her cihaz kendi verilerini kullanarak SGİ gerçekleştirir [30] [47].

$$w_{t+1}^{(k)} = w_t^{(k)} - \eta g_k \tag{32}$$

Eşitlik (33)'te görüldüğü üzere, merkezi sunucu, her cihazdan gelen modellerin ağırlıklı ortalamasını almaktadır [30].

$$w_{t+1} = \sum_{k \in K} \frac{n_k}{n} * \ w_{t+1}^{(k)} \tag{33}$$

FEDSGD yönteminden farklı olarak FEDAVG yönteminde katılımcılara daha fazla hesaplama eklenilebilir, yerel güncellemeler birden fazla sayıda tekrarlanabilir. Sunucuya en son güncellenen model gönderilir.

2.3.3.3. FEDAVGM (ORTALAMA FEDERE ÖĞRENME)

FEDAVGM (kontrollü iniş ile ortalama federe öğrenme) yönteminde literatürde FEDCM (Federated Learning with Client-level Momentum) ismi de kullanılmaktadır. FEDAVGM, yani ortalama federe öğrenmesi kontrollü iniş yöntemi, seyrek veri dağılımı olan sistemlerde verilerin daha hızlı ve istikrarlı eğitimi için kullanılmaktadır. Bunun dışında sınırlı etiket verisinin bulunması, salınımın önemli olduğu durumlarda salınımın azaltılması, yineleme sayısını azaltarak iletişim maliyetinin azaltılması FEDAVGM yönteminde avantaj sağlamaktadır [44][9].

FEDAVGM yönteminde, 2.3.3.2 bölümünde FEDAVG yönteminden, farklı olan kısım FEDAVG yöntemine momentum eklenmesi olarak tanımlanabilir. FEDAVGM

yönteminde SGİ üzerine momentum eklenmesi, eğitimi hızlandırır ve salınımı azaltır. Verideki heterojenlik arttıkça momentumun etkisiyle FEDAVGM yönteminin performansı sabit kalır [49]. Ayrıntılı olarak bölüm 2.2.1.8'de anlatılan momentum, gradyan inişi sırasında geçmiş hesaplamalardan yararlanarak, dikey gürültüyü yok edip yatayda ilerlemeyi arttırmayı hedefler.

$V_{t=0} = 0$ ve $0 < \beta < 1$ olmak üzere aşağıdaki eşitliklerde (35) ve (34) kontrollü iniş yer verilmiştir:

$$w_{t+1} = w_t - \alpha V_t \tag{34}$$

$$V_t = \beta V_{t-1} - (1-\beta)\frac{\partial L}{\partial w_t} \tag{35}$$

Aşağıda matematiksel olarak kontrollü inişin, genel FEDAVG yönteminde nasıl uygulandığının açıklaması yer almaktadır [44].

(n_k) k'ıncı katılımcılarda bulunan veri sayısı olmak üzere,

$$\mathrm{n} = \sum_{\mathrm{k}=1}^{\mathrm{K}} (\mathrm{n_k}) \tag{36}$$

Toplam veri sayısı yukarıdaki gibi ifade edilmiştir.

Ortalama federe öğrenmesinde, sunucuda, katılımcılardan gelen ağırlıklar aşağıdaki eşitlik (35) ile birleştirilmektedir:

$$\nabla w_{t+1} = \sum_{k \in K} \frac{n_k}{n} * \ w_{t+1}^{(k)} \tag{37}$$

Eşitlik (38)'de birikmiş gradyanlar β ile çarpılarak momentumun etkisine karar verilir. Daha sonra birikmiş (geçmiş) gradyanlar , ortalaması alınan gradyanlar ile toplanır:

$$V_{t+1} \leftarrow \beta V_t + \nabla w_{t+1} \tag{38}$$

Eşitlik (39)'da biriken gradyanlar, güncel iletişim turunun ağırlıklarının güncellemesi için kullanılır:

$$w_{t+1} \leftarrow w_t - \eta V_{t+1} \tag{39}$$

FEDAVG ile uygulanan yönteme eşitlik (38) ve (39) eklenmesi demek, her yinelemede modelin sunucunun eski gradyanların işleme sokulmasını ifade eder. Eski gradyanların işleme sokulması salınımları azaltarak öğrenmeyi hızlandırmasına katkıda bulunur.

3. DENEYSEL ÇALIŞMALAR

Bu bölümde, FÖ yönteminin, uygulaması için MATLAB ile deneysel çalışmalarda bulunulmuştur. Metodun uygulanması MNIST "Modified National Institute of Standards and Technology" veri seti ile gerçekleştirilmiştir. MNIST veri seti siyah beyaz görüntülerin bulunduğu, 0'dan 9'a kadar el yazısı ile yazılmış sayıları içerir.

Deneylerimiz; paket (batch) boyutu, cihaz sayısı, cihazın içindeki veri sayısı, epoch sayısı, cihazın içindeki verilerin düzensiz seçilmesi, gibi parametrelerin değiştirilmesiyle, FEDAVGM yönteminin öğrenme performansındaki değişimi gözlemleyip, sonuçlarının karşılaştırmasını içermektedir. Ayrıca FEDAVGM ve FEDAVG yöntemleri için de deneyler oluşturulup, yorumlarda bulunulmuştur.

Deneysel çalışmalar başlığımızın altında 2 adet alt başlığımız bulunmaktadır. Bunlardan "3.2 Deney Mimarisi" isimli başlığımız deneyimizde kullandığımızın kodun mimari yapısının içeriğine değinirken "3.1. Çalışma Sonuçları" isimli başlığımız altında yaptığımız çalışmaların sonuçlarını paylaşıyor olacağız.

3.1. Çalışma Sonuçları

Aşağıda görülen çalışmalarla ilgili olarak her çalışma kapsamında 4 adet çıktı grafiği alınmıştır. 4 adet çıktıdan 1 tanesi, sunucunun test verisiyle oluşturduğu kayıp-yineleme sayısı grafiğidir. 4 adet çıktıdan 1 tanesi, cihazların kendi içinde, kendi test verisiyle oluşturduğu kayıp-yineleme sayısı grafiğidir. Her cihaz için ayrı kayıp grafiği alınmıştır. 4 adet çıktıdan 1 tanesi, cihazların kendi içinde, kendi test verisiyle oluşturduğu doğruluk-yineleme sayısı grafiğidir. Her cihaz için ayrı doğruluk grafiği alınmıştır. 4 adet çıktıdan 1 tanesi, sunucunun test verisiyle oluşturduğu doğruluk-yineleme sayısı grafiğidir.

FEDAVGM metodunun uygulamasında, parametre değişikliğiyle FEDAVGM metodunun davranışları incelenmiş ve heterojen durumlar yaratılarak sistemin ve cihazların performansı karşılaştırılmıştır. Gözlemi yapılmak istenilen parametrelerin dışındaki parametreler sabit tutularak, karşılaştırılmak istenilen durumlar, düzgün bir şekilde ele alınmıştır. Paket boyutu etkisi, epoch değişimi etkisi, cihaz sayısı etkisi, farklı

heterojen dağılımların etkisi, FEDAVGM/FEDAVG yöntemlerinin etkisi farklı başlıklarda ele alınmıştır.

3.1.1. Paket Boyutu Etkisi

Bu bölümdeki çalışmalarda 3 müşterinin aktif olduğu 7 müşterinin pasif olduğu deneyde, federe öğrenme sistemi uygulanarak, 3 adet müşterinin, paket boyunun etkisi incelenmiştir. Paket boyutu 2.2.1.5. bölümünün altında bahsedilen Paket Boyutu isimli alt başlıkta anlatıldığı üzere bir seferde işlenecek olan örnek sayısıdır.

Katılımcıların ayrı ayrı kayıp ve doğruluk grafiği, sistemin genel kayıp ve doğruluk grafiği, görselleriyle birlikte açıklamalarda bulunulmuştur. Yorumlar bu bölümün altında deneyler bittikten sonra yapılmıştır.

Çizelge 3.1 numaralı tabloda aşağıda paket boyutuna göre koşum sonrası alınan kayıp/doğruluk grafiklerinin numerik değerleri verilmiştir:

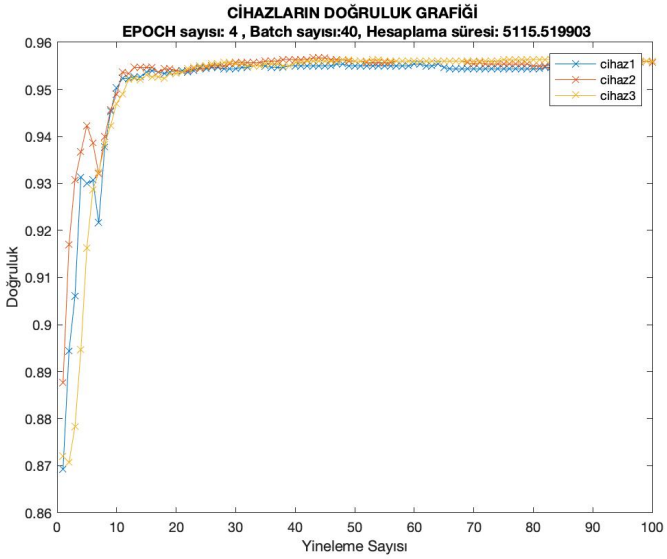
Momentum: M
Öğrenme Katsayısı: ÖK

Paket Boyutu	**Kayıp Değeri**	**Doğruluk Değeri**	**Hesaplama Süresi**	**Her Bir Cihazdaki Veri Sayısı**	**Epoch Sayısı**	**Cihaz Sayısı**	**M/ÖK**
4	%19,77	%96,97	5416	2000	4	3	M
40	%22,92	%96,23	5115	2000	4	3	M
100	%20,43	%95,87	5504	2000	4	3	M
1000	%33,49	%89,97	5828	2000	4	3	M
1000	%25,66	%92,40	11658	4000	4	3	M

Çizelge 3.1. Paket boyutu çalışmalarına yönelik kayıp/doğruluk değerleri

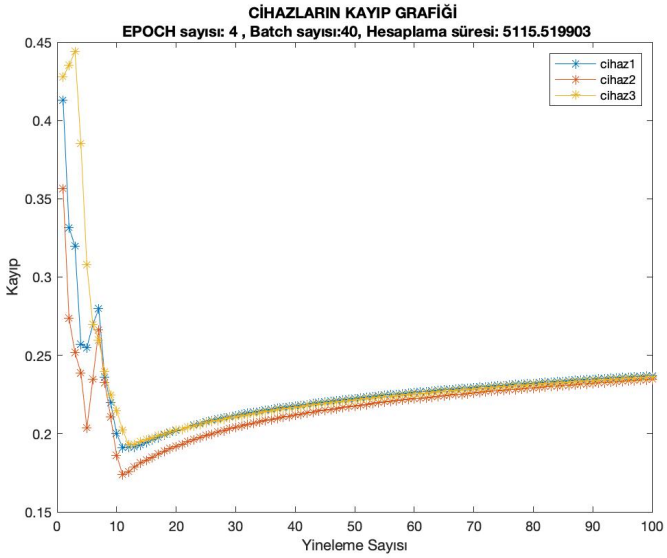
3.1.1.1. Paket Boyutu=40

Aşağıdaki grafik için paket boyutu 40, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri mevcutken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



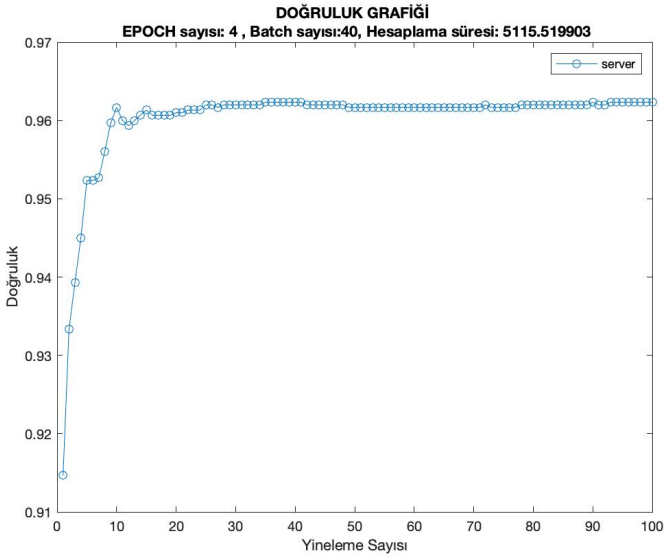
Şekil 3.1. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 40, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



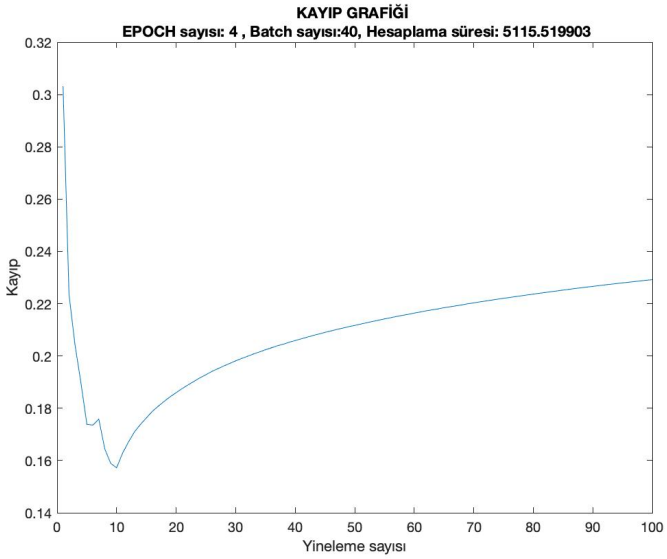
Şekil 3.2. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 40, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.3. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

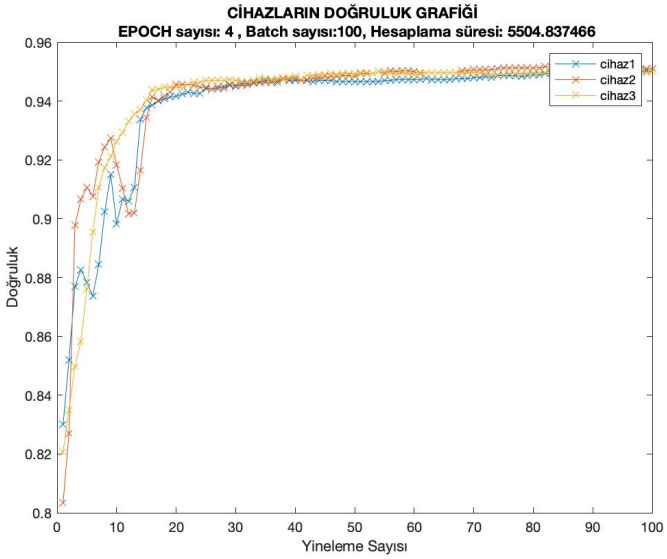
Aşağıdaki grafik için paket boyutu 40, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.4. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

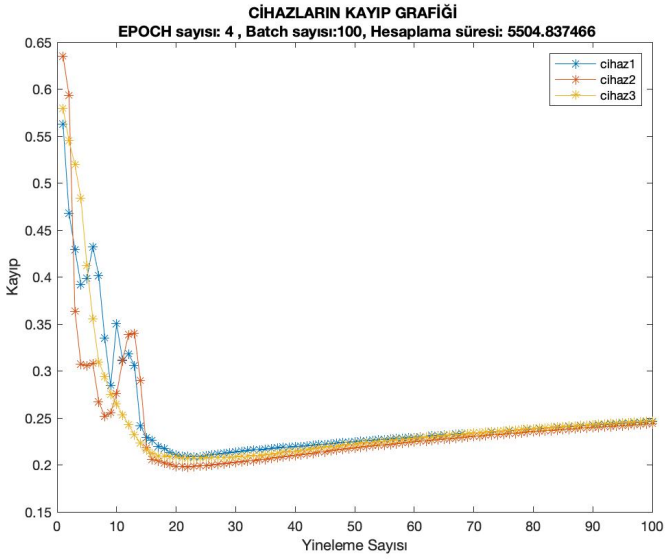
3.1.1.2. Paket Boyutu=100

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



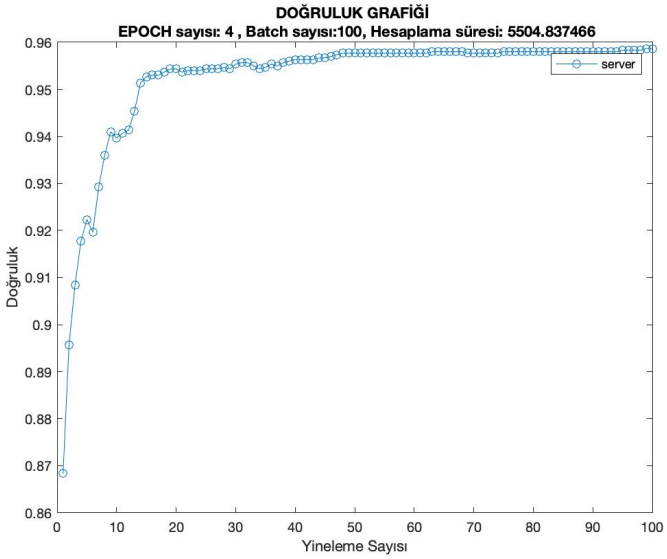
Şekil 3.5. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



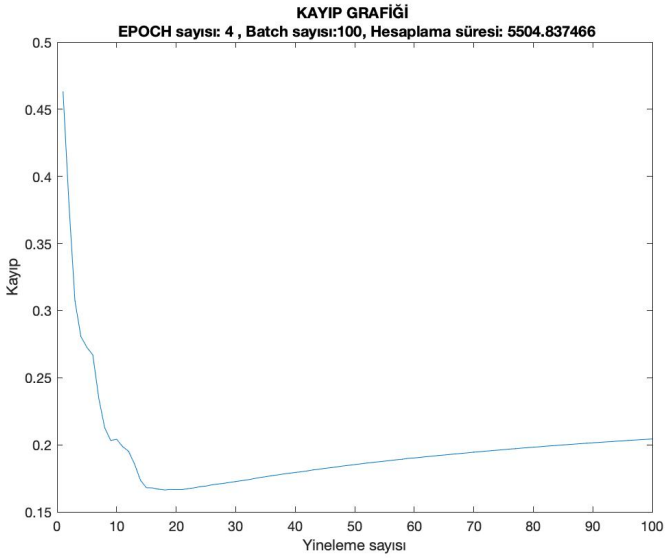
Şekil 3.6. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.7. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

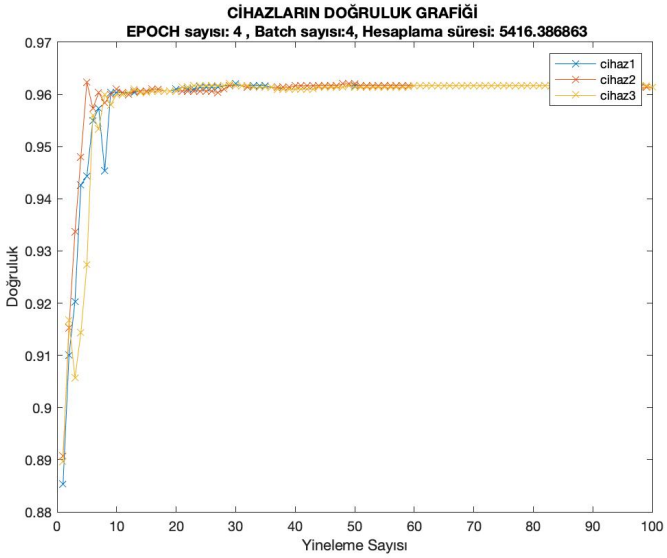
Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.8. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

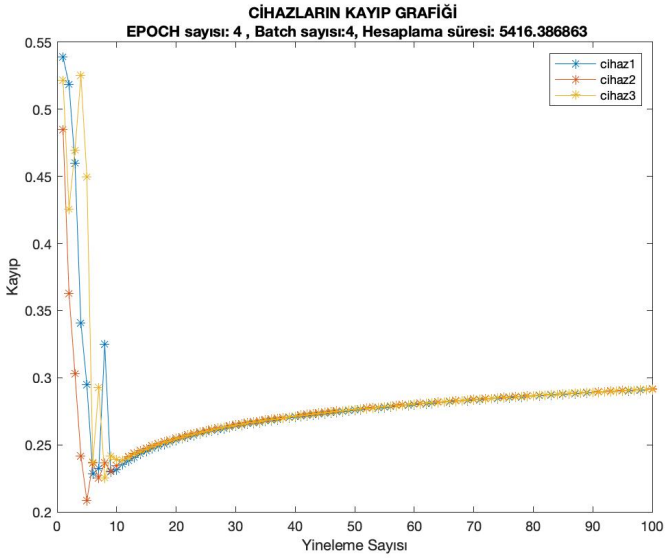
3.1.1.3. Paket Boyutu=4

Aşağıdaki grafik için paket boyutu 4, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



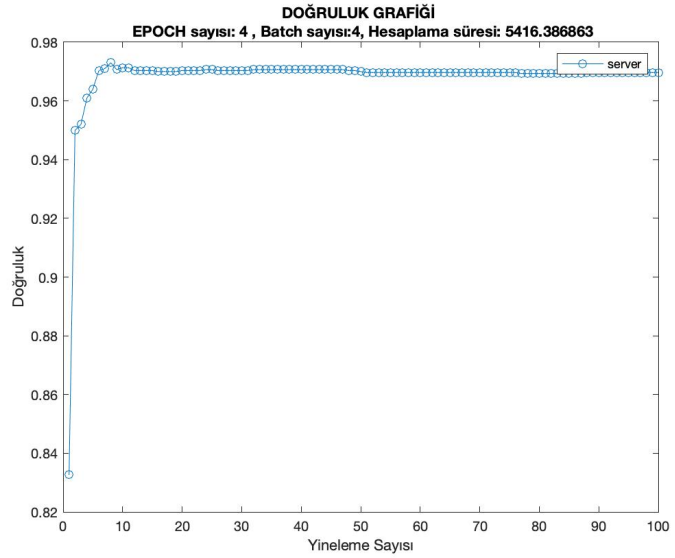
Şekil 3.9. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 4, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



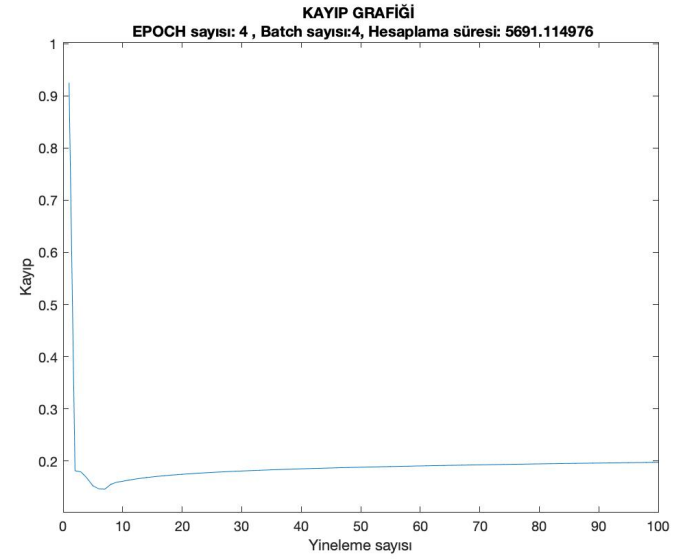
Şekil 3.10. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 4, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.11. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

Aşağıdaki grafik için paket boyutu 4, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.

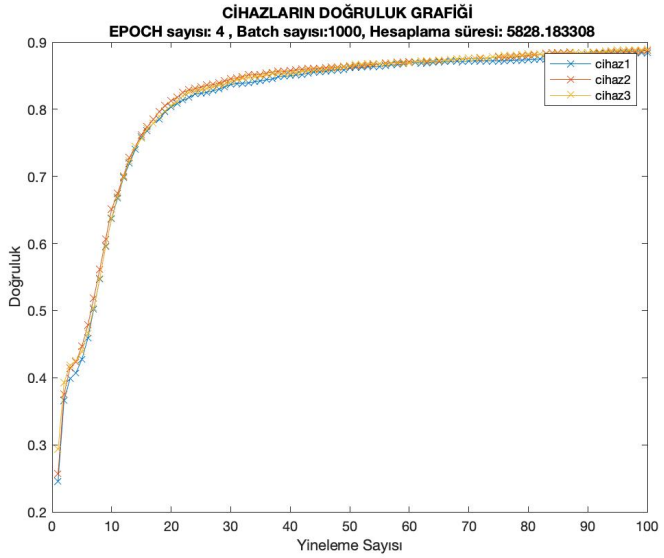


Şekil 3.12. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

3.1.1.4. Paket Boyutu=1000

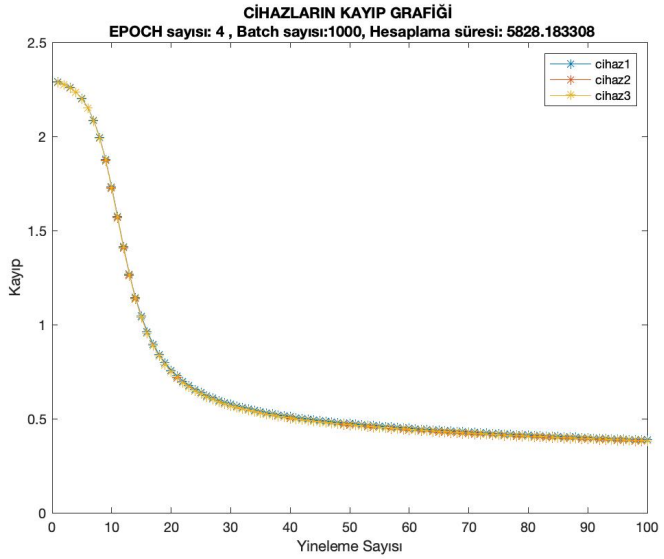
-Her Cihaz İçin Veri Boyutu 2000 :

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



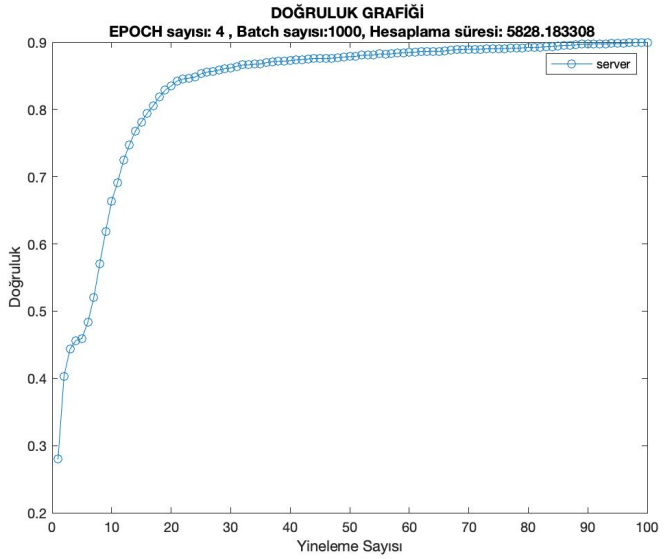
Şekil 3.13. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



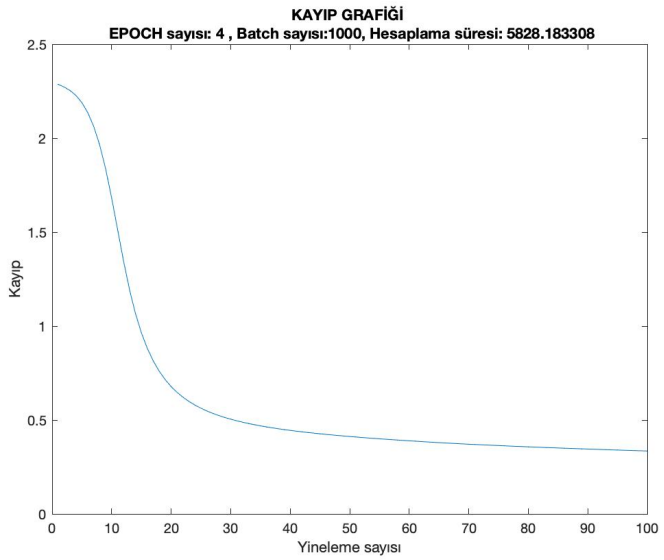
Şekil 3.14. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.15. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

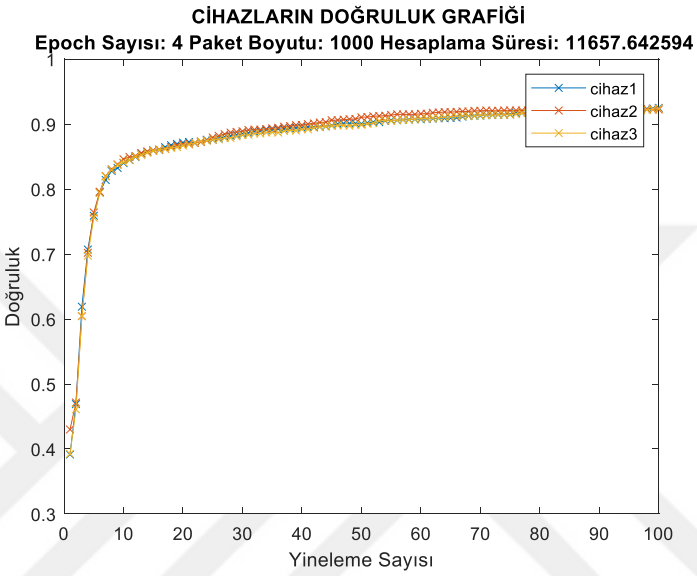
Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.16. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

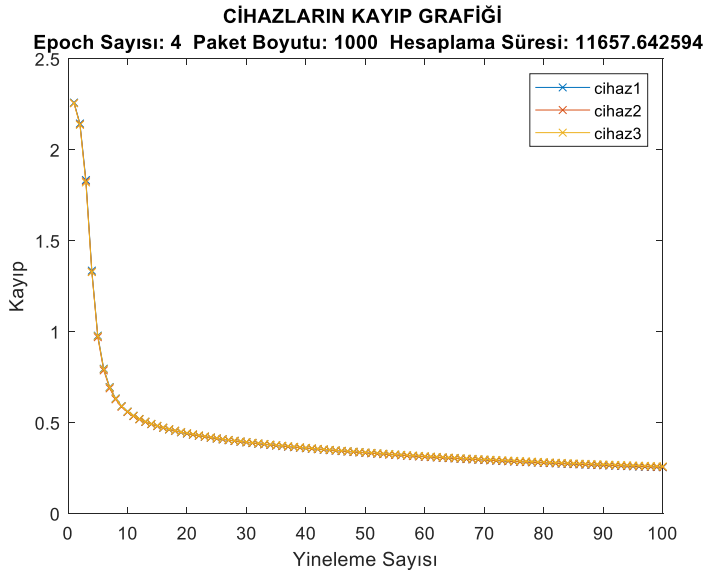
-Her Cihaz İçin Veri Boyutu 4000 :

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



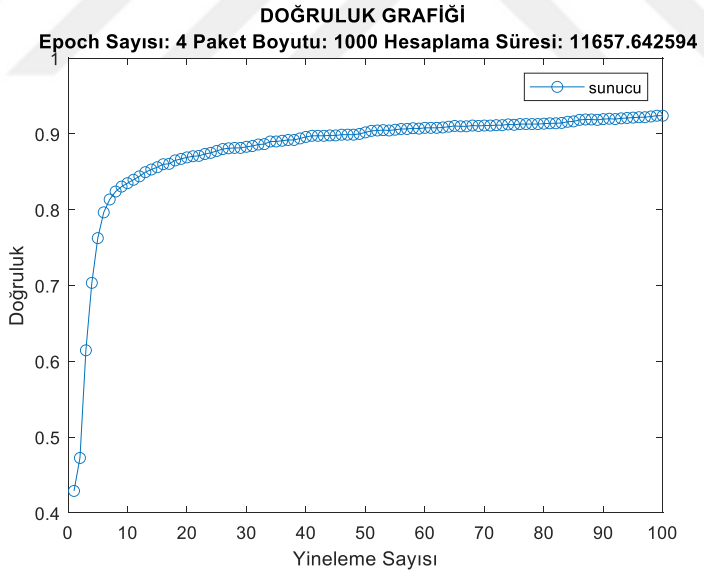
Şekil 3.17. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



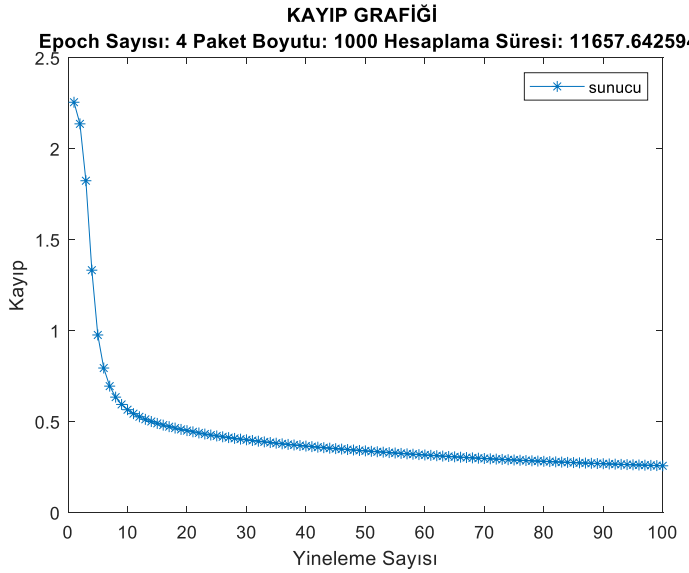
Şekil 3.18. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.19. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.20. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

Paket boyutunun büyük miktarda değişmesinin sonuçları bölüm 3.1.1.4. ve 3.1.1.3'te incelenebilir. Bölüm 3.1.1.3. yapılan deneyin paket boyutunu bölüm 3.1.1.4.'tekine göre küçük seçildiği için her işlemde ortaya çıkan maliyet fonksiyonu farklı olması muhtemeldir. Çünkü paket boyutunun azalmasıyla tek seferde işleme alınacak olan verinin azalmasına sebep olur. Tüm veriyi elden geçirmek için gereken yineleme sayısı artar. Her yineleme farklı maliyet fonksiyonuna sebep olabilir. Bu sebeple bölüm 3.1.1.3'te, Şekil 3.9. ve 3.10 grafiklerinde cihazlardaki kayıp ve doğruluk grafikleri gösterilmiştir. Değerler yakınsayana kadar düşük paket boyutu sebebiyle daha çok gürültü gözlemlenir. Bölüm 3.1.1.4'teki görsellerde yumuşak geçişli kayıp ve doğruluk grafikleri gözlemlenir. Paket boyutu 4 seçildiğinde, gürültü en fazla olup, sırasıyla 40,100,1000 değerlerinde giderek azalmıştır.

Bu bölümde yapılan çalışmalarda, paket boyutu arttıkça eğitim süresinin de genelde arttığı gözlemlenmektedir. Paket boyutunun artmasıyla tek seferde işleme alınacak olan verinin artmasına sebep olup, bu durum da öğrenme hızını azaltabilir. Paket boyutunun büyük olması, daha düşük çözünürlüklü bir öğrenmeye sebep olmaktadır. Paket boyutu 1000 olarak seçildiğinde, sistemin 100. yinelemede yakınsamaya yakın olduğu fakat tam yakınsamadığı gözlemlenebilir. Bu durumda da öğrenmenin geciktiği yorumu yapılabilir. Öğrenme süresinin gecikmesinin aksine hesaplama süresi de diğer paket boyutlarına göre daha düşüktür. İstenilen duruma bağlı olarak hızlı bir öğrenme sistemi için paket boyutunun büyütülmesi önerilebilir.

Bir önceki paragrafta paket boyutu büyük seçildiği zaman, hesap süresinin azalmasına ve öğrenme çözünürlüğünün gecikmesine sebep olduğu yorumu yer almaktadır. Aynı paket boyutuyla (1000) her bir cihazdaki işlenilen veri 2 katına çıkarılarak, 4000 olarak belirlenmiştir. Bu durum, öğrenmeyi, veri sayısı 2000 seçildiği duruma göre arttırmıştır. Fakat hesaplama süresi de 2 katına çıkmıştır. Bu durumun sebebi olarak işlenilmesi gereken verinin çok artması olarak gösterilebilir. Şekil 3.19'da veri boyutunun arttırılmasıyla, sistem yakınsamaya daha yakın olduğu görülmektedir.

Bütün parametreler sabitken ve paket boyutu azaltıldığında, modelin, cihazlarda aşırı öğrenmeye (overfitting) doğru gittiğini görükmektedir. (Şekil 3.10.) Paket boyutu küçüldükçe sistemin öğrenmesi yönünde daha pozitif bir durum sağlanırken, epoch sayısını sabit tuttuğumuz için bu durum aşırı öğrenme durumuna sebep olmuştur.

3.1.2. Epoch Etkisi

Bu bölümdeki çalışmalarda 3 müşterinin aktif olduğu 7 müşterinin pasif olduğu deneyde federe öğrenme gerçekleştirilmiştir. 3 adet müşterinin, federe öğrenme sisteminde, epoch sayısının etkisi incelenmiştir.

Katılımcıların ayrı ayrı kayıp ve doğruluk grafiği, sistemin genel kayıp ve doğruluk grafiği, görselleriyle birlikte açıklamalarda bulunulmuştur. Yorumlar bu bölümün altında deneyler bittikten sonra yapılmıştır.

Çizelge 3.2 numaralı tabloda aşağıda epoch sayısı ve paket boyutuna göre koşum sonrası alınan kayıp/doğruluk grafiklerinin numerik değerleri verilmiştir:

Momentum: M
Öğrenme Katsayısı: ÖK

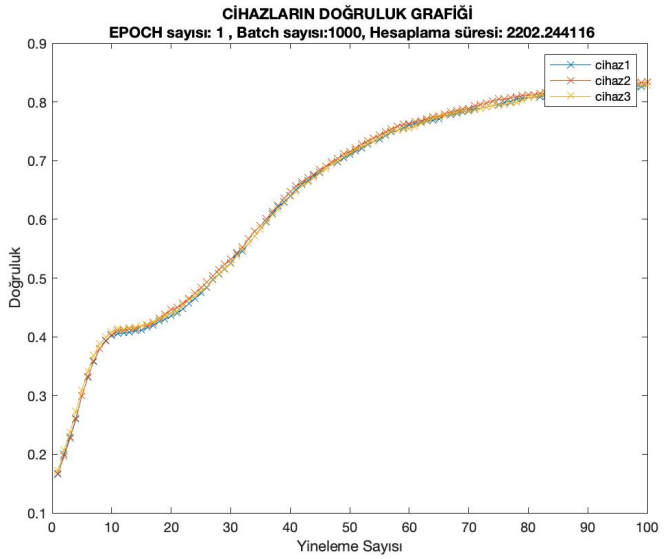
Epoch Sayısı	Paket Boyutu	Kayıp Değeri	Doğruluk Değeri	Hesaplama Süresi	Her Bir Cihazdaki	Cihaz Sayısı	M/ÖK

					Veri Sayısı		
1	1000	%56,28	%85,30	2202	2000	3	M
4	1000	%33,49	%89,97	5828	2000	3	M
4	100	%20,43	%95,87	5504	2000	3	M
1	100	%16,45	%95,57	2265	2000	3	M

Çizelge 3.2. Epoch sayısı çalışmalarına yönelik kayıp/doğruluk değerleri

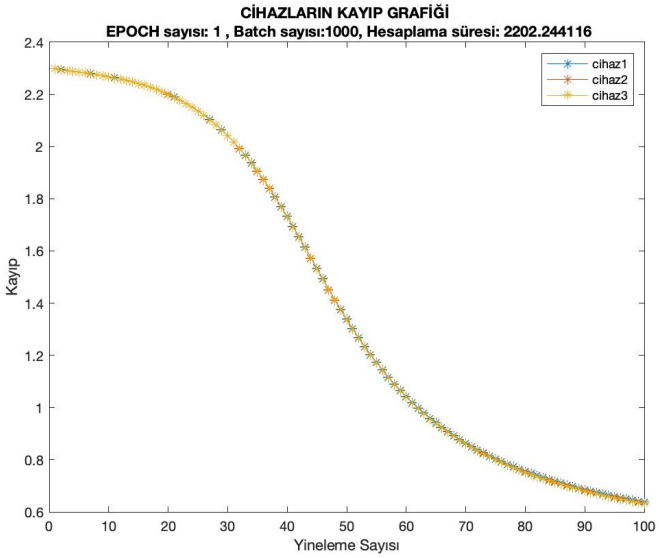
3.1.2.1. Epoch Sayısı=1 ve Paket Sayısı=1000

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



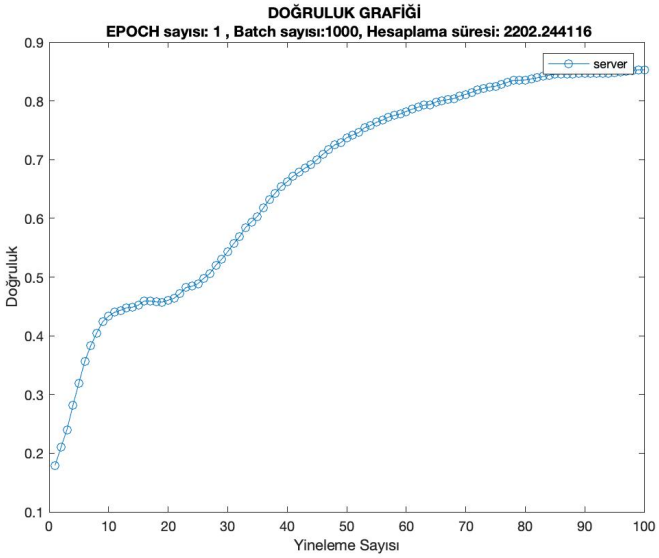
Şekil 3.21. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



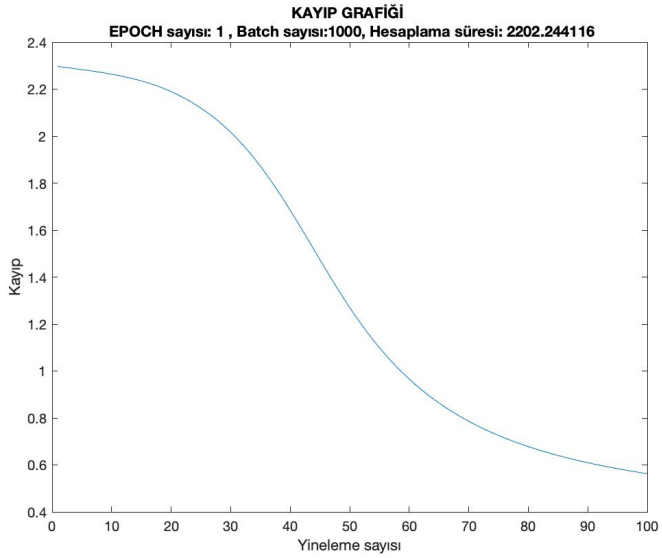
Şekil 3.22. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.23. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

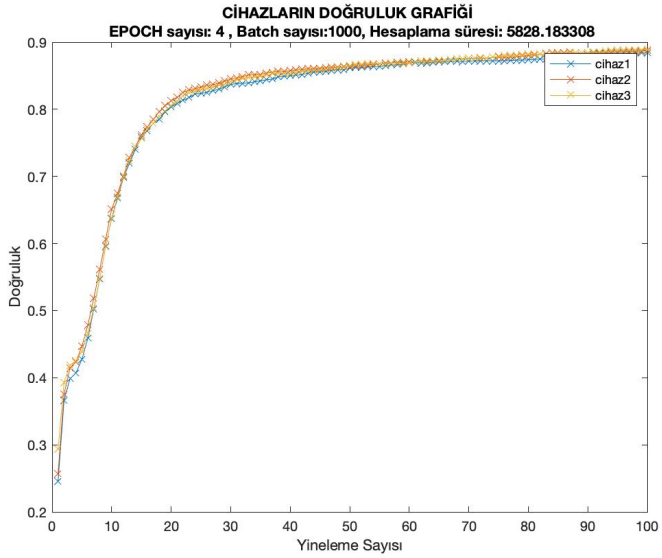
Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.24. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

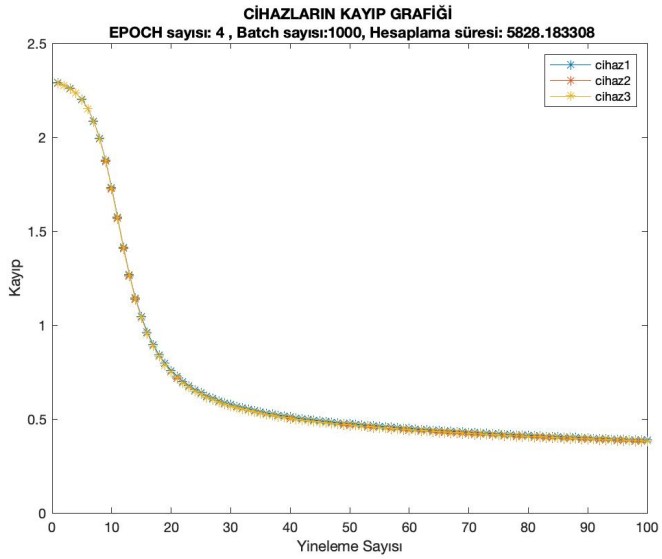
3.1.2.2. Epoch Sayısı=4 ve Paket Sayısı=1000

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



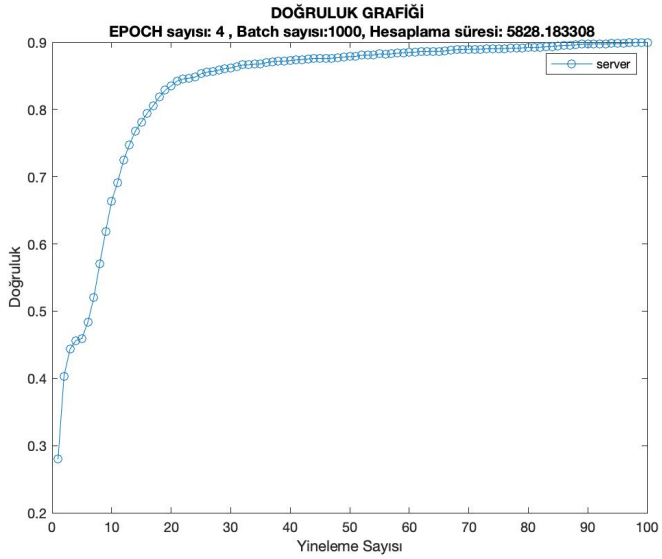
Şekil 3.25. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



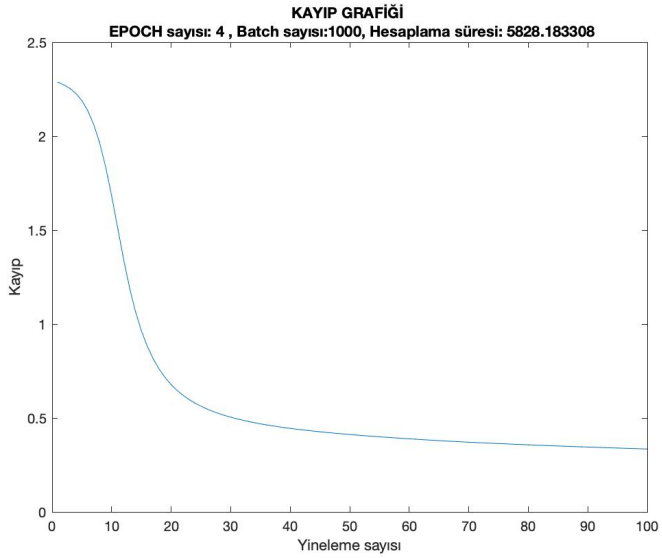
Şekil 3.26. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.27. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

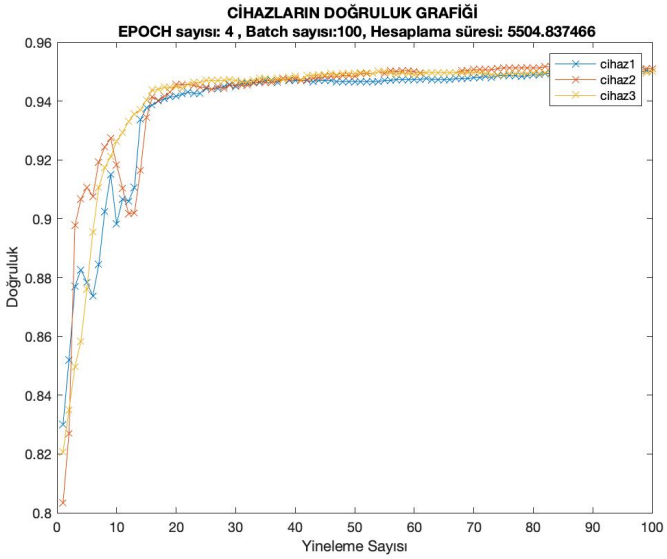
Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.28. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

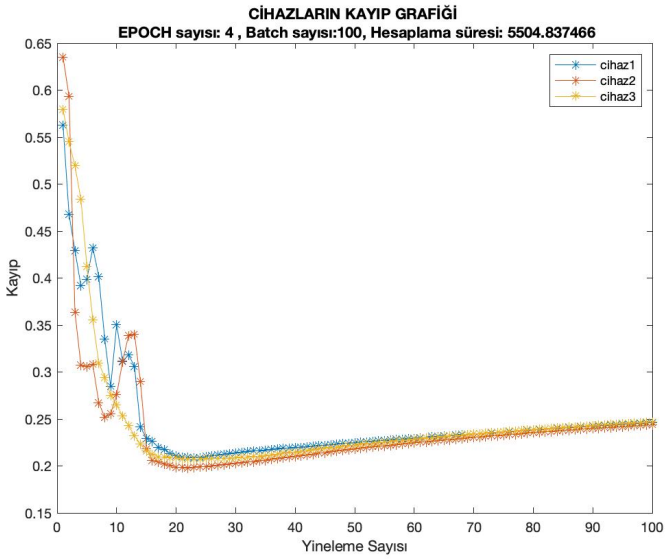
3.1.2.3. Epoch Sayısı=4 ve Paket Sayısı=100

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



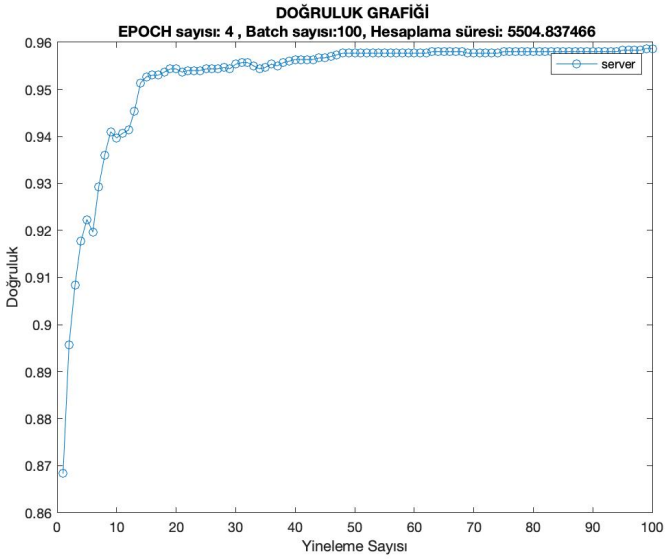
Şekil 3.29. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



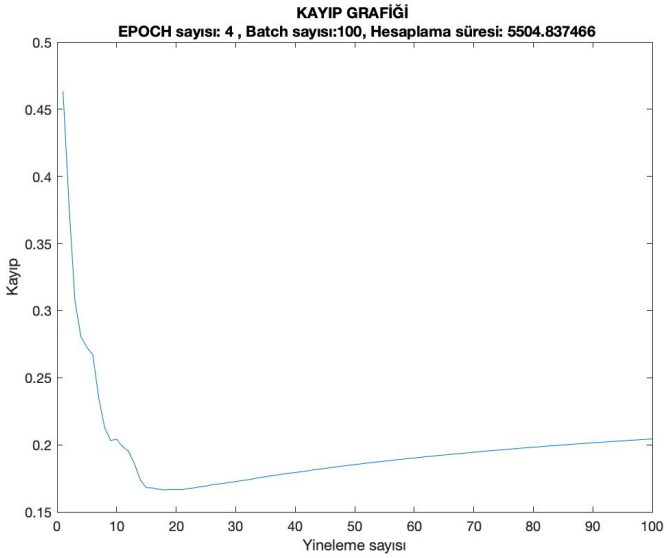
Şekil 3.30. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.31. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

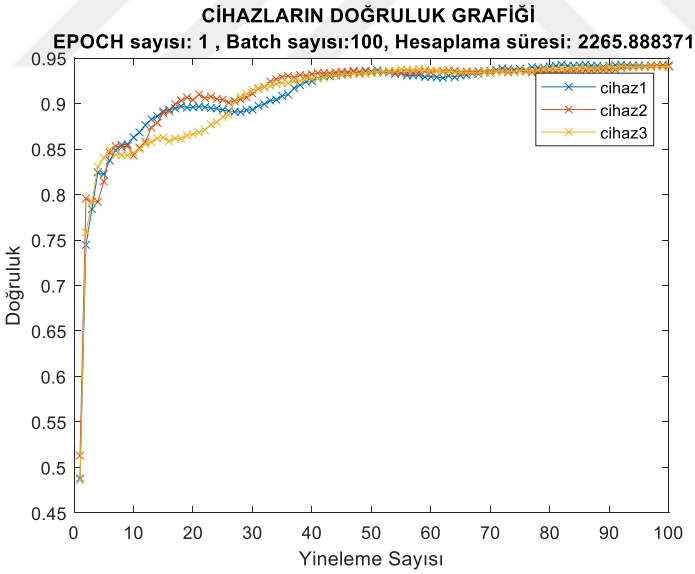
Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.32. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

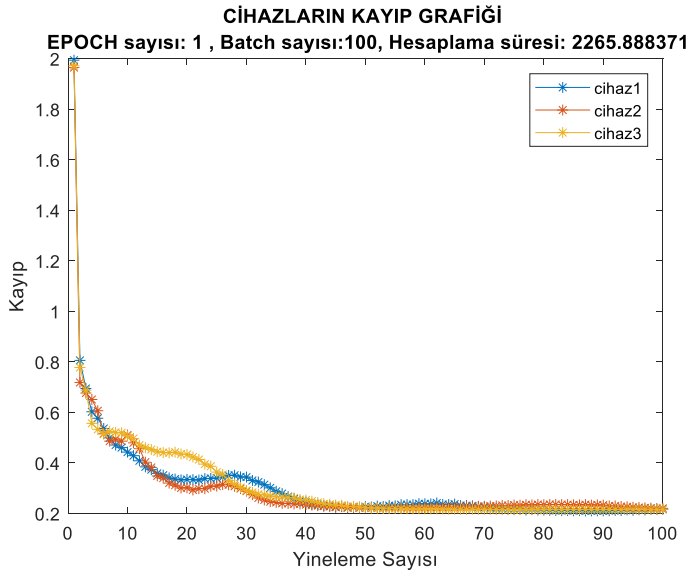
3.1.2.4. Epoch Sayısı=1 ve Paket Sayısı=100

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



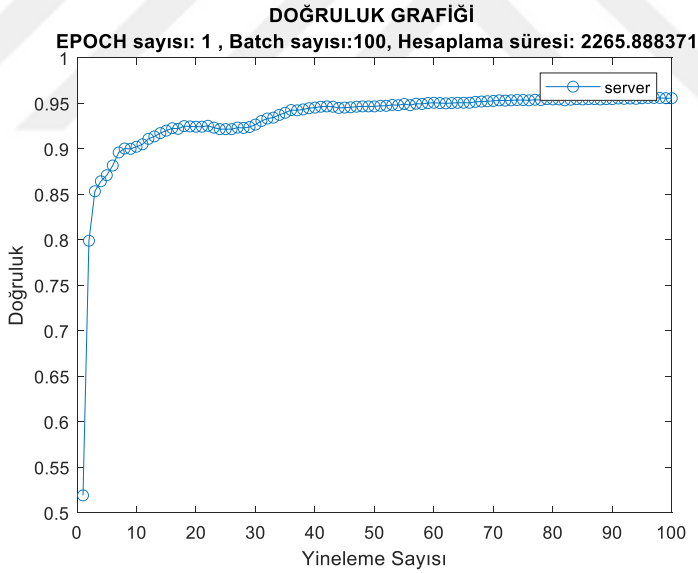
Şekil 3.33. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



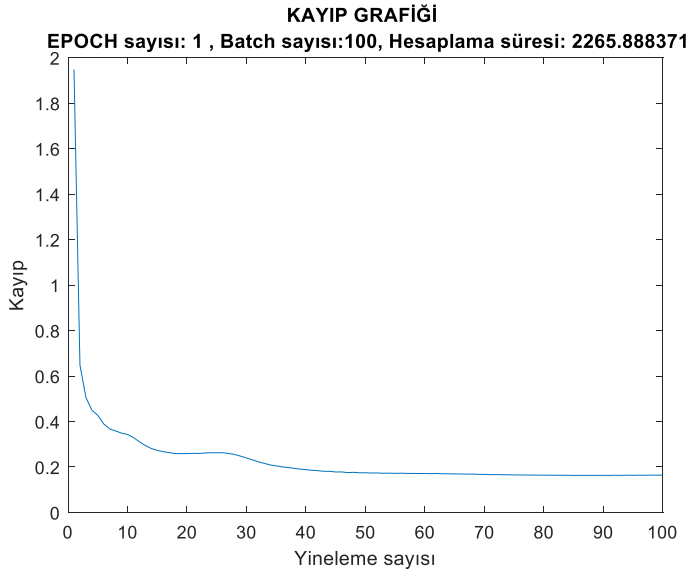
Şekil 3.34. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.35. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.36. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

Bölüm 3.1.2.3'te incelenen grafiklerden Şekil 3.29'da cihazlarda gürültü gözlemlenmiştir. Şekil 3.30'da cihazların ve Şekil 3.32.'de sunucunun aşırı öğrenmeye gösterdiği gözlemlenebilir. Sunucunun ve cihazların kayıp değerinin minimum noktaya ulaştıktan, sonra kaybın giderek arttığını gözlemlenmektedir. Bu durumda federe öğrenme sisteminin performansını arttırmak amacıyla parametreleri en uygun duruma ayarlayarak, aşırı öğrenmenin önüne geçilebilir. Bölüm 3.1.2.4'te epoch sayısı 1 olarak değiştirildiğinde, sistemde ve cihazlarda aşırı öğrenmenin azaldığı, sistemin yakınsamaya meyilli olduğu gözlemlenebilir. Bu durumda epoch sayısının değişmesine alternatif olarak başka parametrelerde değişim yapılarak sistem performansı düzenlenebilir.

Epoch sayısının sistem üzerindeki performansını değerlendirmek amacıyla yüksek paket boyutlarında da yorum yapılabilir. Paket boyutunu 1000 olarak değiştirerek sistem performansını incelenmek istenildiğinde; bölüm 3.1.2.1'de paket boyutu büyük olduğu için öğrenme çözünürlüğü düşmüştür. Bununla beraber, epoch sayısı da düşük olduğu için sistem öğrenmesi grafiklerde görüldüğü üzere, uygun değere yakınsayamamıştır. Ek olarak, sistemdeki gürültü de paket boyutunun yüksek olmasından dolayı düşük, geçişler daha yumuşaktır. Fakat hata oranı yüksek, doğruluk oranı düşük kalmıştır.

Bölüm 3.1.2.2'de epoch değerini, paket boyutu büyük seçildiği zaman arttırıldığında, paket boyutu düşük duruma göre daha sağlıklı sonuçlar elde edilmiştir. Öğrenmesini düşük epoch sayısıyla tamamlayamayan sistem, epoch sayısının da artmasıyla uygun

değere yakınsamaya başlamış, doğruluk oranı yükselirken kayıp değerinde de düşüş gözlemlenmiştir.

Hesaplama süresi değerlendirmesinde ise yüksek epoch sayısının sistemde gecikmeye sebep olduğu, iterasyonun öğrenmeyi arttırırken, hesaplama süresini de arttırdığı söylenilebilir.

3.1.3. Cihaz Sayısı Etkisi

Bu bölümdeki çalışmalarda federe öğrenmede sık karşılaşılan bir durum olan farklı cihaz sayısının öğrenmeye etkisi incelenecektir. Toplam 10 adet cihazın var olduğu varsayılmıştır. Bölüm 3.1.3.1'de var olan bütün cihazlar aktif olarak kabul edilirken, bölüm 3.1.3.2'de 3 cihaz aktifken 7 cihazın bağlantısının koptuğu, bölüm 3.1.3.3'de 1 cihaz aktifken 9 cihazın öğrenmeye katılamadığı kabul edilmiştir. 3 deney için de cihaz sayısı dışındaki parametreler sabit ve epoch sayısı da aşırı öğrenmeyi önleme amaçlı düşük tutulmuştur.

Her deney için katılımcıların ayrı ayrı kayıp ve doğruluk grafiği, sistemin genel kayıp ve doğruluk grafiği, görselleriyle birlikte eklenerek, bölüm sonunda yorumlarda bulunulmuştur.

Çizelge 3.3 numaralı tabloda aşağıda aktif olan cihaz sayısına göre koşum sonrası alınan kayıp/doğruluk grafiklerinin numerik değerleri verilmiştir:

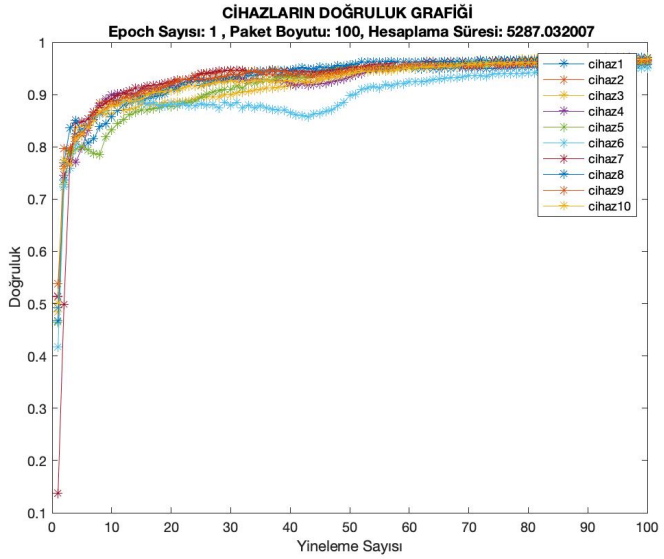
Momentum: M
Öğrenme Katsayısı: ÖK

Cihaz Sayısı	Kayıp Değeri	Doğruluk Değeri (%)	Hesaplama Süresi	Paket Boyutu	Her Bir Cihazdaki Veri Sayısı	Epoch Sayısı	M/ÖK
10	**%9,87**	**%97,10**	**5287**	100	2000	1	M
3	**%16,45**	**%95,57**	**2265**	100	2000	1	M
1	**%30,77**	**%93,20**	**948**	100	2000	1	M

Çizelge 3.3. Paket boyutu çalışmalarına yönelik kayıp/doğruluk değerleri

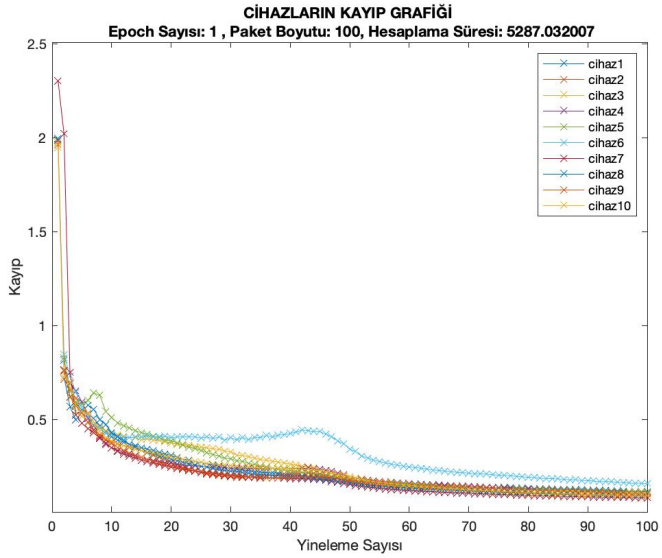
3.1.3.1. Cihaz Sayısı=10

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 10 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



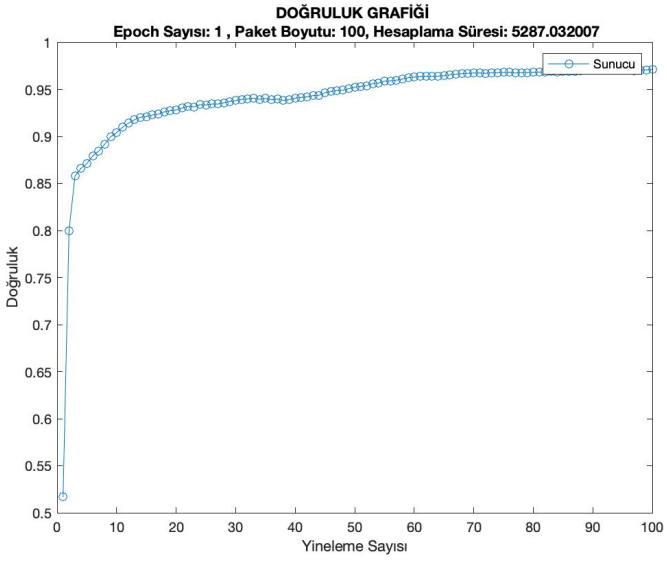
Şekil 3.37. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 10 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



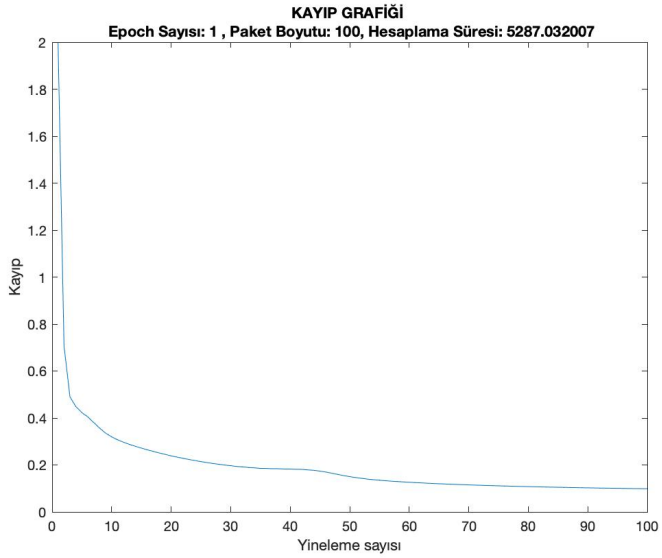
Şekil 3.38. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 10 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.39. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

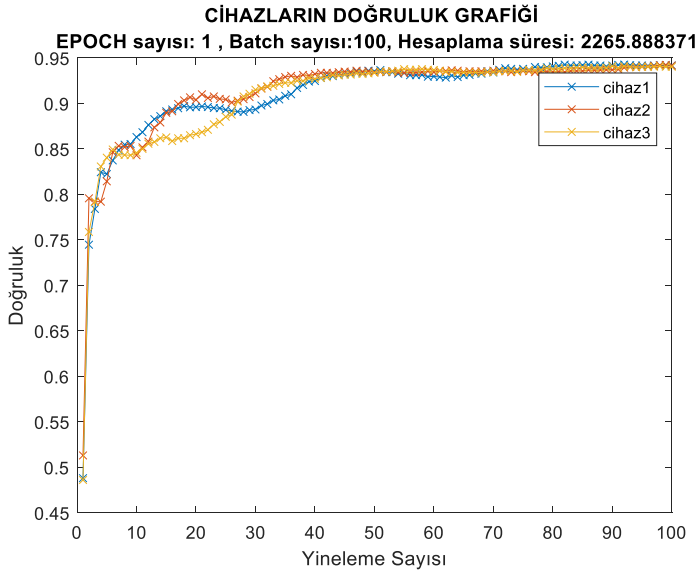
Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 10 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.40. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

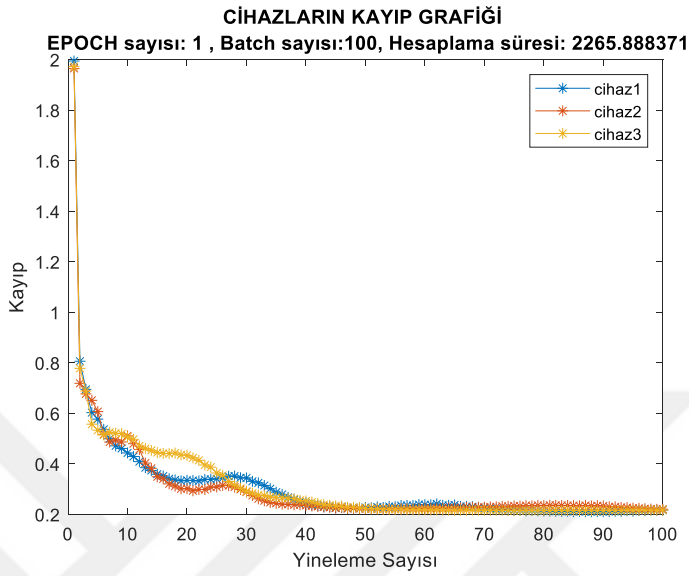
3.1.3.2. Cihaz Sayısı=3

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



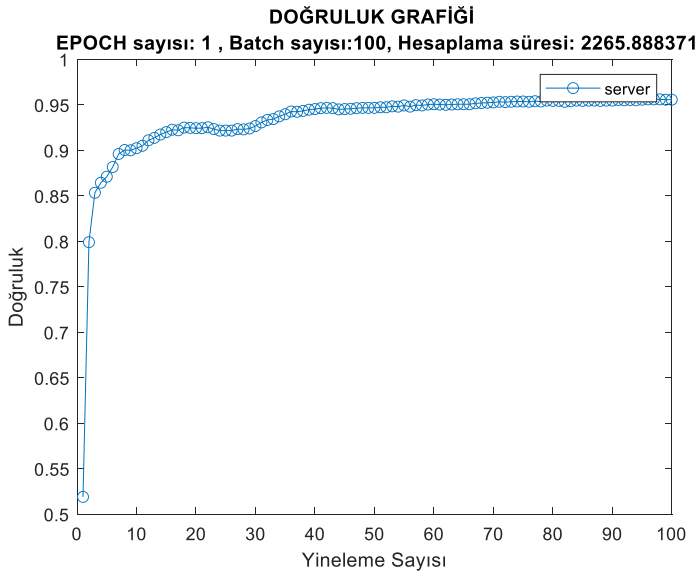
Şekil 3.41. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



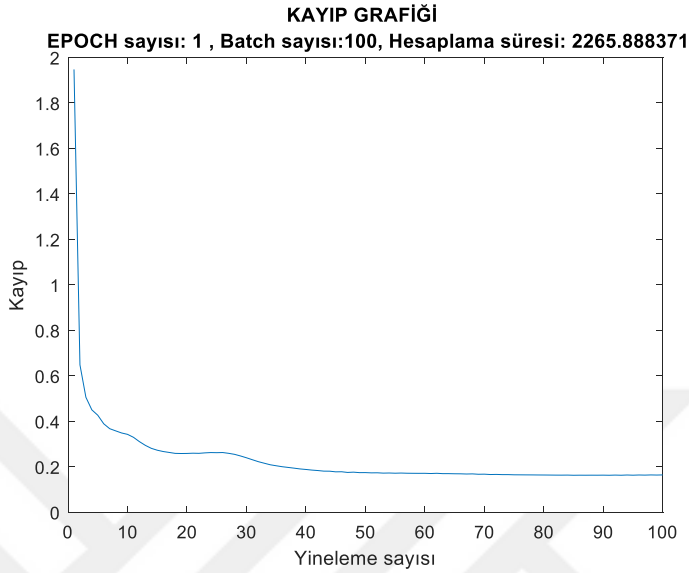
Şekil 3.42. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.43. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

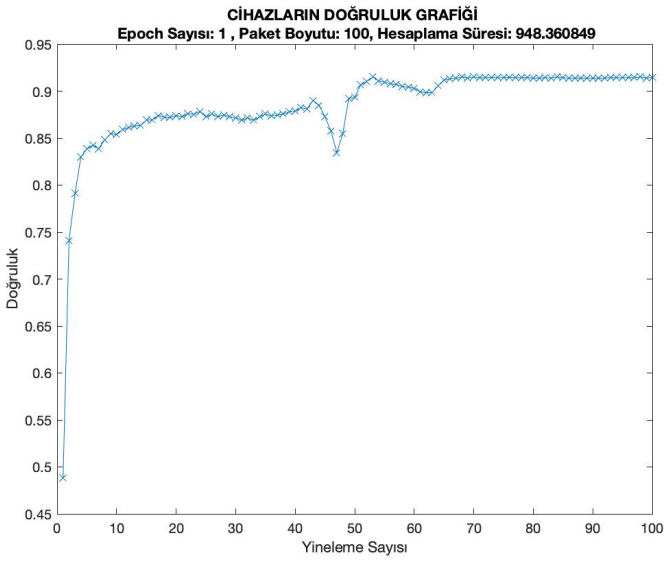
Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.44. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

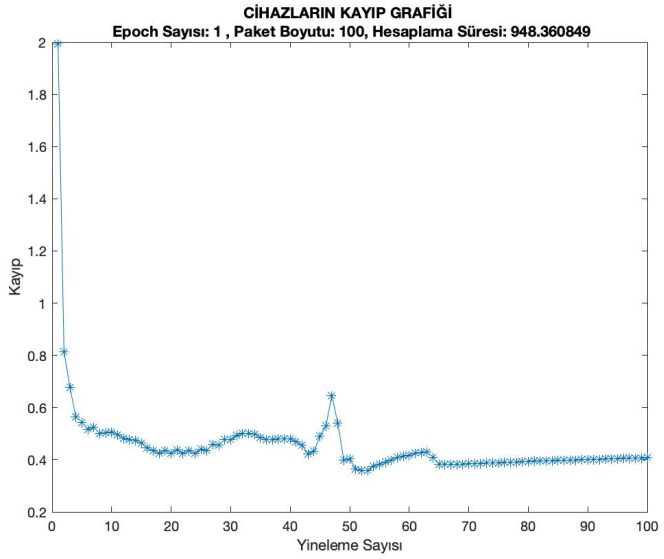
3.1.3.3. Cihaz Sayısı=1

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve katılımcıda homojen olarak dağıtılmış veri var iken, 1 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



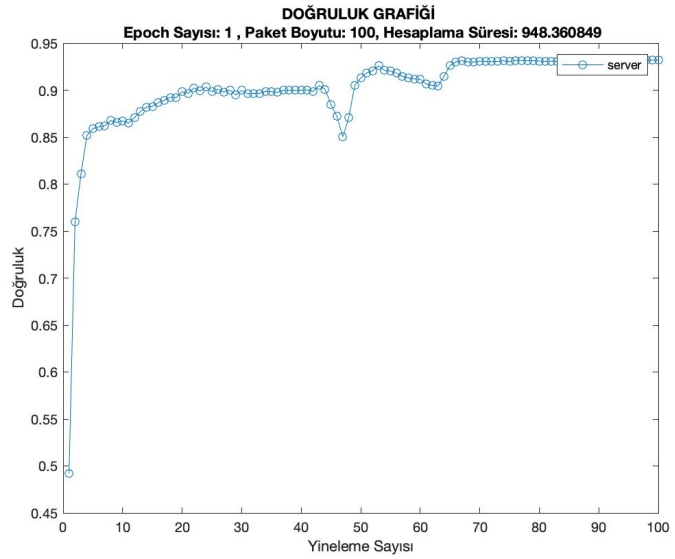
Şekil 3.45. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve katılımcıda homojen olarak dağıtılmış veri var iken, 1 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



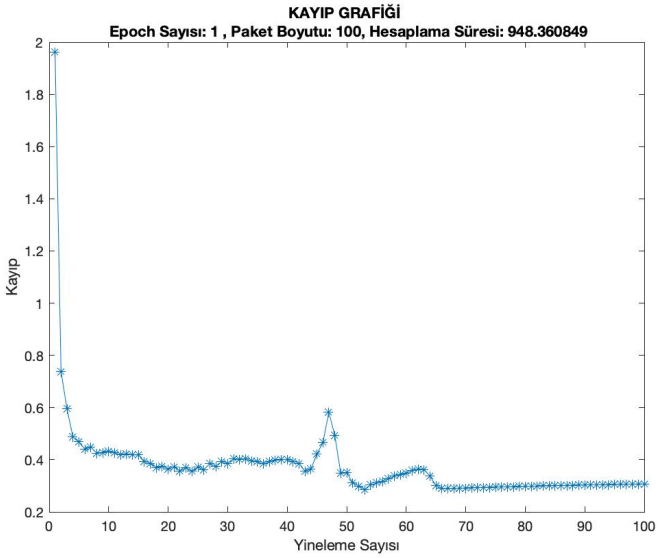
Şekil 3.46. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve katılımcıda homojen olarak dağıtılmış veri var iken, 1 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.47. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 ve katılımcıda homojen olarak dağıtılmış veri var iken, 1 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.48. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

Federe öğrenmede aktif cihaz sayısı önemli bir parametredir. Federe öğrenmenin diğer yöntemlerden farklılığı, cihazların, yerellerinde oluşturduğu modellerle birlikte etkileşime girmesi sonucu, global gelişmiş bir model elde etmeyi hedefler. Bu da FÖ'de öğrenme çözünürlüğünü arttırdığından bölüm 2.1'de bahsedilmiştir. Bu konunun daha ayrıntılı incelenmesi için koşumlar farklı sayıda cihazların aktif edilerek tekrarlanması sağlanmıştır.

10 farklı cihaza sahip ve her birinin farklı, homojen dağılmış verisi olan, sistemin doğruluğu %97,10 ile diğer aktif cihaz sayılı sistemlerden daha iyi performans göstermiştir. Ayrıca kayıp oranı %9,87 ile en düşük olan sistem, 10 farklı cihazın da aktif olduğu sistemdir. Bunun sebebi, federe öğrenmenin belirgin bir özelliği olan, farklı cihazlardaki modelleri sunucuda toplayıp, eğitimin cihaz içinde sınırlandırmasını önlemesidir. 10 farklı cihaz demek her cihazın kendi verisini eğitmesi ve toplanan modelin geliştirilerek tekrardan sisteme dağıtılması demektir.

Tek 1 cihazın aktif olması,cihazın sadece kendi verisiyle eğitimine devam ettiği, başka cihazların modellerinden etkilenmediği sistemdir. Bu sistemde kayıp oranı yüksek, doğruluk oranı daha düşük kalmıştır.

Aktif cihaz sayısının doğruluğa katkısı büyük iken, hesaplama süresinin de büyüklüğü göz önüne alınmalıdır. Hesaplama süresi 1 aktif cihazdan, 10 aktif cihaza geçişinde, 10 kat arttığı gözlemlenmiştir. Doğruluğun yükselme oranı , hesap süresindeki artış oranı kadar yüksek olmamıştır. Federe öğrenmenin zorluğundan olan her farklı cihazın modelinin toplanıp sunucuda işleme sokulup tekrar dağıtılması hesaplama süresinde büyük bir artışa sebep olmaktadır. Bu durum sistemin öğrenmesinin yükselmesine katkıda bulunurken yavaşlamasına da sebep olmaktadır. Uygulanılan alanlara göre ödünleşim yapılarak en iyi durum seçimi yapılması mantıklıdır.

10 cihaz için sürenin bu kadar çok artmasıyla, iletişimdeki kopukluklara zemin hazırlaması çok olasıdır. Sürenin bu kadar uzun olması, batarya eksikliğinde aktif cihazların azalmasına sebep olabilir.

3.1.4. Heterojen Dağılım Etkisi

3.1.4.1. Homojen Veri ile Cihazlardaki Veri Sayısı Düzensizliği

Bölüm 2.3.2.1'de ayrıntılı olarak yer alan verinin her cihazda aynı boyutta bulunmaması veri düzensizliğine sebep olabilmektedir. Cihazların, gerçek hayatta birbirlerinden çok farklı teknolojiye ve hafızaya sahip olabildikleri için bu durum federe öğrenmede çözülmesi gereken zorluklardan olduğu söylenebilir . Aşağıdaki 2 örnekte cihazlara farklı sayıda homojen veriler dağıtılmış ve farklı epoch sayısı ile inceleme yapılmıştır.

Her deney için katılımcıların ayrı ayrı kayıp ve doğruluk grafiği, sistemin genel kayıp ve doğruluk grafiği, görselleriyle birlikte eklenerek açıklamalarda bulunulmuştur. Genel yorum bu bölümün sonunda yapılmıştır.

Yapılan deneyde, her cihaza farklı miktarda veri, homojen veri dağılımı ile dağılmıştır. Örneğin, 1. Cihazda 2000 adet veri vardır ve verinin içeriğinde 0'dan 9'a kadar olan

etiketler homojen miktarda dağıtılmıştır. 3. Cihazda 6000 adet veri vardır ve verinin içeriğinde 0'dan 9'a kadar olan etiketler homojen miktarda dağıtılmıştır. Aynı deney epoch sayısı 1 ve 4 olarak 2 adet olarak koşulmuştur.

Çizelge 3.4 numaralı tabloda düzensiz veri sayısı ile koşum sonrası alınan kayıp/doğruluk grafiklerinin numerik değerleri verilmiştir:

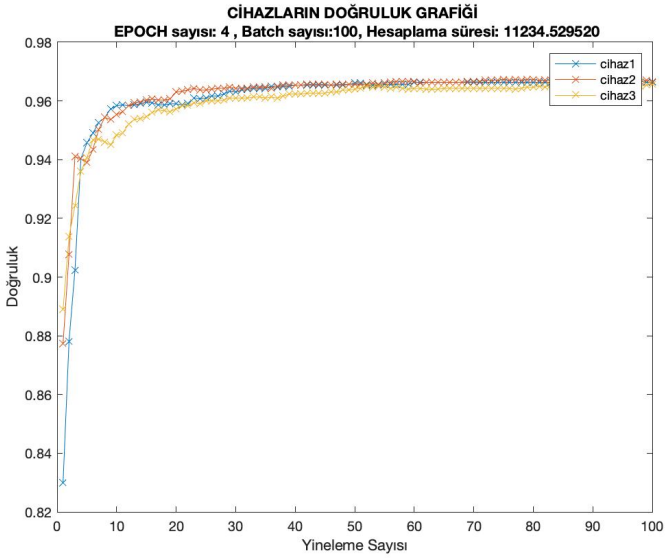
Momentum: M
Öğrenme Katsayısı: ÖK

Epoch Sayısı	**Kayıp Değeri**	**Doğruluk Değeri (%)**	**Hesaplama Süresi**	**Paket Boyutu**	**Her Bir Cihazdaki Veri Sayısı**	**Cihaz Sayısı**	**M/ÖK**
4	**%13,04**	**%97,50**	**11235**	100	2000-4000-6000	3	M
1	**%11,48**	**%97,07**	**3290**	100	2000-4000-6000	3	M

Çizelge 3.4. Homojen veri ile cihazlardaki veri sayısı düzensizliği çalışmalarına yönelik kayıp/doğruluk değerleri

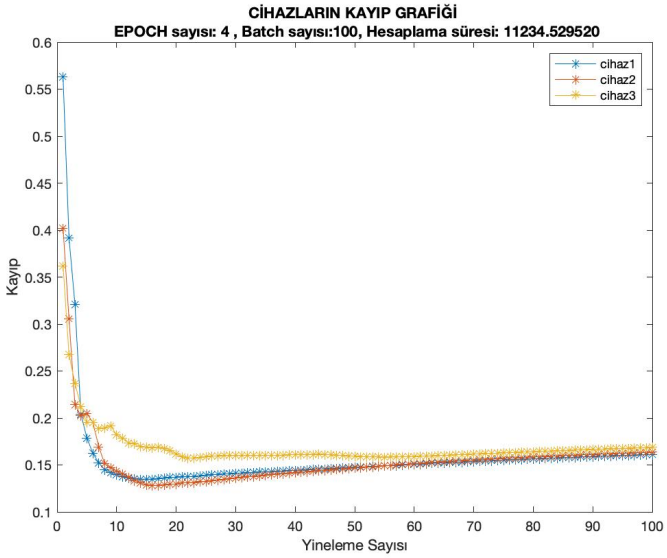
Düzensiz Dağılmış Veri Epoch=4:

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 olan her katılımcıya veri düzensiz olarak dağıtılmış. Her cihazın veri sayısı farklı olarak şu şekilde belirlenmiştir: 1. katılımcı için 2000 veri, 2. katılımcı için 4000 veri, 3. katılımcı için 6000 veri. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.49. Bütün katılımcıların doğruluk grafiği

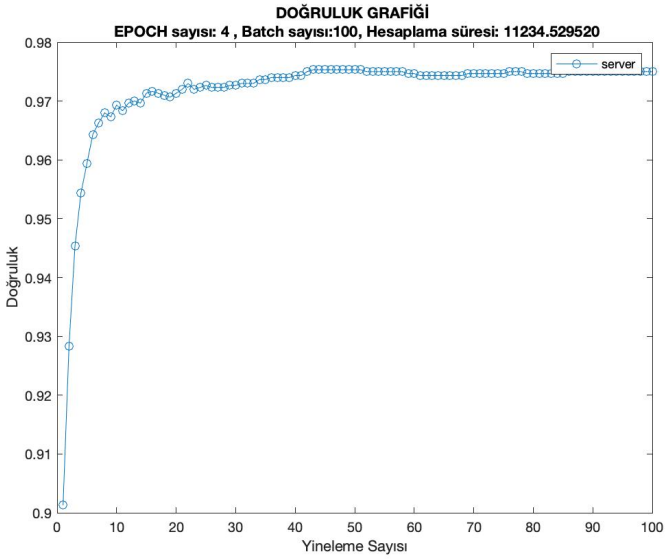
Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 olan her katılımcıya veri homojen olarak dağıtılmış. Fakat her cihazın veri sayısı farklı olarak şu şekilde belirlenmiştir: 1. katılımcı için 2000 veri, 2. katılımcı için 4000 veri, 3. katılımcı için 6000 veri. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.50. Bütün katılımcıların kayıp grafiği

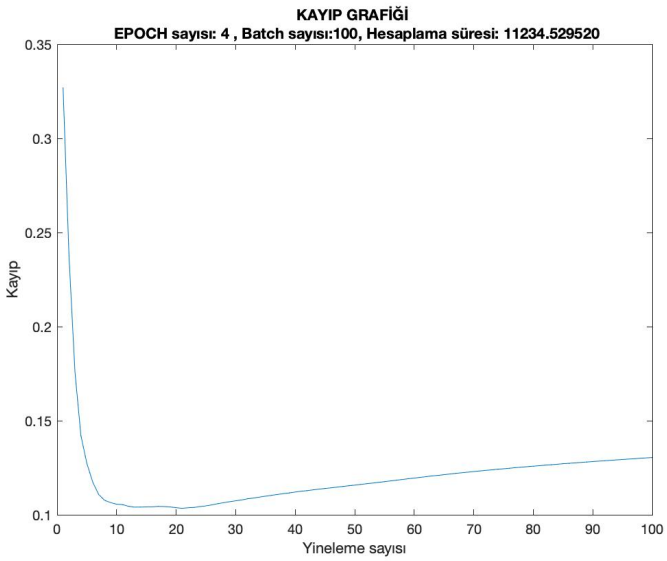
Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 olan her katılımcıya veri homojen olarak dağıtılmış. Fakat her cihazın veri sayısı farklı olarak şu şekilde belirlenmiştir: 1.

katılımcı için 2000 veri, 2. katılımcı için 4000 veri, 3. katılımcı için 6000 veri. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.51. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

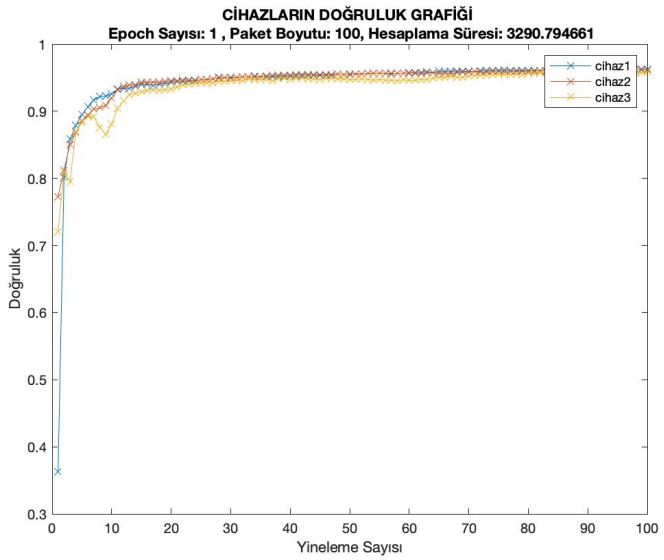
Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 olan her katılımcıya veri homojen olarak dağıtılmış. Fakat her cihazın veri sayısı farklı olarak şu şekilde belirlenmiştir: 1. katılımcı için 2000 veri, 2. katılımcı için 4000 veri, 3. katılımcı için 6000 veri. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.52. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

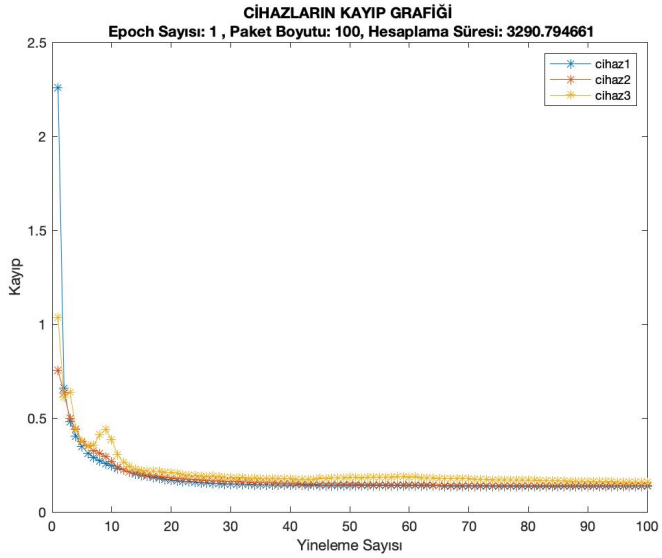
Düzensiz Dağılmış Veri Epoch=1:

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 olan her katılımcıya veri homojen olarak dağıtılmış. Fakat her cihazın veri sayısı farklı olarak şu şekilde belirlenmiştir: 1. katılımcı için 2000 veri, 2. katılımcı için 4000 veri, 3. katılımcı için 6000 veri. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



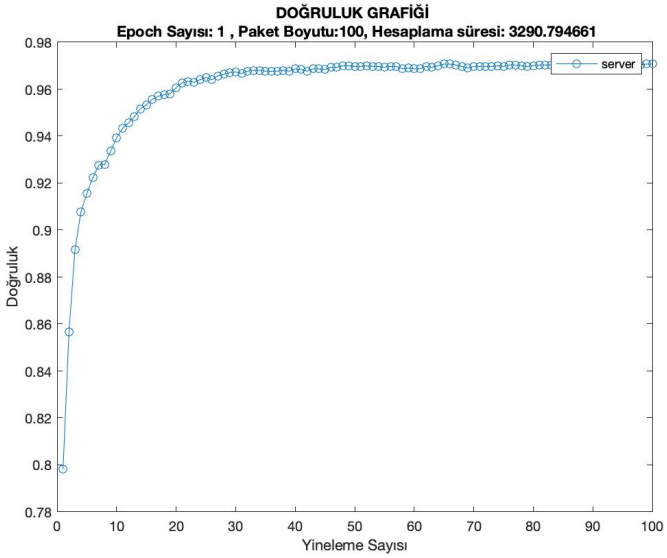
Şekil 3.53. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 4 olan her katılımcıya veri homojen olarak dağıtılmış. Fakat her cihazın veri sayısı farklı olarak şu şekilde belirlenmiştir: 1. katılımcı için 2000 veri, 2. katılımcı için 4000 veri, 3. katılımcı için 6000 veri. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.54. Bütün katılımcıların kayıp grafiği

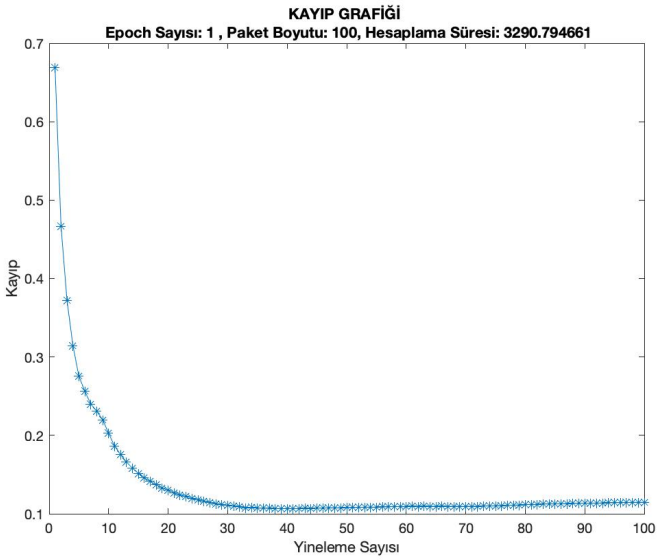
Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 olan her katılımcıya veri homojen olarak dağıtılmış. Fakat her cihazın veri sayısı farklı olarak şu şekilde belirlenmiştir: 1. katılımcı için 2000 veri, 2. katılımcı için 4000 veri, 3. katılımcı için 6000 veri. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.55. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1 olan her katılımcıya veri homojen olarak dağıtılmış. Fakat her cihazın veri sayısı farklı olarak şu şekilde belirlenmiştir: 1.

katılımcı için 2000 veri, 2. katılımcı için 4000 veri, 3. katılımcı için 6000 veri. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.56. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

Bu deneyde epoch sayısının değişimi incelendiğinde doğruluk oranında büyük bir artış gözlemlenemezken, hesap süresinde ciddi oranda bir artış gerçekleşmiştir. Doğruluk oranının büyük oranda artmamasının sebebi, 3 farklı cihazdaki toplam veri sayısı 12000 adet olarak belirlenmiştir. 2 deneyde de veri sayısının fazla olmasıyla sistemin öğrenmesi için gerekli veri sağlanmış olup, doğruluk da yakınsamış bulunmaktadır. Veri sayısının öğrenme için yeterli olduğu düşünülürse epoch sayısının arttırılması, hesaplama süresindeki uzamaya ve sistemin aşırı öğrenmeye yatkınlığından dolayı artan kayba sebep olabilir.

Epoch sayısı 1 olarak seçildiğindeki sistemin kayıp değeri %11,48 değerine yakınsayarak düşük bir kayıp elde ettiği söylenilebilir. Düşük kayıp değerinde etkili olan etkenler; paket boyutunun 100 olarak seçilmiş olması ve toplam veri sayısının da 12000 gibi bir sayı olması olmasıdır. Epoch sayısı 4'e çıkarıldığında kayıp değeri %13,04 olarak 100. yinelemede görülmüş daha fazla artış sağlanırsa aşırı öğrenmeden dolayı artması da beklenmektedir. Sistemin hesaplama süresi, 1 epochtan 4 epoch sayısına geçişte, 3 katın üzerinde bir artış gözlemlenmişttir, Fakat doğruluk değeri neredeyse sabit kalmıştır.

Yukarıdaki paragraflarda sistemin genel performansı incelenmiştir. Cihazların veri düzensizliğinden olan cihazlardaki performansı incelenmemiştir. 1. cihazdaki veri sayısı daha az olduğu için yinelemenin başında düşük doğruluk oranı ile başlamış daha sonra 2. ve 3. Cihazın modellerinin birleşimiyle aynı değere yakınsamayı başarmışlardır.

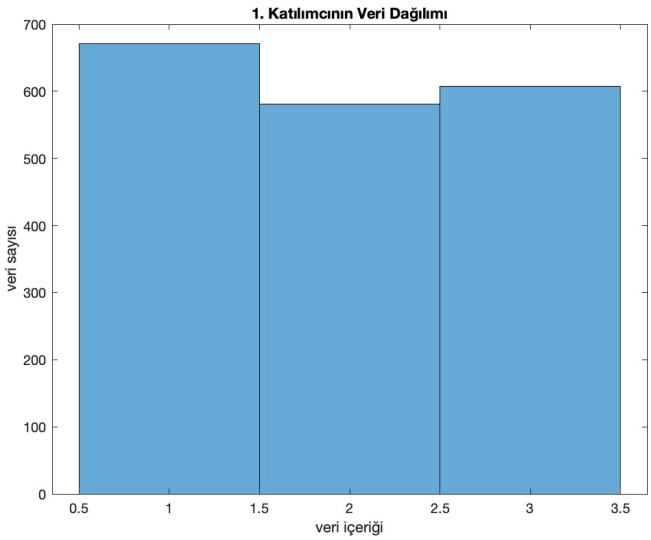
4 epoch sayısı ile yapılan deneyde eğitimin tamamlanma süresi 11235 gibi bir sayıdır. 3.1.2.3'de eşit olarak verinin homojen olarak dağıtılması deneyinde bu sayı yaklaşık 5500'dür. Aradaki zaman farkı, verisi az olan cihaz için önemli ölçüde fark ettirmektedir. Bu durumda gerçek hayatta verisi az olan cihaz, öğrenimini daha erken bitirebilir gibi düşünülmesine rağmen, veri sayısı kendisine göre daha fazla olan cihazları beklerken vakit kaybetmektedir. Veri sayısının daha az olduğu düşünülen 1. cihazın teknolojisi 2. ve 3. cihaza göre daha eskiyse uzun eğitim süresinden dolayı bağlantı sorunlarına ya da batarya eksikliğinden iletişimin kopmasına sebep olabilir. Bu durum sistem heterojenliğinin getirdiği zorluğa bir örnektir.

3.1.4.1. Veri Heterojenlik-1

Bölüm 3.1.4'de heterojen dağılım etkisi inceleme bölümünde 2 farklı adet veri heterojenlik ele alınacaktır. Bu paragraf, veri heterojenliği-1 ve veri heterojenliği-2 arasındaki farkı ortaya koymayı hedeflemektedir. veri heterojenliği-1 bölümünde cihazlardaki veri tamamen birbirlerinden farklı iken, veri heterojenliği-2 bölümünde cihazlardaki veri birbirleriyle aynı olup verinin dağılım şekilleri farklıdır. Daha ayrıntılı incelemek için katılımcıların veri içeriği-veri sayısı grafikleri incelenebilir.

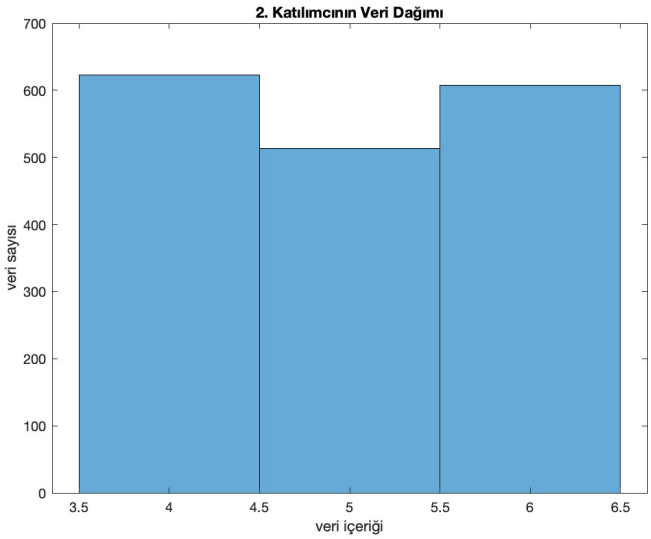
Aşağıda veri heterojenliğinin etkisi incelenmiştir. Öğrenmeye girecek her aktif cihazda, diğer cihazlardan farklı tip veri olduğu kabul edilmiştir. Bu bölümde yapılan deneylerde veri sayısının cihaz içlerindeki dağılımı Şekil 3.57., Şekil 3.58., Şekil 3.59. görsellerinden incelenebilir. Aşağıda yapılmış her deney için katılımcıların ayrı kayıp ve doğruluk grafiği, sistemin genel kayıp ve doğruluk grafiği, görselleriyle birlikte eklenerek açıklamalarda bulunulmuştur. Yorumlar bu bölümün sonunda yer almaktadır.

Şekil 3.57. 1.Katılımcının hangi verilerden kaç adet veri sayısı olduğunu içermektedir.



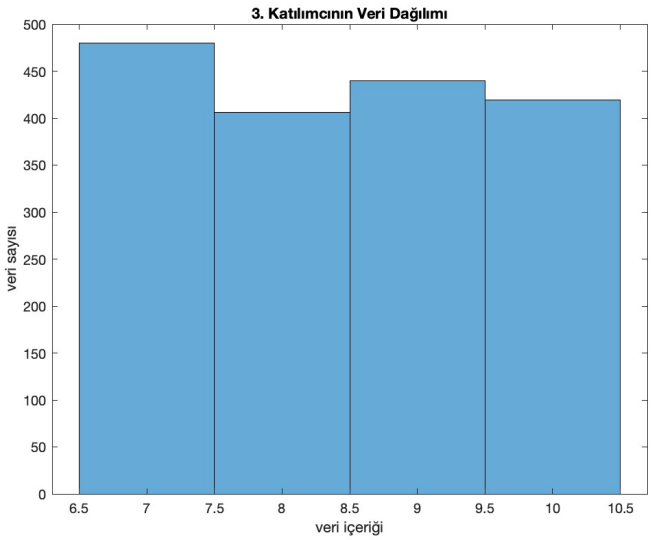
Şekil 3.57. 1. Katılımcının veri içeriği-veri sayısı grafiği

Şekil 3.58. 2.Katılımcının hangi verilerden kaç adet veri sayısı olduğunu içermektedir.



Şekil 3.58. 2. Katılımcının veri içeriği-veri sayısı grafiği

Şekil 3.59. 3.Katılımcının hangi verilerden kaç adet veri sayısı olduğunu içermektedir.



Şekil 3.59. 3. Katılımcının veri içeriği-veri sayısı grafiği

Çizelge 3.5 numaralı tabloda heterojen verilerle oluşturulmuş istemcilerle yapılan koşum sonrası alınan kayıp/doğruluk grafiklerinin numerik değerleri verilmiştir:

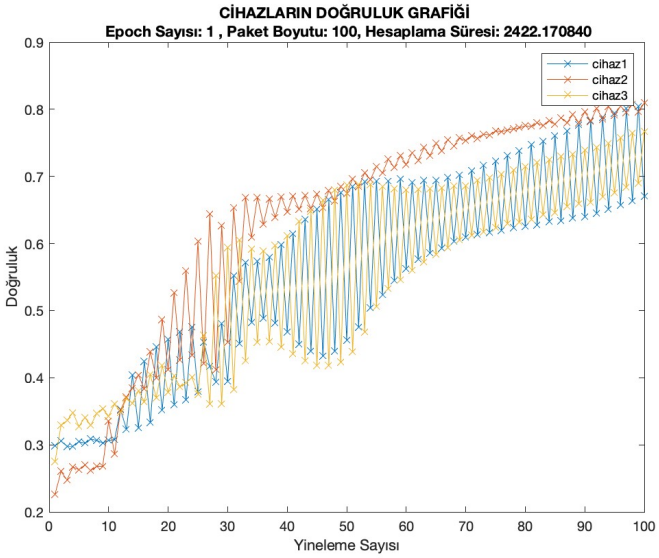
Momentum: M
Öğrenme Katsayısı: ÖK

Epoch Sayısı	**Kayıp Değeri**	**Doğruluk Değeri (%)**	**Hesaplama Süresi**	**Paket Boyutu**	**Her Bir Cihazdaki Veri Sayısı**	**Cihaz Sayısı**	**M/ÖK**
3	**%17,48**	**%95,37**	**4496**	100	1860-1745-2395	3	M
1	**%17,87**	**%94,60**	**2422**	100	1860-1745-2395	3	M

Çizelge 3.5. Heterojen veri dağılımı çalışmalarına yönelik kayıp/doğruluk değerleri

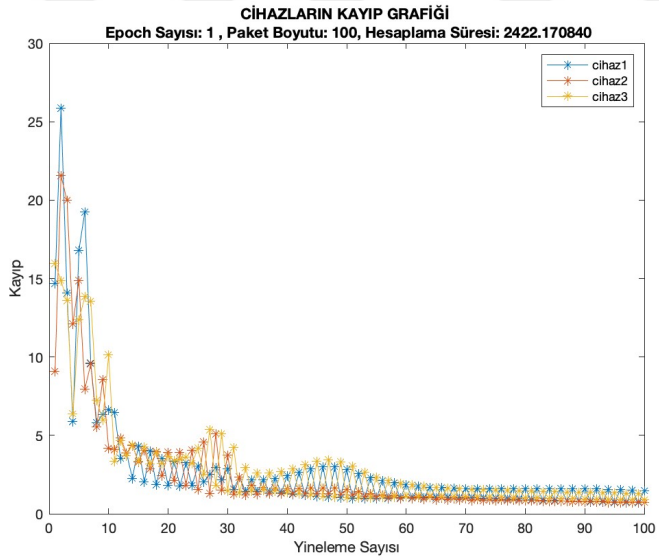
Heterojen Dağılmış Veri ve Epoch=1:

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1'dir. Her katılımcıda veri heterojen olarak dağıtılmış olup, her cihazın farklı sınıflandırma verisi bulunmaktadır. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



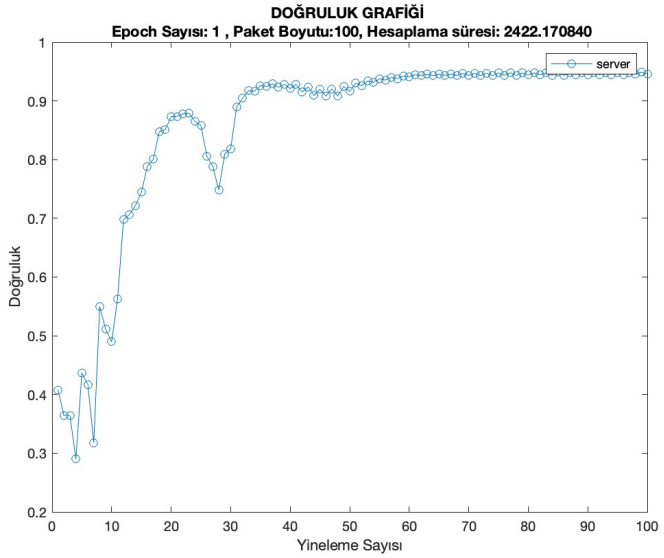
Şekil 3.60. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1'dir. Her katılımcıda veri heterojen olarak dağıtılmış olup, her cihazın farklı sınıflandırma verisi bulunmaktadır. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



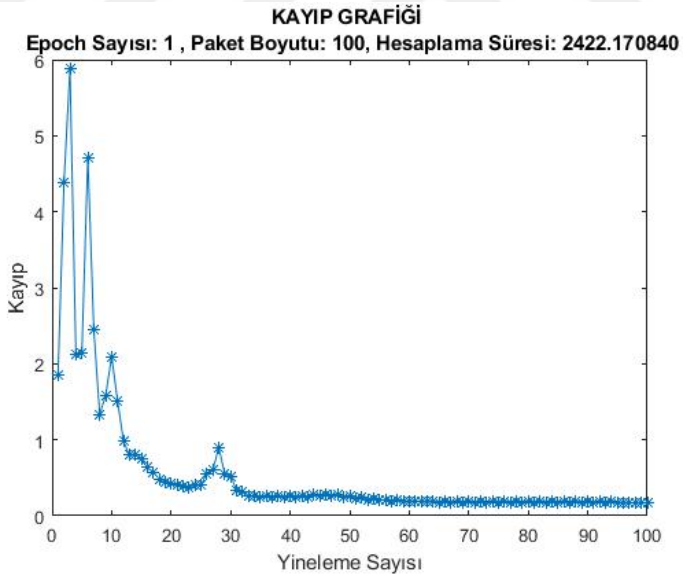
Şekil 3.61. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1'dir. Her katılımcıda veri heterojen olarak dağıtılmış olup, her cihazın farklı sınıflandırma verisi bulunmaktadır. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.62. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

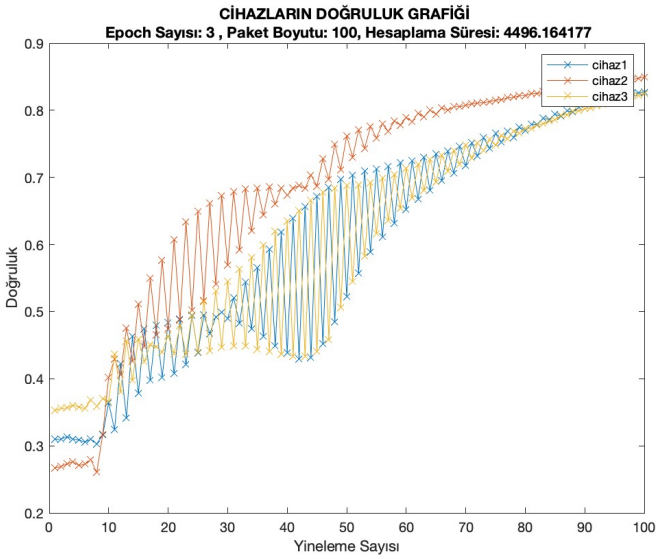
Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1'dir. Her katılımcıda veri heterojen olarak dağıtılmış olup, her cihazın farklı sınıflandırma verisi bulunmaktadır. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.63. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

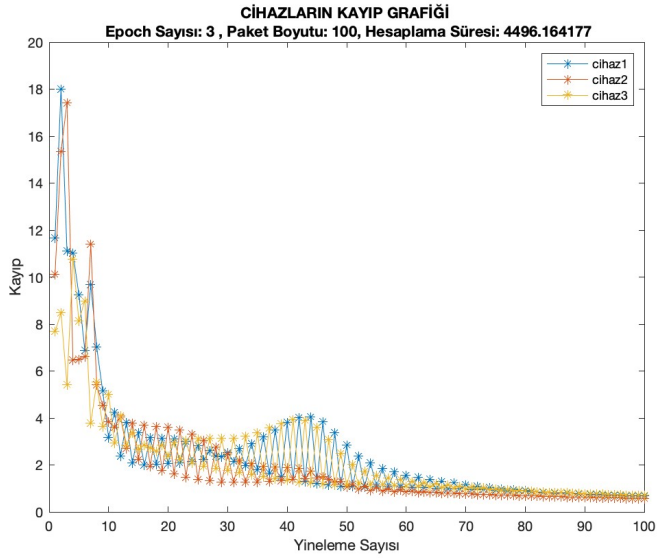
Heterojen Dağılmış Veri ve Epoch=3:

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 3'tür. Her katılımcıda veri heterojen olarak dağıtılmış olup, her cihazın farklı sınıflandırma verisi bulunmaktadır. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



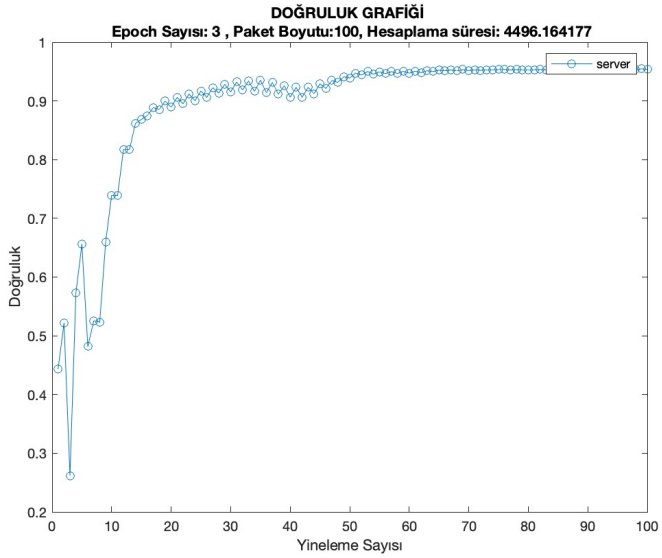
Şekil 3.64. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 3'tür. Her katılımcıda veri heterojen olarak dağıtılmış olup, her cihazın farklı sınıflandırma verisi bulunmaktadır. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



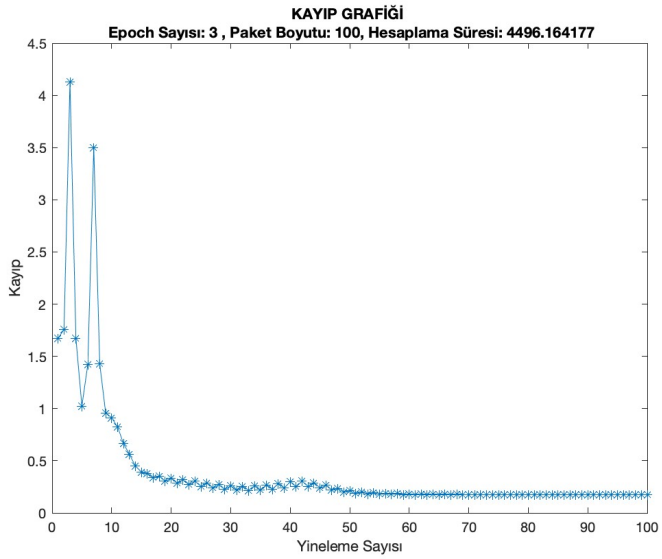
Şekil 3.65. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 3'tür. Her katılımcıda veri heterojen olarak dağıtılmış olup, her cihazın farklı sınıflandırma verisi bulunmaktadır. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.66. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 3'tür. Her katılımcıda veri heterojen olarak dağıtılmış olup, her cihazın farklı sınıflandırma verisi bulunmaktadır. 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.67. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

Bu deneylerde oluşturulan grafikler, 3 farklı cihaza, farklı veriler dağıtılması yardımıyla oluşturulmuştur. 10 farklı cihazdan 7 tanesinin o sırada pasif olduğu, 3 farklı cihazla öğrenime başlandığı kabul edilmiştir. Bu bölümde yapılan deneyde, veriler cihazlara

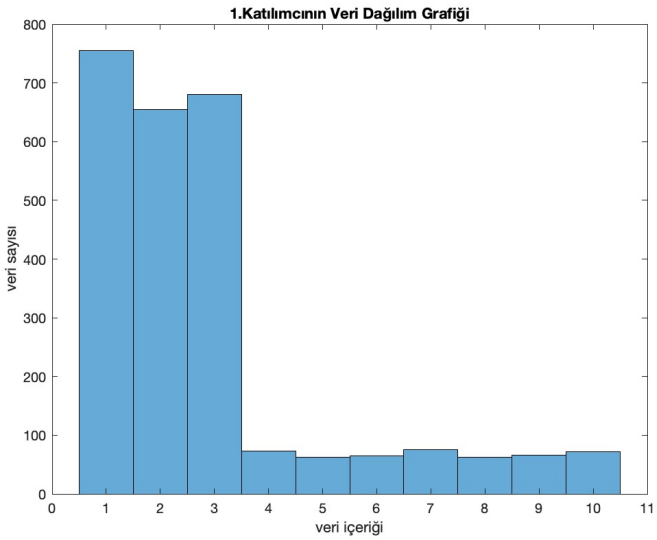
homojen bir şekilde dağıtılmamıştır. Örneğin, birinci cihazda ‘1’ verisi bulunmazken, 2. Cihaza ‘4 ’ verisi eklenmemiştir. Bu şekilde, cihazların ortak verisinin olmadığı bilgisi verilebilir. Bu bölümde farklı veriler ve verilerin farklı kullanıcılara dağılımı heterojenliği arttırıp, gözlemler yapılmaktadır. Genel yorum olarak heterojenlik düzeyinin artışının, ortak bir model oluşturmayı zorlaştırdığını söyleyebiliriz. Bu durum federe öğrenmenin zorluklarından istatistiksel heterojenlik kavramına örnek olabilir.

Cihaz doğruluk ve kayıp grafiklerinde, epoch sayısından bağımsız olarak (Şekil 3.60. ve Şekil 3.64.) farklı veri dağılımlarından ötürü dalgalanmaların çok sık olduğu gözlemlenebilir. Şekil 3.58’de farklı verilerin, sunucuda toplanması, sunucudaki test verilerinin doğruluğunun, bir süre dalgalandıktan sonra %95,37 doğruluğa ulaşmasını sağlamıştır. Sunucu farklı veri ağırlıklarını toplayıp cihazlara daha eğitilmiş bir model sunmaya çalışmıştır. Fakat cihazlardaki veri farklılığı sebebiyle cihazların bir değere yakınsaması 100. Yinelemede, 100 paket boyutu ve 1 epoch ile gerçekleşmemiştir. Aynı deney epoch sayısı 3 olarak belirlenerek tekrardan yapılmıştır. Bu durumda sunucuda büyük bir değişiklik yaşanmamasına rağmen, cihazların 100. yinelemede doğruluk oranının arttığı gözlemlenmiştir. Doğruluk oranının, yakınsamasını sağlamanın yollarından biri de epoch sayısını arttırarak yapmak olduğunu söylenilebilir. Fakat eğitimin tamamlanma süresinin de 2422’den 4496’ya uzayan bir artışa sebep olduğu da gözlemlenmektedir.

3.1.4.3. Veri Heterojenliği-2

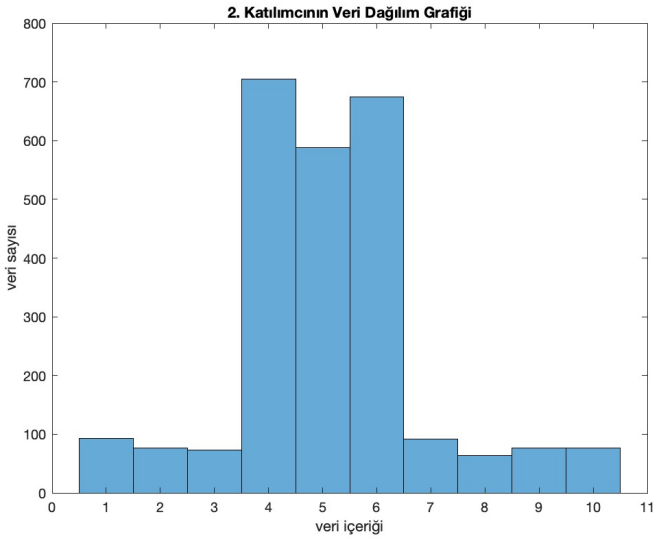
Aşağıda istatistiksel heterojenliğinin etkisi incelenmiştir. Öğrenmeye katılacak her aktif cihazda, diğer cihazlarla aynı tip veri olduğu, fakat bu verilerin farklı sayıda dağıtıldığı kabul edilmiştir. Bu bölümde yapılan deneylerde veri sayısının, cihaz içindeki dağılımı Şekil 3.68. , Şekil 3.69. Şekil 3.70. görsellerinden incelenebilir. Aşağıda yapılmış her deney için katılımcıların ayrı ayrı kayıp ve doğruluk grafiği, sistemin genel kayıp ve doğruluk grafiği, görselleriyle birlikte eklenerek açıklamalarda bulunulmuştur. Yorum bu bölümün sonunda yapılmıştır.

Şekil 3.68. 1.Katılımcının hangi verilerden kaç adet veri sayısı olduğunu içermektedir.



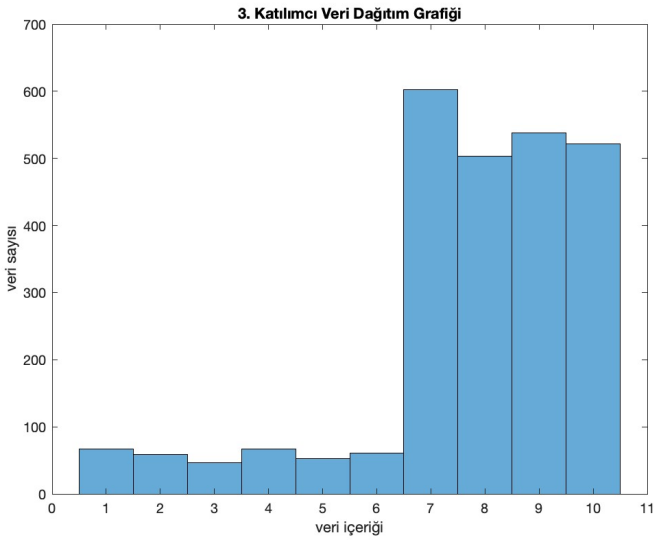
Şekil 3.68. 1. Katılımcının veri içeriği-veri sayısı grafiği

Şekil 3.69. 2. Katılımcının hangi verilerden kaç adet veri sayısı olduğunu içermektedir.



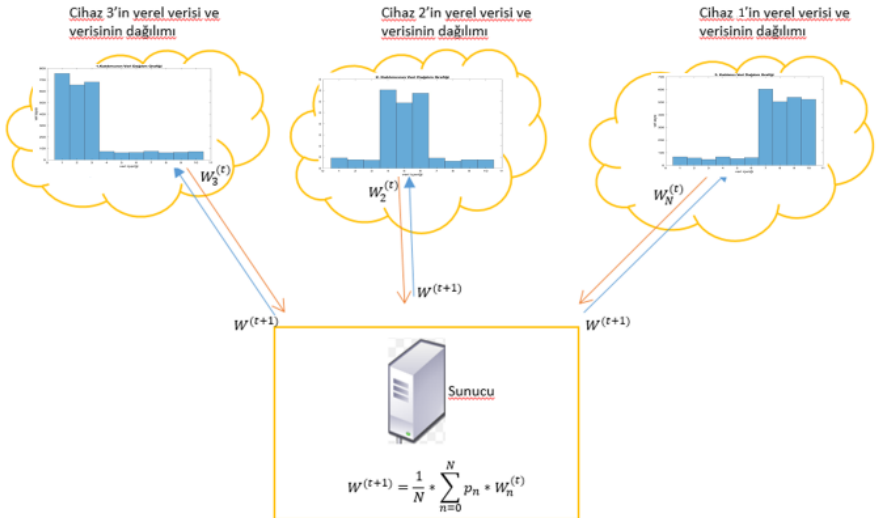
Şekil 3.69. 2. Katılımcının veri içeriği-veri sayısı grafiği

Şekil 3.70. 3. Katılımcının hangi verilerden kaç adet veri sayısı olduğunu içermektedir.



Şekil 3.70. 3. Katılımcının veri içeriği-veri sayısı grafiği

Şekil 3.71'de 3 cihaz için oluşturulan sistem gözülmektedir:



Şekil 3.71. 3. Katılımcının veri içeriği-veri sayısı grafiği

Çizelge 3.6 numaralı tabloda, 1. satırda heterojen veri paketleriyle yapılan koşum sonrası alınan kayıp/doğruluk grafiklerinin numerik değerleri ve 2. Satırda 3.1.2.4 bölümünde yapılan, homojen veri sayısı ve dağılımıyla oluşturulmuş koşumda alınan kayıp/doğruluk grafiklerinin numerik değerleri verilmiştir:

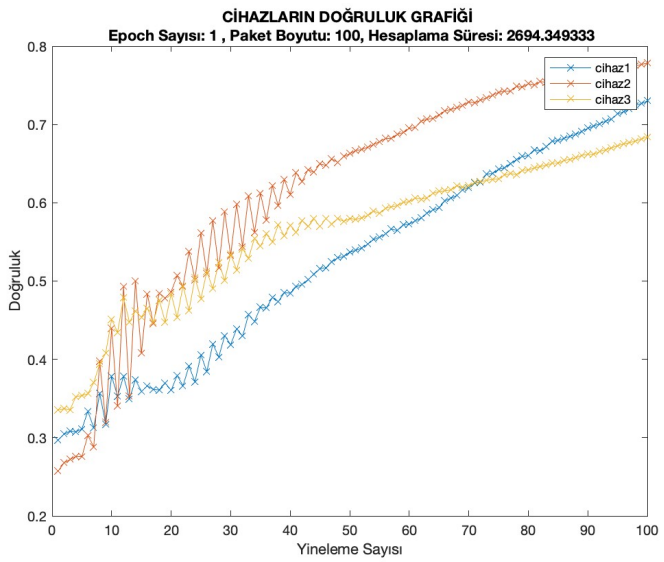
Momentum: M

Öğrenme Katsayısı: ÖK

Her Bir Cihazdaki Veri Sayısı	Epoch Sayısı	Paket Boyutu	Kayıp Değeri	Doğruluk Değeri	Hesaplama Süresi	Cihaz Sayısı	M/ÖK
2567-2518-3017 (hetero)	1	100	**%20.15**	**%94,80**	**2694**	3	M
2000-2000-2000 (homo)	1	100	**%16,45**	**%95,57**	**2265**	3	M

Çizelge 3.6. Cihazlara dağıtılan verinin heterojen olmasına yönelik çalışmaların kayıp/doğruluk değerleri

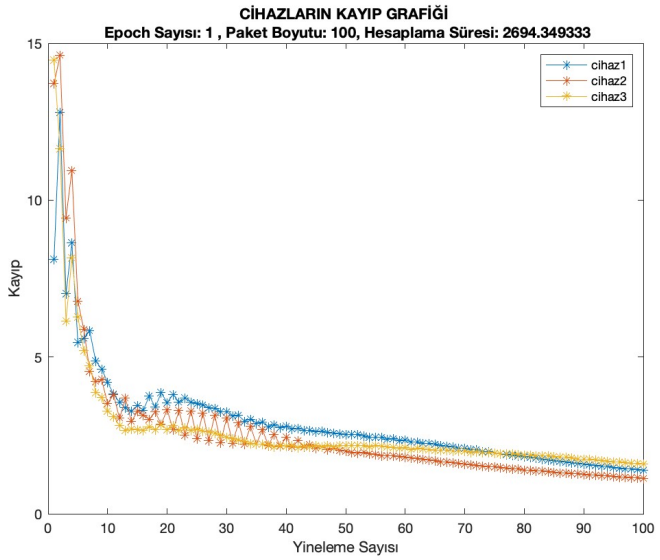
Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1'dir. Her katılımcıda veri heterojen olarak dağıtılmış olup, her cihazın aynı sınıflandırma verisi (Hepsinde 0,1,2,3,4,5,6,7,8,9 verileri mevcuttur.) vardır. Fakat, farklı sayılarda bulunmaktadır (Hepsinin 0,1,2,3,4,5,6,7,8,9 verilerinin dağımı farklıdır.). 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.72. Bütün katılımcıların doğruluk grafiği

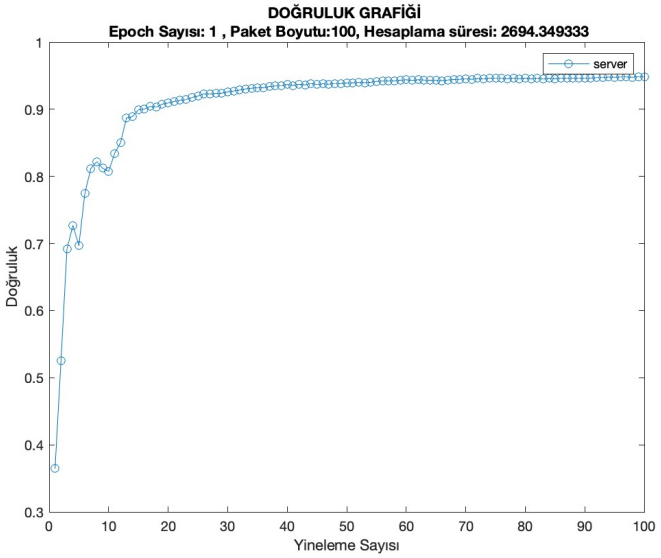
Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1'dir. Her katılımcıda veri heterojen olarak dağıtılmış olup, her cihazın aynı sınıflandırma verisi (Hepsinde 0,1,2,3,4,5,6,7,8,9 verileri mevcuttur.) vardır. Fakat farklı sayılarda bulunmaktadır (Hepsinin

0,1,2,3,4,5,6,7,8,9 verilerinin dağımı farklıdır.). 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



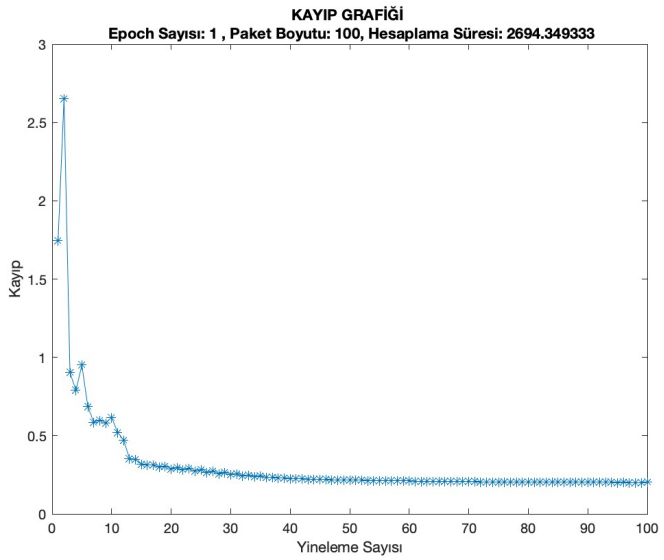
Şekil 3.73. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1'dir. Her katılımcıda veri heterojen olarak dağıtılmış olup, her cihazın aynı sınıflandırma verisi (Hepsinde 0,1,2,3,4,5,6,7,8,9 verileri mevcuttur.) vardır. Fakat farklı sayılarda bulunmaktadır (Hepsinin 0,1,2,3,4,5,6,7,8,9 verilerinin dağımı farklıdır.). 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.74. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 1'dir. Her katılımcıda veri heterojen olarak dağıtılmış olup, her cihazın aynı sınıflandırma verisi (Hepsinde 0,1,2,3,4,5,6,7,8,9 verileri mevcuttur.) vardır. Fakat farklı sayılarda bulunmaktadır (Hepsinin 0,1,2,3,4,5,6,7,8,9 verilerinin dağımı farklıdır.). 3 cihazın aktif olduğu bu deneyde, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.75. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

Bu bölümde federe öğrenmenin zorluklarından olan cihazlardaki veri düzensizliğin bir örneği incelenmiştir. Gerçek hayatta, her cihazda aynı veriden olsa bile verilerin aynı sayıda olmayacak olması yüksek bir ihtimaldir. Federe öğrenmede bu durumda doğruluk oranında düşme ve yakınsamada gecikme görülebilir. Epoch sayısı etkisini incelerken yapılan çalışmada bu bölümde yapılan çalışmadan farklı olarak cihazda verilerin homojen olarak dağıtılması sağlanmıştır. Bu sebeple karşılaştırmaya 3.1.2.4. bölümünde yapılan deneyle başlanılabilir. Verilerin düzenli dağılması sonucu %95 değerinde bir doğruluk oranında yakınsama görülürken, kayıp yüzde 16'lara kadar gerilemiştir. Heterojen dağılımındaki grafiklere baktığımız zaman sistem, Şekil 3.74. ve Şekil 3.75'te verilen kayıp ve doğruluk görsellerinde uygun değere yakınsamıştır. Cihazlardaki veri heterojenliğinden ötürü Şekil 3.72., Şekil 3.73'te görüldüğü üzere, cihazlarda uygun değere yakınsama daha tam görüntülenememiştir. Yakınsamış olan sistemin doğruluk değeri %94 oranı ile homojen veriden elde edilen doğruluk değerinin altında kalmıştır. Kayıp ise %20 değerindedir.

Sonuç olarak federe öğrenmede, bu deney heterojen durumda öğrenmenin geciktiği, buna ek olarak öğrenme süresinin de uzadığı yorumu yapılabilir.

3.1.5.FEDAVGM ve FEDAVG

Bu bölüme kadar yapılan çalışmalarda geçmiş hesaplamalar yararlanarak dikey gürültüyü yok edip, yataydaki ilerlemeyi hızlandıran momentum yöntemi kullanılmıştır. Bu bölümde ise momentum ve öğrenme katsayısı arasındaki performansı inceleyerek, performansı yüksek olan parametreyi bulmayı hedeflemektedir.

Bu bölümdeki çalışmalarda ilk önce öğrenme katsayı seçiminin performans üzerindeki etkisini inceleyerek başlanmış, daha sonra ise uygun olan öğrenme katsayısı ile momentum yönteminin incelemesi yapılmıştır.

3.1.5.1 Öğrenme Katsayısı Etkisi

Bu bölümde, bölüm 2.2.1.8'de anlatılan öğrenme katsayısının öğrenme üzerine etkisi incelenmiştir. Bu bölümdeki kazanımlar, bölüm 3.1.5.2'ye girdi olarak sağlanabilir. Bu bölümden elde edilen uygun öğrenme katsayısı ile momentum kullanılan sistemlerin performansları karşılaştırılırken uygun olan öğrenme katsayısı seçilebilir.

Aşağıda farklı öğrenme katsayıları ile yapılan deneylerin grafikleri yer almaktadır. Deney yapılan öğrenme katsayıları 0.1,0.01,0.001 olarak belirlenmiştir. Deneyler için yapılan yorumlar bu bölümün sonundadır.

Çizelge 3.7 numaralı tabloda farklı öğrenme katsayısı ile koşum sonrası alınan kayıp/doğruluk grafiklerinin numerik değerleri verilmiştir:

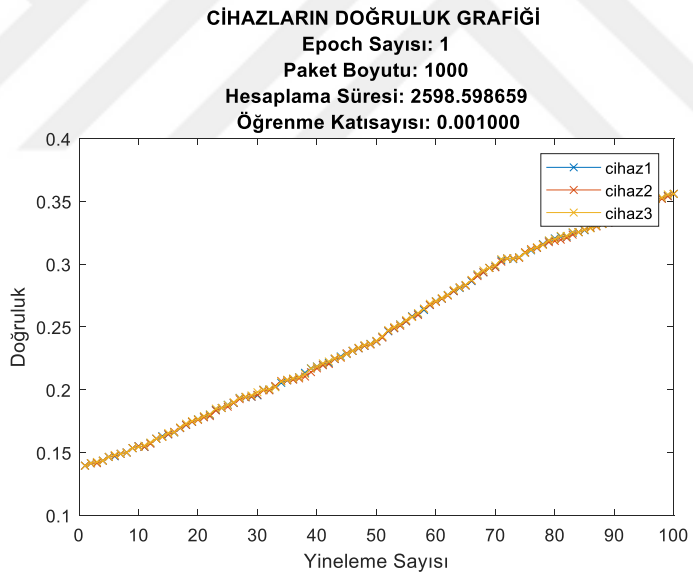
Momentum:M
Öğrenme Katsayısı: ÖK

Cihaz Sayısı	Kayıp Değeri	Doğruluk Değeri (%)	Hesaplama Süresi	Paket Boyutu	Her Bir Cihazdaki Veri Sayısı	Epoch Sayısı	M/ÖK
3	**2.2785**	**%38,65**	**2598**	1000	2000	1	**ÖK=0.001**
3	**0.8916**	**%78,80**	**2682**	1000	2000	1	**ÖK=0.01**
3	**0.3153**	**%91,10**	**2674**	1000	2000	1	**ÖK=0.1**

Çizelge 3.7. Öğrenme katsayısı çalışmalarına yönelik kayıp/doğruluk değerleri

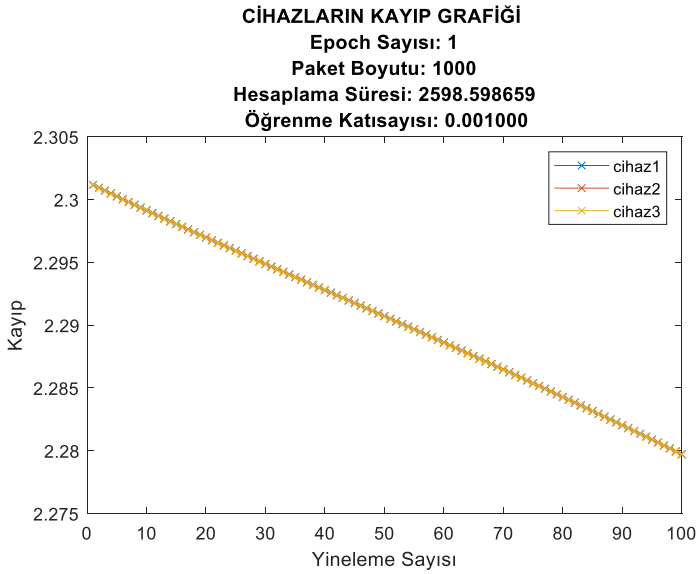
Öğrenme Katsayısı=0.001

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı (0.001) kullanılarak uygulanmıştır.



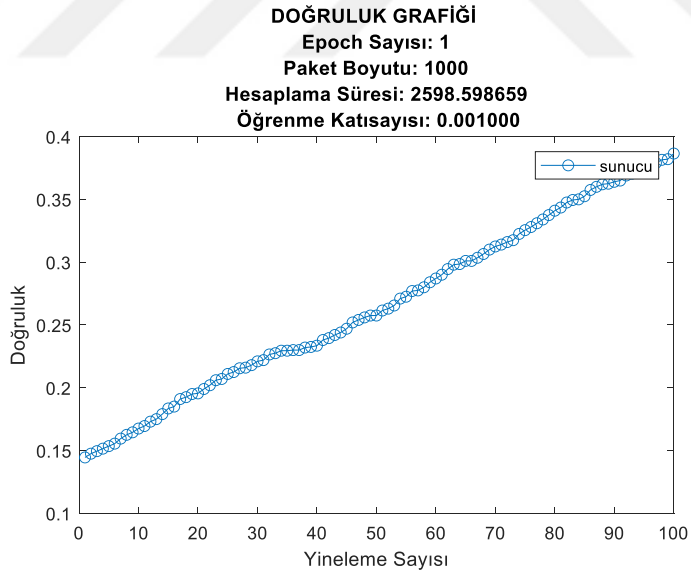
Şekil 3.76. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı (0.001) kullanılarak uygulanmıştır.



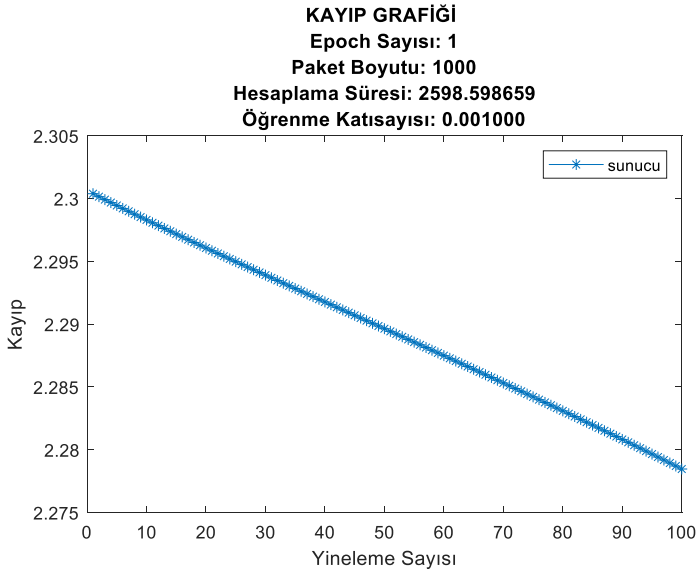
Şekil 3.77. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı (0.001) kullanılarak uygulanmıştır.



Şekil 3.78. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

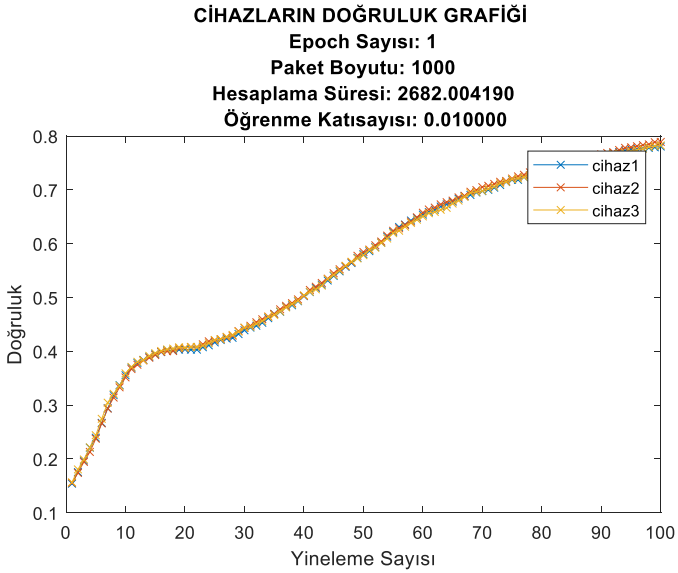
Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı (0.001) kullanılarak uygulanmıştır.



Şekil 3.79. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

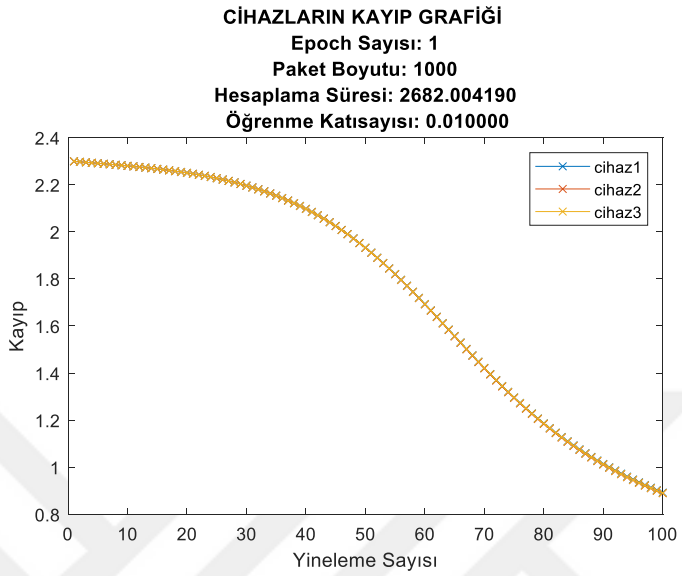
Öğrenme Katsayısı=0.01

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı (0.01) kullanılarak uygulanmıştır.



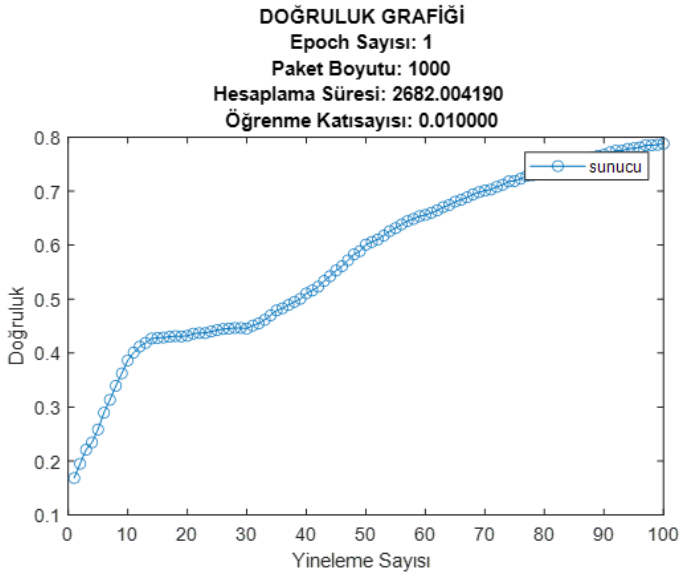
Şekil 3.80. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı (0.01) kullanılarak uygulanmıştır.



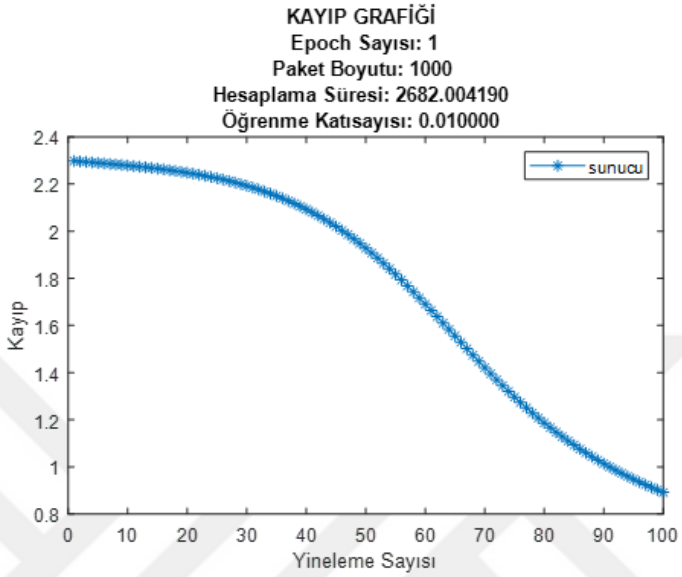
Şekil 3.81. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı (0.01) kullanılarak uygulanmıştır.



Şekil 3.82. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

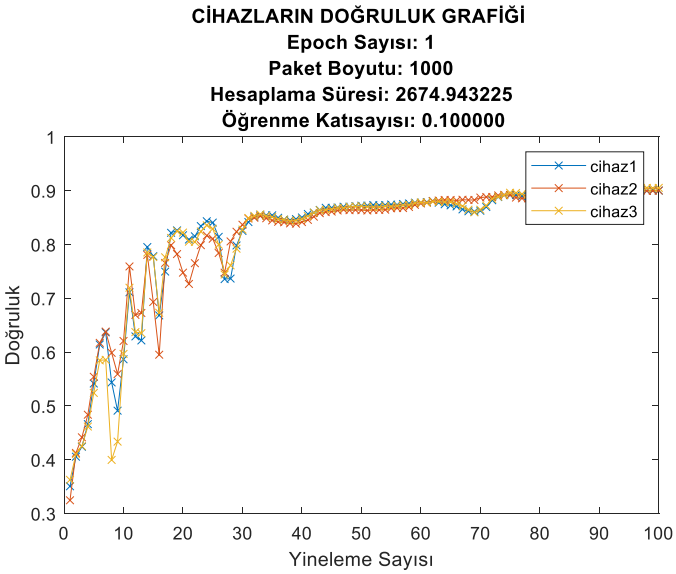
Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı (0.01) kullanılarak uygulanmıştır.



Şekil 3.83. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

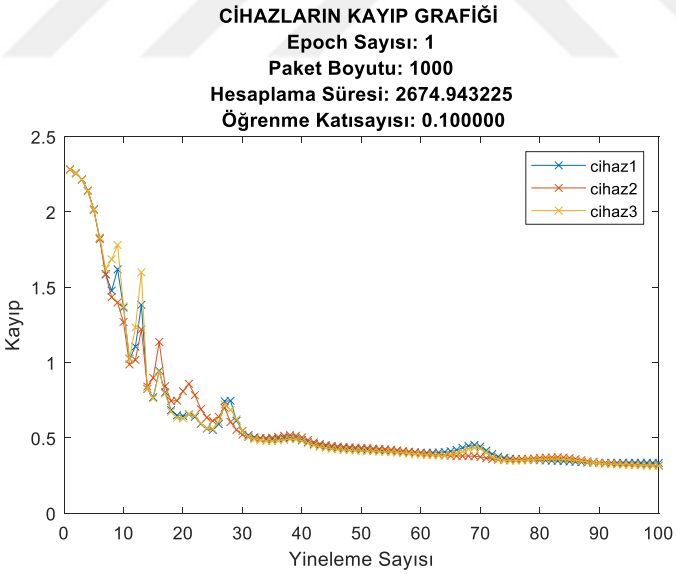
Öğrenme Katsayısı=0.1

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı (0.1) kullanılarak uygulanmıştır.



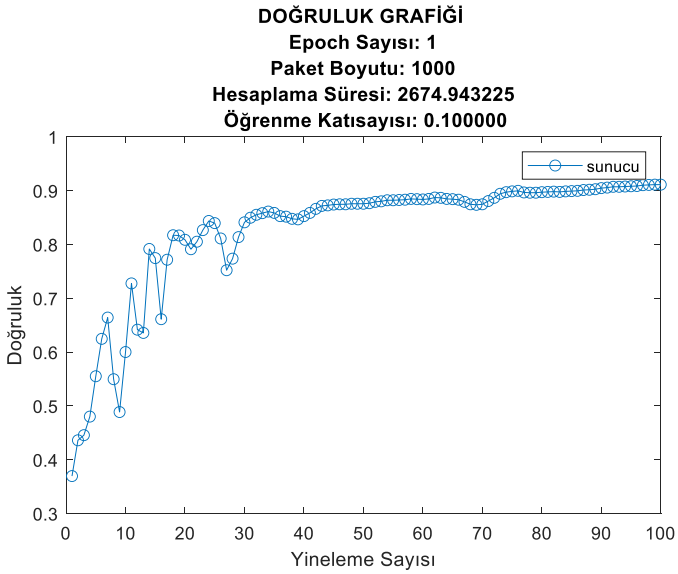
Şekil 3.84. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı (0.1) kullanılarak uygulanmıştır.



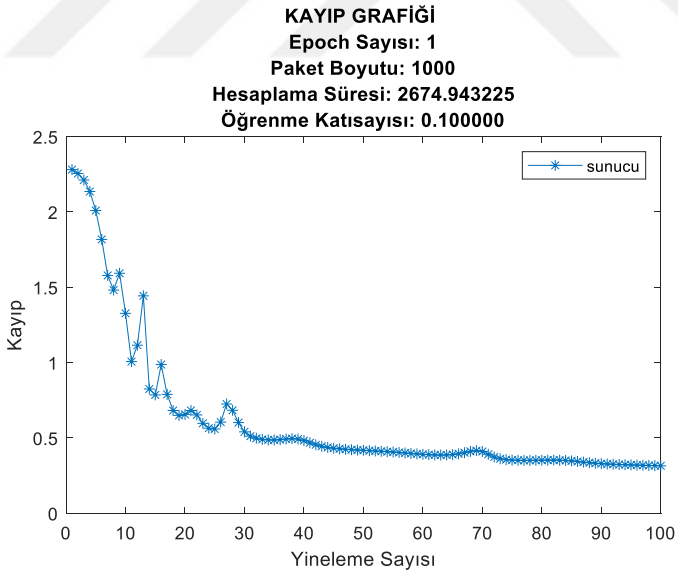
Şekil 3.85. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı (0.1) kullanılarak uygulanmıştır.



Şekil 3.86. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

Aşağıdaki grafik için paket boyutu 1000, epoch sayısı 1 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı (0.1) kullanılarak uygulanmıştır.



Şekil 3.87. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

Deney sonuçları incelendiğinde, öğrenme katsayısı en büyük olarak seçilen 0.1 öğrenme katsayısı ile, doğruluk değeri en yüksek olan ve kayıp değeri en düşük olan değer elde edilmiştir. Uygun değere en çok yakınsayan büyük boyutlu olan öğrenme kat sayısı olarak

Şekil 3.86. ve Şekil 3.87 grafikleriyle Şekil 3.78. ve Şekil 3.79'daki grafikler karşılaştırılarak anlaşılabilir.

Öğrenme katsayısının büyük seçilmesi, model ağırlıkları daha büyük bir değişimle güncelleneceği için model ağırlıklarının daha büyük bir ivme ile değiştiği, en uygun değere daha hızlı bir iniş sağlanabileceği söylenebilir.

Öğrenme sayısının büyük seçilmesinin sistemden bir ödünleşmesi vardır. O da öğrenme katsayısı büyüdükçe adımların büyüdüğü ve en uygun değere ulaşıp hatta aşabileceği durumunun olmasıdır. Bu da salınıma, kararsızlığa ve kayıp değerinin artmasına sebep olur.

3.1.5.2. FEDAVG

Bu bölümde, bu zamana kadar kullanılan FEDAVGM yönteminden farklı olarak FEDAVG yönteminin uygulanmasını içerir. Ayrıntılı bilgi için 2.2.1.8 bölümdeki kontrollü iniş ve öğrenme katsayısı isimli bölümden ulaşılabilir.

Bölüm 3.1.5.1'de farklı öğrenme katsayıları denenmiştir. Deneme yapılan katsayılarından, salınımı arttırmayan, aynı zamanda süreyi de çok uzatmadığı gözlemlenen 0.01, öğrenme kat sayısı olarak seçilmiştir. Yorumlar bu bölümün sonunda verilmiştir.

Çizelge 3.8 numaralı tabloda öğrenme katsayısı kullanılarak uygulanan FEDAVG yönteminin koşum sonrası alınan kayıp/doğruluk grafiklerinin numerik değerleri verilmiştir:

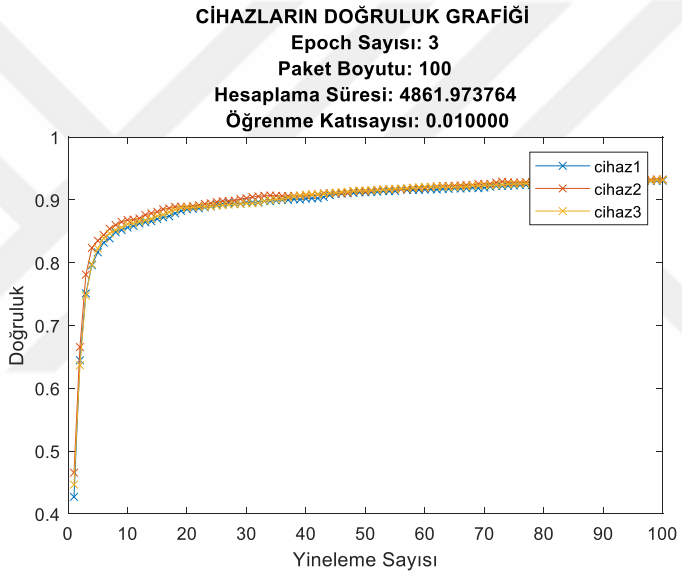
Momentum: M
Öğrenme Katsayısı: ÖK

Cihaz Sayısı	Kayıp Değeri	Doğruluk Değeri (%)	Hesaplama Süresi	Paket Boyutu	Her Bir Cihazdaki	Epoch Sayısı	M/ÖK

					Veri Sayısı		
3	**%22,07**	**%93,70**	**4861**	100	2000	3	**ÖK=0.01**

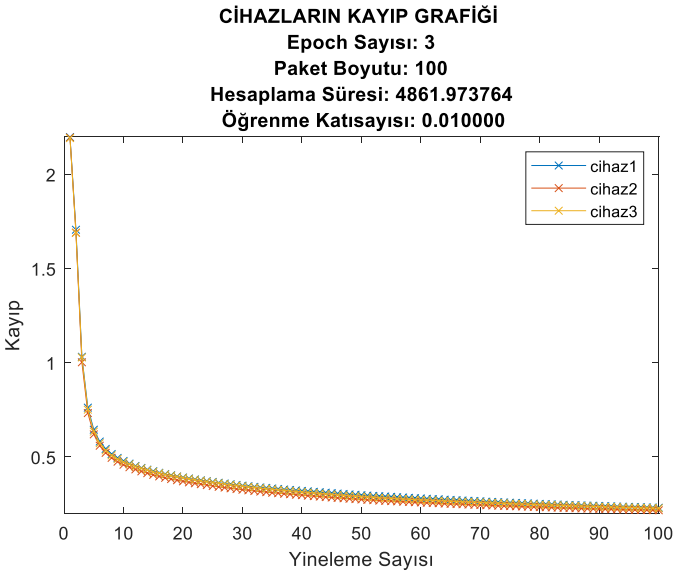
Çizelge 3.8. FEDAVG- Öğrenme katsayısı sisteminin performans çalışmalarına yönelik kayıp/doğruluk değerleri

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 3 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı kullanılarak uygulanmıştır.



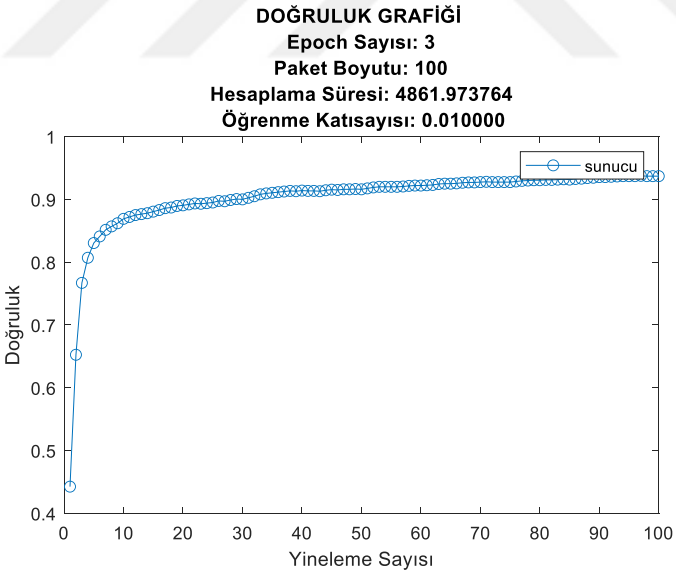
Şekil 3.88. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 3 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı kullanılarak uygulanmıştır.



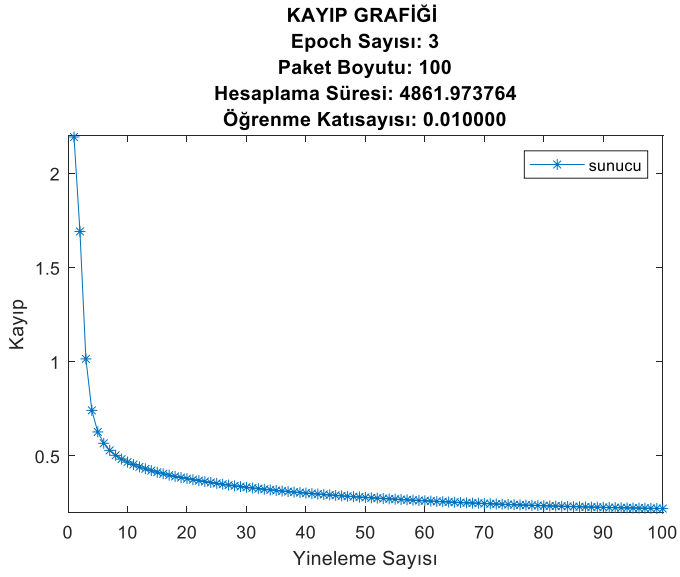
Şekil 3.89. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 3 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı kullanılarak uygulanmıştır.



Şekil 3.90. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 3 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVG yöntemi yerel cihazlarda öğrenme katsayısı kullanılarak uygulanmıştır.



Şekil 3.91. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

Aşağıda cihaz ve sistem grafikleri incelendiğinde, salınımın ve gürültünün çok olmadığını grafiklerin uygun değere yakınsadığını gözlemleyebiliriz. Doğruluk değerinin %93,70 değerine, kayıp değeri %22,07 değerine yakınsadığı, çizelge 3.8. ve grafiklerden gözlemlenebilir.

Bu deneydeki amaç parametrelerin uygun değere nasıl yakınsayacağı olmayıp, asıl amaç FEDAVGM ve FEDAVG etkisini gözlemlemektir. Performansı arttırmak için yapılan çalışmalar bölümlerden 3.1.1, 3.1.2, 3.1.3, 3.1.4'te mevcuttur.

3.1.5.3. FEDAVGM

Bu deneyde, bölüm 3.1.5.2.'de FEDAVG yönteminin uygulanması ve deneyin sonuçları karşılaştırarak incelenmiştir.

3.1.5.1 bölümünde federe öğrenme sisteminde farklı öğrenme katsayıları denenmiştir. Deneme yapılan katsayılarından, salınımı arttırmayan, aynı zamanda süreyi de çok uzatmayacağı düşünülen 0.01, öğrenme kat sayısı olarak 3.1.5.2. bölümünde seçilmiştir. Bu bölümde ise FEDAVGM yöntemi uygulanmış olarak deney yapılacaktır. Deneyler için yorumlar bu bölümün sonundadır.

Çizelge 3.9 numaralı tabloda kontrollü iniş kullanılarak uygulanan FEDAVG yönteminin koşum sonrası alınan kayıp/doğruluk grafiklerinin numerik değerleri verilmiştir:

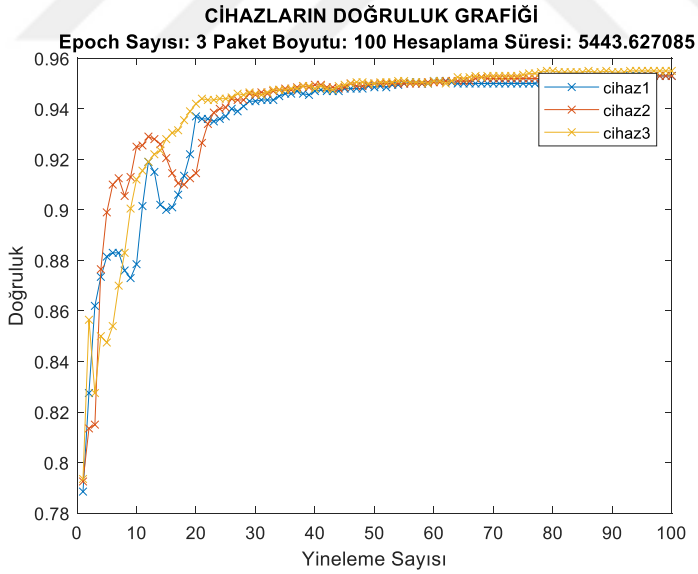
Momentum: M

Öğrenme Katsayısı: ÖK

Cihaz Sayısı	Kayıp Değeri	Doğruluk Değeri (%)	Hesaplama Süresi	Paket Boyutu	Her Bir Cihazdaki Veri Sayısı	Epoch Sayısı	M/ÖK
3	**%22,27**	**%95,00**	**5443**	100	2000	3	**M**

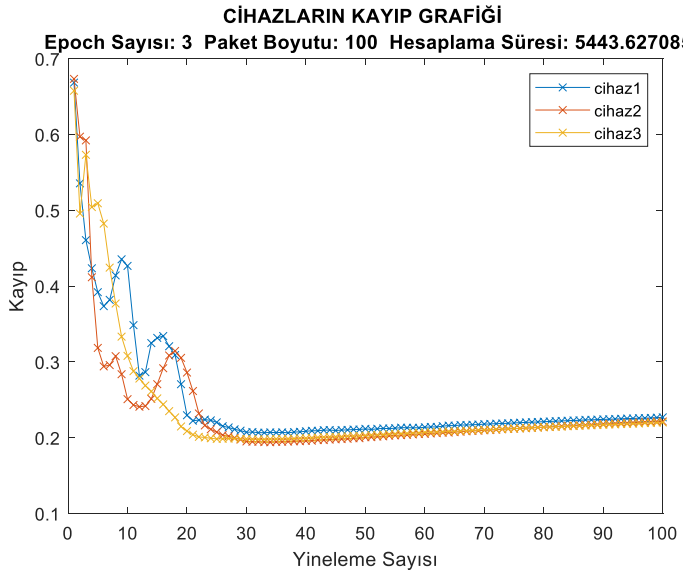
Çizelge 3.9. FEDAVG- Momentum sisteminin performans çalışmalarına yönelik kayıp/doğruluk değerleri

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 3 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



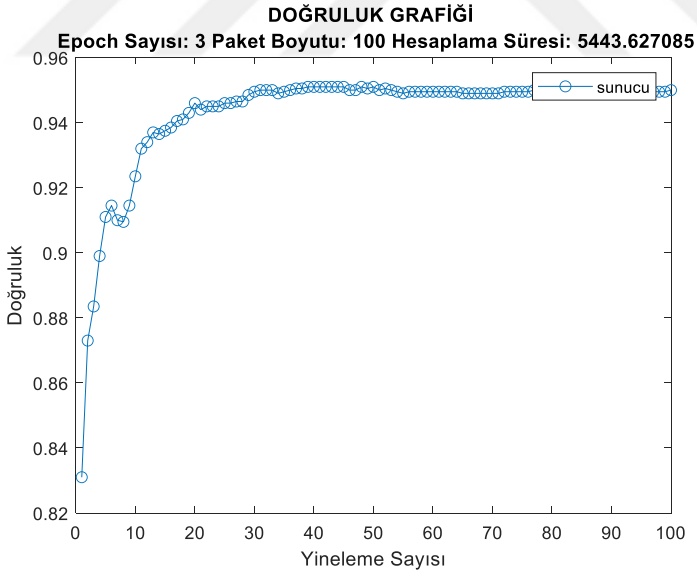
Şekil 3.92. Bütün katılımcıların doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 3 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



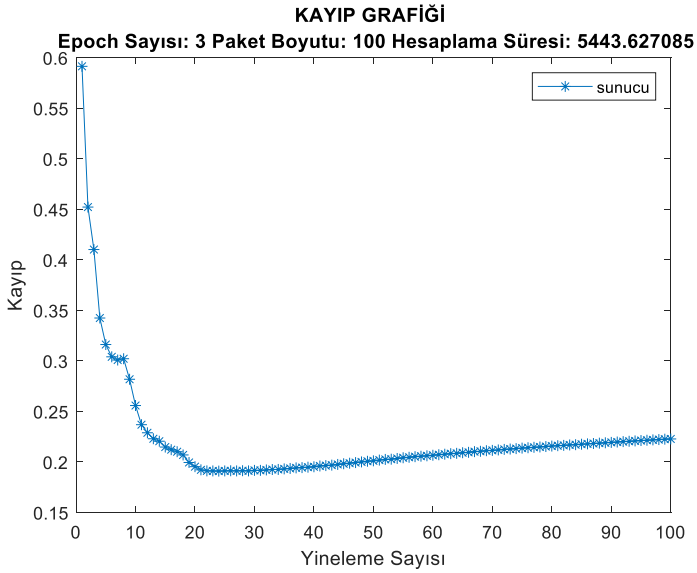
Şekil 3.93. Bütün katılımcıların kayıp grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 3 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.94. Sunucunun kendi test verisiyle yaptığı sistemin genel doğruluk grafiği

Aşağıdaki grafik için paket boyutu 100, epoch sayısı 3 ve her bir katılımcıda homojen olarak dağıtılmış veri var iken, 3 cihazın aktif olduğu durumda, FEDAVGM kullanılarak uygulanmıştır.



Şekil 3.95. Sunucunun kendi test verisiyle yaptığı sistemin genel kayıp grafiği

Aşağıda cihaz ve sistem grafikleri incelendiğinde, salınımın ve gürültünün mevcut olduğunu, grafiklerin uygun değere yakınsadığını gözlemleyebiliriz. Doğruluk değerinin %95,00 değerine, kayıp değeri %22,27 değerine, çizelge 3.9. ve grafiklerden gözlemlenebilir.

3.1.5.2. ve 3.1.5.3 deneylerinin hesaplama süresi karşılaştırmasında, FEDAVGM uygulandığında sürenin bir miktar arttığı görülmektedir. Süre artışı duruma göre bağlantı kopmasına, iletişim için gereksiz maliyet yaratmasına sebep olabilir bu sebeple seçimin nasıl yapılması gerektiği aşağıdaki paragrafta ayrıntılı olarak incelenmiştir.

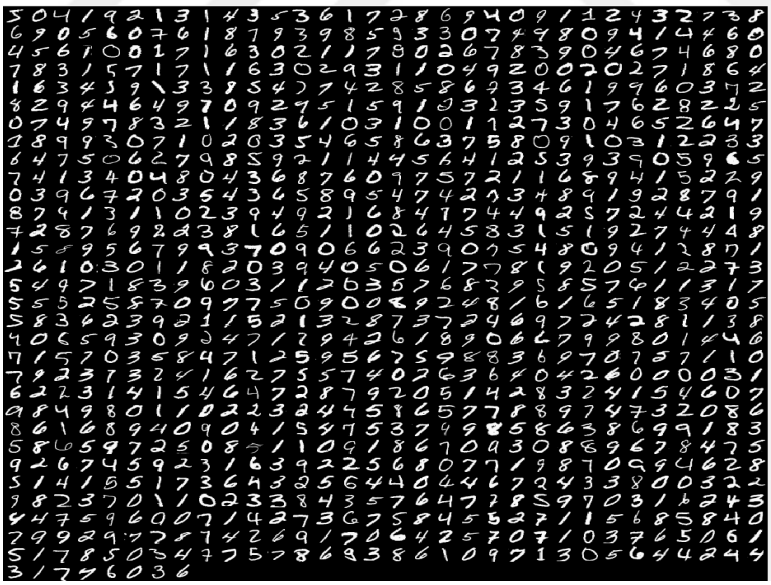
Deneylerde, doğruluk ve kayıp değerlerine bakarak FEDAVGM yönteminin, FEDAVG yöntemine göre daha üstün olduğu düşünülebilir. Fakat bu durum her zaman doğru olmayabilir. Duruma göre hangi algoritmanın seçileceği çok önemlidir. Veri sayısı azsa, salınımın az olması isteniyorsa, yeterli etiket verisi yoksa, iletişim maliyeti önemli bir husussa FEDAVGM yöntemi daha uygun olabilir. Fakat, az sayıda yineleme gerçekleştirilen deneylerde, ağ üzerinde yüksek miktarda bilgi taşınılmayacağı durumlarda, veri sayısı az olan durumlarda FEDAVGM kullanmak verimli olmayabilir. Bu sebeple seçilen algoritmalardan, ne istenildiği bilinmeli hangi kriterlere uyulması gerektiğine dikkat edilmelidir.

3.2.Deney Mimarisi

Bu bölümde deneysel çalışmaların yapılması için MATLAB aracı kullanarak oluşturulan mimari anlatılmıştır:

3.2.1. Evrişimsel Sinir Ağları Mimarisi

ESA için kullandığımız veri setinin piksel boyutu 28*28'dir. Kullandığımız veri setinde her bir etiket eşit sayıda ve eşit oranda dağılmış olup, 70000 adet el yazısı ile yazılmış rakamlar oluşturulmuştur. 10000 adet veri olan set ayrı tutulup, bu verinin bir kısmı test aşamasında kullanılmıştır. Şekil 3.96 görselinde, MNIST veri setinden 1000 adet görselin olduğu bir parçayı içermektedir. Rakamlar 0'dan 9'a kadar 10 farklı sınıfa ayrılır. Her bir görselin boyutu 28*28 piksel olduğu için, 784 adet giriş noktası vardır. 20 adet 9*9 boyutunda Evrişimsel filtre kullanılmıştır [14].



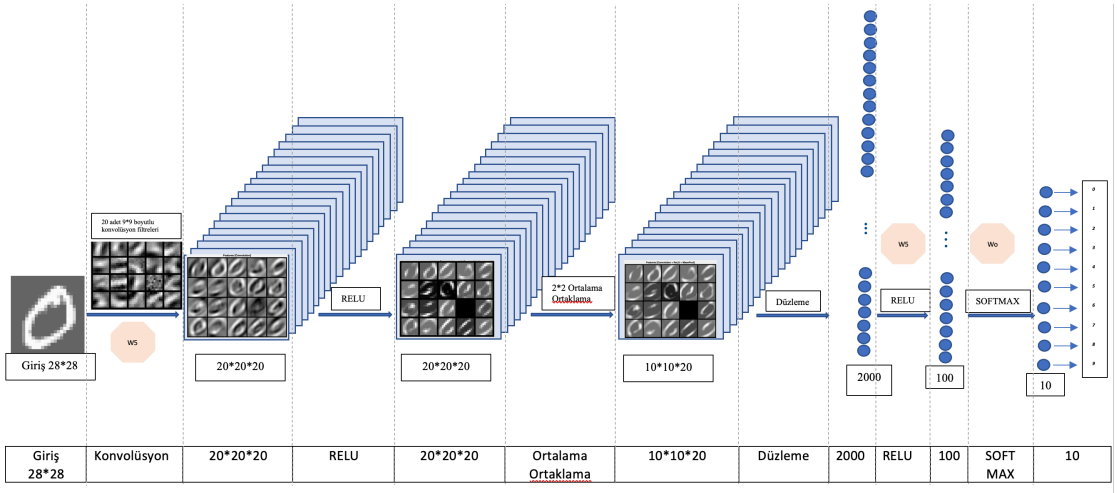
Şekil 3.96. MNIST veri seti örneğinden bir parça

Girdiler evrişimsel filtreye ile hesaplamaya girdikten sonra doğrusal düzeltme birimi (Relu) aktivasyon fonksiyonu uygulanır. Relu, aktivasyon fonksiyonu olarak çalışır. Sıfırdan küçük girdileri sıfıra eşitlerken, sıfırdan büyük girdiler için de doğrusal davranır.

Sıfırdan küçük değerleri sıfıra eşitlemesi hesaplamanın hızlı ve kolay olmasına katkıda bulunur. Relu fonksiyonundan sonra ortaklama katmanına girip boyutları 20x20x20 olan çıktıyı 10x10x20 boyutlarına dönüştürür. Bu aşamadan sonra matris, 2000x1 olarak, tek sütun halinde gizli bir katmana girer. Bu katman ile sistemin çıktısı 100x1 boyutlarına indirgenir. Bu aşamada 100 adet çıktı noktası vardır. Aktivasyon fonksiyonu olarak Relu fonksiyonu uygulanır. 10 adet çıktımız olması gerektiği için çıktıyı 10 adet çıktı olarak sınıflandırılması gerekir. Wo ağırlık matrisiyle 10x1 boyutlarına dönüştürülür Çıkışta kullandığımız aktivasyon fonksiyonu softmax olup, 10 noktanın toplam çıkış değerleri 1'e eşittir. Çıktı değerlerini sınıflar arasında olasılık dağılımı olarak verir.

ESA mimarimiz şekil 3.97'de verilmiştir. Eğitim isteyen katmanlar W1, W5, Wo olup şekil 3.97'de turkuaz renk ile gösterilmiştir. W1 konvolüsyon filtreleri ile kullanılırken, W5 ve Wo sinir ağında sınıflandırma için bağlantı ağırlıklarını içerir. W5 ve Wo, önceki katmanın çıktıları ile çarpılır. Daha sonra çıktıları aktivasyon fonksiyonuna girdi sağlanır.

Sistemin elde edilen çıktısı sonucunda, gerçek değer ve çıktı değer arasında bir fark oluşur. Bu fark hata adını alır. Hatayı azaltmak için ağırlık değerleri geri yayılım yoluyla güncellenerek minimize değer elde edilmeye çalışılmaktadır. Bu konuyla ilgili ayrıntılı bilgi 2.2.1.9 numaralı bölümde ayrıntılı olarak verilmiştir.



Şekil 3.97. ESA mimari yapısı

3.2.2. Birleşik Ortalama Öğrenme Mimarisi

FEDAVG yöntemi, yerel cihazlarda ESA kullanılarak, MATLAB aracı ile oluşturulmuştur. ESA'nın mimari yapısı, girdi ve çıktı boyutları dahil, 3.2.1 bölümünün altında verilmiştir. Bu başlık altında ise federe öğrenmenin mimarisi yer almaktadır.

Federe öğreniminin en belirgin özelliği, verinin cihazların dışına çıkmadan, model eğitiminin cihaz içinde gerçekleştirilmesidir. Bu sebeple cihazlar için ayrı fonksiyonlar oluşturulmuş, MNIST verisinin bir kısmı cihazlarda test ve öğrenme verisi olarak belirtilmiştir. Her cihaz için ayrı bir tanımlama yapılmış, cihazların verilerinin farklı ve ayrı olması sağlanmıştır.

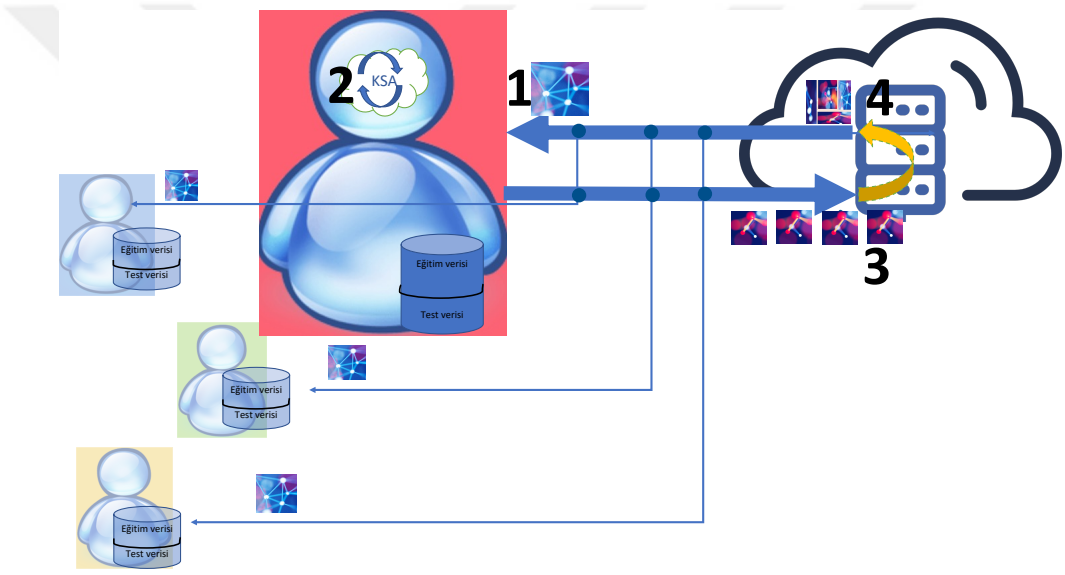
Federe öğrenmedeki bir diğer kavram ise sunucudur. Sunucunun görevi, eğitilmiş modelleri, bütün cihazlardan toplayarak, var olan modeli günceller. Güncellenmiş olan model, en uygun değere yakınsayana kadar cihazlar için dağıtım yapar. Sunucu fonksiyonu gelen W1,W5,Wo ağırlıklarını her yinelemede günceller. Her cihazdan gelen W1,W5,Wo ağırlıklarının ağırlıklı ortalamasını alır. Elde edilen güncellenmiş, W1,W5,Wo ağırlıkları sunucuda bulunan test veri seti ile test edip sistemin genel kayıp ve doğruluk grafiğini elde etmesinde görev alır.

Cihazlara dağıttığımız veri boyutu başta cihazların kendi fonskiyonunda belirlenir. Cihazların kendi eğitim verisi ile, cihaz içinde ESA kullanılarak daha önceden belirlenmiş olan parametrelerle eğitim gerçekleştirilir. Eğitim sonunda oluşan W1,W5,Wo ağırlıkları ile belirlediğimiz test seti ile test edilerek doğruluk ve kayıp değerleri elde edilir. Her yinelemede kayıp doğruluk değerleri her bir cihaz ve sunucu için kayıt altına alınır.

Şekil 3.98'de görüldüğü üzere, numaralandırılma yapılmıştır. İşlemlerin hangi sıra ile gerçekleştiğini göstermektedir:

1. Sunucu, mevcut modeli aktif bütün cihazlara dağıtır.

2. Her cihaz eğitim verisi ile ESA fonksiyonunda sunucudan gelen modeli eğitir. Test verileriyle doğruluk ve kayıp değerleri matris olarak tutulur. Eğitim sonucunda W1,W5,Wo ağırlıkları elde edilir.
3. Model, aktif cihazlardan, sunucuya gönderilir.
4. Sunucu, gelen modellerin ağırlıklı ortalamasını alarak birleştirir. Mevcut model güncelleştirilmiş olur. Diğer cihazların da iş birliğinde bulunarak geliştirdiği model, tekrardan dağıtılarak, cihazların kendi modeliyle sabit kalmamasına, eğitimine devam etmesini sağlar. Bu işlem kayıp ve doğruluk değerleri yakınsayana kadar devam eder.



Şekil 3.98. Federe Öğrenme Akış Sırası

4. SONUÇLAR VE TARTIŞMA

Bu bölümde, bölüm 3'te, yapılmış olan deneysel çalışmaların genel yorumu yapılacaktır. Deneysel çalışmalarda federe makine öğrenmesinin, FEDAVGM yöntemi ile öğrenme performanslarını değerlendirmek, nelerin etkilediği üzerinde yorum yapıp, etkileyen faktörleri duruma göre en uygun değere ayarlamayı amaçlamaktadır. Her parametrenin en iyi olarak seçilmesi mükemmel bir sistemi oluşturmaz. Bunun yerine mevcut durumlar ve gereksinimler incelenmeli ona göre parametreler en uygun şekilde seçilmelidir. Yani doğruluğu en yüksek olan çalışmayı en iyisi olarak değerlendirmek doğru olmayabilir.

Aşağıda verilmiş olan 4.1. numaralı çizelgede, bütün çalışmaların içerikleri ve çıktıları yer almaktadır. Bütün deneylerdeki girdi/çıktı bilgilerinin birleştirilmiş hali olarak da düşünülebilir.

Deney Numarası	Kayıp Değeri	Doğruluk Değeri	Hesaplama Süresi	Her Bir Cihazdaki Veri Sayısı	Epoch Sayısı	Cihaz Sayısı	M/ÖK	Paket Boyutu
Deney-1	30.77	93.2	948	2000	1	1	M	100
Deney-2	56.28	85.3	2202	2000	1	3	M	1000
Deney-3	16.45	95.57	2265	2000	1	3	M	100
Deney-4	16.45	95.57	2265	2000	1	3	M	100
Deney-5	16.45	95.57	2265	2000	1	3	M	100

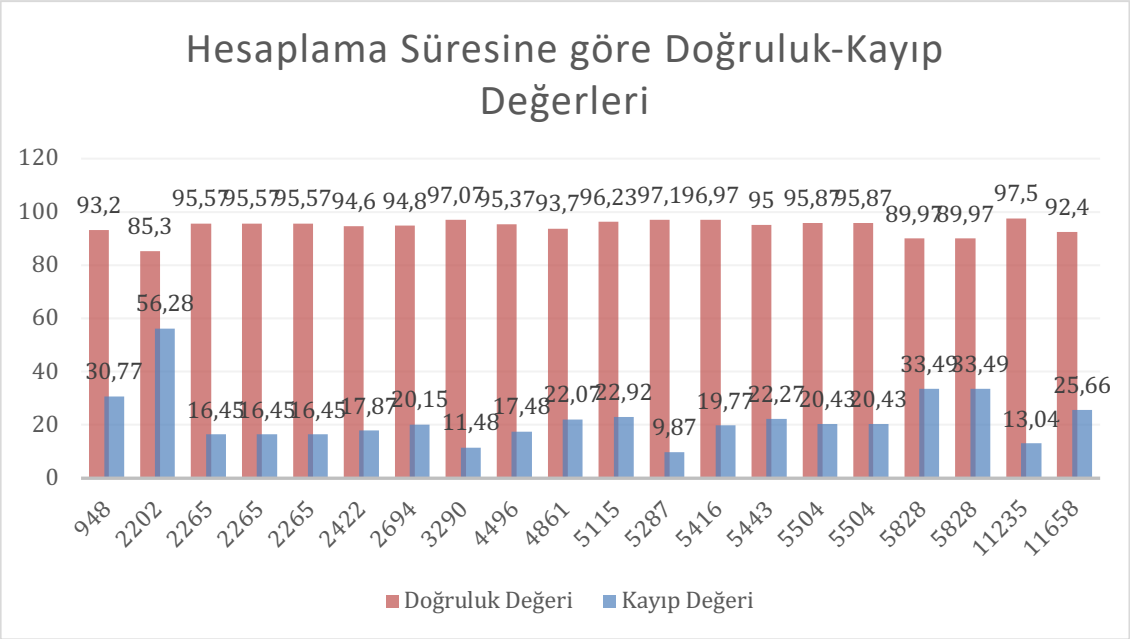
Deney-6	17.87	94.6	2422	1860-1745-2395	1	3	M	100
Deney-7	20.15	94.8	2694	2567-2518-3017 (hetero)	1	3	M	100
Deney-8	11.48	97.07	3290	2000-4000-6000	1	3	M	100
Deney-9	17.48	95.37	4496	1860-1745-2395	3	3	M	100
Deney-10	22.07	93.7	4861	2000	3	3	ÖK=0.01	100
Deney-11	22.92	96.23	5115	2000	4	3	M	40
Deney-12	9.87	97.1	5287	2000	1	10	M	100
Deney-13	19.77	96.97	5416	2000	4	3	M	4
Deney-14	22.27	95	5443	2000	3	3	M	100
Deney-15	20.43	95.87	5504	2000	4	3	M	100
Deney-16	20.43	95.87	5504	2000	4	3	M	100
Deney-17	33.49	89.97	5828	2000	4	3	M	1000
Deney-18	33.49	89.97	5828	2000	4	3	M	1000

Deney-19	13.04	97.5	11235	2000-4000-6000	4	3	M	100
Deney-20	25.66	92.4	11658	4000	4	3	M	1000

Çizelge 4.1. Deneysel çalışma sonuçları

Aşağıda verilmiş olan 4.2. numaralı çizelgede, bütün çalışmaların içeriklerinin ve çıktılarının yer aldığı çizelgeden elde edilen verilerle oluşturulmuş grafiktir. Bütün deneylerdeki çıktı bilgilerinin görselleştirilmiş hali olarak da düşünülebilir. Grafikte yatay eksende her deneyin hesaplama süresi vardır. Dikey eksende ise doğruluk ve kayıp değerlerinin sayısal değerleri vardır. Grafikte kırmızı sütun doğruluğu, mavi sütun ise kayıp değerlerini simgelemektedir.

Çizelge 4.2'de doğruluğun en yüksek olması demek hesaplama süresinin en yüksek olduğu senaryolardan biri olarak gözükmektedir. Bu senaryoda federe öğrenme gerçekleşirken uzun hesaplama sürelerinin yaşanması, bağlantı kopukluklarına ve cihazların tam katılım sağlayamamasına sebep olabilir. Hesaplama süresinin yüksek olmasının aksine, doğruluk oranı yüksek ve kayıp oranı da düşük olarak sayılabilir. Yüksek doğruluk oranının çok gerekli olduğu bir durumda, güçlü bir iletişim bağlantısı ve katılımcıların donanım olarak güçlü katılım sağlamasıyla başarılı bir sistem elde edilecek olabilir.



Çizelge 4.2. Hesaplama süresine göre kayıp/doğruluk değerleri

Çizelge 4.2'de hesaplama süresinin 2202 olduğu durum, elimizdeki güncel koşullarla en tercih edilmemesi gereken, öğrenimin en yavaş olduğu durum olarak değerlendirilebilir. Öğrenimi tamamlamamış daha fazla yinelemeye ihtiyaç duyulan durum olarak düşünülebilir. Yineleme süresi arttırıldığında, öğrenimini tamamlanabileceği yorumu yapılabilir. Çünkü deneyde paket boyutu çok büyük olduğu ve epoch küçük olduğu için öğrenme çözünürlüğü daha düşük kalmıştır.

Hesaplama süresinin 5287 olduğu durum ise doğruluk ve kayıp değerleri arasındaki farkın en büyük olduğu durum olabilir. Yani, hesaplama değeri ortalama civarda olmasına rağmen kayıp, doğruluk oranına göre düşük kalmıştır. Doğruluk oranı da ortalamanın üstü olarak düşünülebilir. Burada hesaplama süresi 11658 değerine çok fazla düşük olduğu için tercih sebebi olabilir fakat burada da çevre şartları ve durum önem kazanmaktadır. Çok fazla katılımın sağlanmasıyla oluşan bu doğruluk oranı, bağlantı sorunları sebepleriyle cihazların iletişimden düşmesine sebep olabilir. Öğrenmeler cihazlara bağlı olarak yarıda kalabilir. Bu sebeple hesaplama süresi ve sistem birlikte değerlendirilmelidir.

Günümüzde teknik olarak federe öğrenmeyi optimize eden yöntemlerin geliştirilmesi yönünde adımlar atılması çok önemlidir. Bu konu üzerinde ne kadar çok çalışma yapılıp adım atılırsa zorluklar daha ön plana çıkacak ve zorlukların çözümleri de bir o kadar hızlı

olacaktır. Her piyasaya açılabilecek olan federe öğrenmesi savunma sanayisindeki dijital ikiz projelerinden, tıp alanındaki sağlık analizlerine, kolay teşhise, Iot'den, eğlence sektörüne, endüstriye kadar birçok konuda kullanıcının veri gizliliğini dert etmeden kullanabileceği bir alan olması ile vizyonu geniş bir makine öğrenmesi olarak adlandırılabilir.

5. KAYNAKLAR

[1] Kairouz, P., Mcmahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'oliveira, R. G. L., Eichner, H., Rouayheb, S. El, Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Zhao, S. ,Advances And Open Problems İn Federated Learning, **(2019).**

[2] Abdulkadir Şeker, Banu Diri, Hasan Hüseyin Balık ', Derin Öğrenme Yöntemleri Ve Uygulamaları Hakkında Bir İnceleme , Gazi Mühendislik Bilimleri Dergisi, 47-64 **(2017).**

[3] Majeed, A., Zhang, X., & Hwang, S., , **(2022).**

[4] Shaheen, M., Farooq, M. S., Umer, T., & Kim, B. S. Applications Of Federated Learning; Taxonomy, Challenges, And Research Trends, Electronics (Switzerland), **(2022).**

[5] Pouriyeh, S., Shahid, O., Parizi, R. M., Sheng, Q. Z., Srivastava, G., Zhao, L., & Nasajpour, M.. Secure Smart Communication Efficiency İn Federated Learning: Achievements And Challenges. Applied Sciences (Switzerland), **(2022).**

[6] Huang, X., Ding, Y., Jiang, Z.L. Et Al. Dp-Fl: A Novel Differentially Private Federated Learning Framework For The Unbalanced Data. **(2020).**

[7] Long, G., Shen, T., Tan, Y., Gerrard, L., Clarke, A., Jiang, J. Federated Learning For Privacy-Preserving Open Innovation Future On Digital Health. In: Chen, F., Zhou, J. (Eds) Humanity Driven Aı. Springer, Cham, **(2021).**

[8] Konecný, J., Mcmahan, H. B., Ramage, D., & Richtárik, P., Federated Optimization: Distributed Machine Learning For On-Device Intelligence, **(2016).**

[9] Jing Xu, Sen Wang, Liwei Wang, Andrew Chi-Chih Yao, Fedcm: Federated Learning With Client-Level Momentum, **(2021).**

[10] Fei-Fei Li, Convolutional Neural Networks (Cnns / Convnets), Cs231n: Deep Learning For Computer Vision, Https://Cs231n.Github.İo/Convolutional-Networks/#Architectures Stanford - Spring **(2023).**

[11] Tuncer Ergin , Convolutional Neural Network (Convnet Yada Cnn) Nedir, Nasıl Çalışır? , Https://Medium.Com/@Tuncerergin/Convolutional-Neural-Network-Convnet-Yada-Cnn-Nedir-Nasil-Calisir-97a0f5d34cad, **(2018)..**

[12] Andrew Ng, Convolutional Neural Network, Https://Www.Coursera.Org/Learn/Convolutional-Neural-Networks#About, **(2017).**

[13] Yamashita, R., Nishio, M., Do, R.K.G. Et Al. Convolutional Neural Networks: An Overview And Application İn Radiology. Insights Imaging 9, 611–629 **(2018).**

[14] Nuruzzaman Faruqui, Convolutional Neural Network İn Matlab, Https://Www.Youtube.Com/Watch?V=Zoxowyuvcqw, **(2018).**

[15] Mk Gurucharan, Basic Cnn Architecture: Explaining 5 Layers Of Convolutional Neural Network, Https://Www.Upgrad.Com/Blog/Basic-Cnn-Architecture/, **(2022).**

[16] Kleinberg Et Al. , When Does Sgd Escape Local Minima, **(2018).**

[17] Ebubekir Seyyarer, Faruk Ayata, Taner Uçkan, Ali Karcı, Derin Öğrenmede Kullanilan Optimizasyon Algoritmalarinin Uygulanmasi Ve Kiyaslanmasi, **(2020).**

[18] Sebastian Ruder, An Overview Of Gradient Descent Optimization Algorithms ,

(2017).

[19] Gabrıel Gohuc Davis, Why Momentum Really Works, Https://Distill.Pub/2017/Momentum/, **(2017).**

[20] Fei-Fei Li, Cs231n: Deep Learning For Computer Vision, Https://Cs231n.Github.İo/Neural-Networks-1/, Stanford - Spring **(2023).**

[21] Jason Brownlee, How To Choose An Activation Function For Deep Learning, **(2021).**

[22] Ye, Y., Li, S., Liu, F., Tang, Y., Hu, W. Edgefed: Optimized Federated Learning Based On Edge Computing, **(2020).**

[23] Mayank Mishra, Convolutional Neural Networks, Explained, Towards Data Science, **(2020).**

[24] Andrew Ng , Understanding Mini-Batch Gradient Descent, **(2018).**

[25] Muhammad Asad, Ahmed Moustafa, And Takayuki Ito, Federated Learning Versus Classical Machine Learning: A Convergence Comparison, Nagoya Institute Of Technology, 466-8555, **(2021).**

[26] Abdulrahman, S., Tout, H., Ould-Slimane, H., Mourad, A., Talhi, C., Guizani, M. A Survey On Federated Learning: The Journey From Centralized To Distributed On-Site Learning And Beyond, **(2021).**

[27] Ahmet M. Elbir, Sinem Coleri, Anastasios K. Papazafeiropoulos, Pandelis Kourtessis, Symeon Chatzinotas, A Hybrid Architecture For Federated And Centralized Learning, **(2021).**

[28] Yang, Y., The Improvement Of Federated Learning In Communication Efficiency, In Highlights İn Science, Engineering And Technology, **(2018).**

[29] Ashish Rauniyar , Desta Haileselassie Hagos, Debesh Jha, Jan Erik Ha˚kegar˚d, Senior, Ulas Bagci, Senior, , Danda B. Rawat, Senior , Vladimir Vlassov, Senior , Federated Learning For Medical Applications: A Taxonomy, Current Trends, Challenges, And Future Research Directions, **(2022).**

[30] Mcmahan, H. B., Moore, E., Ramage, D., Hampson, S., Communication-Efficient Learning Of Deep Networks From Decentralized Data, **(2016).**

[31] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, Zhihua Zhang, On The Convergence Of Fedavg On Non-Iıd Data, **(2019).**

[32] Alferaidi, A., Yadav, K., Alharbi, Y., Viriyasitavat, W., Kautish, S., Dhiman, G. Federated Learning Algorithms To Optimize The Client And Cost Selections, Mathematical Problems İn Engineering, **(2022).**

[33] Adrian Nilsson, Simon Smith, Gregor Ulm, Emil Gustavsson, Mats Jirstrand, A Performance Evaluation Of Federated Learning Algorithms, **(2018).**

[34] Zhang, X., Yin, W., Hong, M., & Chen, T. , Hybrid Federated Learning: Algorithms And Implementation, **(2020).**

[35] Brendan Mcmahan And Daniel Ramage, Federated Learning: Collaborative Machine Learning Without Centralized Training Data, **(2017).**

[36] Rodolfo Stoffel Antunes, Cristiano André Da Costa, Arne Küderle, Imrana Abdullahi Yari, Björn Eskofier, Federated Learning For Healthcare: Systematic Review And Architecture Proposal, **(2022).**

[37] Vivek Kumar Prasad , Pronaya Bhattacharya Orcıd, Darshil Maru 1orcıd, Sudeep Tanwar ,Orcıd, Ashwin Verma , Arunendra Singh Orcıd, Amod Kumar Tiwari , Ravi Sharma , Ahmed Alkhayyat ,Orcıd,Florin-Emilian Țurcanu, Orcıd And Maria Simona Raboaca, Federated Learning For The Internet-Of-Medical-Things: A Survey, **(2022).**

[38] Ahmed Imteaj1,2 , Urmish Thakker3 , Shiqiang Wang4 , Jian Li5 And M. Hadi Amini1,2∗, Federated Learning For Resource-Constrained Iot Devices: Panoramas And State-Of-The-Art, **(2020).**

[39] Rumelhart, D., Hinton, G. & Williams, R. Learning Representations By Back-Propagating Errors, **(1986).**

[40] Thomas Wood, Softmax Function, Https://Deepai.Org/Machine-Learning-Glossary-And-Terms/Softmax-Layer, **(2020).**

[41] Kadir Güzel, Geri Yayılımlı Çok Katmanlı Yapay Sinir Ağları-2, **(2018).**

[42] Raed Kontar, Naıchen Shı, Xubo Yue, Seokhyun Chung, Eunshın Byon, Mosharaf Chowdhury, Judy Jın, Wıssam Kontar, Neda Masoud, Maher Noueıhed, Chınedum E. Okwudıre, Garvesh Raskuttı, Romesh Saıgal, Karandeep Sıngh, And Zhısheng Ye ,The Internet Of Federated Things (Ioft): A Vision For The Future And In-Depth Survey Of Data-Driven Approaches For Federated Learning, **(2021).**

[43] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, Raman Arora Proceedings Of The 37th International Conference On Machine Learning, Fetchsgd: Communication-Efficient Federated Learning With Sketching, (**2020).**

[44] T.-M. H. Hsu, H. Qi, And M. Brown, Measuring The Effects Of Non-Identical Data Distribution For Federated Visual Classification, **(2019).**

[45] Jason Brownlee, Understand The Impact Of Learning Rate On Neural Network Performance, **(2019).**

[46] Charu C. Aggarwal, Springer International Publishing Ag, Neural Networks And Deep Learning, Chapter 8, A Textbook, **(2018).**

[47] Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V. ,Federated Optimization For Heterogeneous Networks, **(2019).**

[48] Levent Özbek, Çağri Temuçın, Bülent Cengız Fikri Öztürk -"Synaptic Transmission: A Model On The Formation Of End-Plate Potential And A Study On Simulation", **(2019).**

[49] Ece Isik-Polat , Gorkem Polat, Altan Kocyigit, Alptekin Temizel, Evaluation and Analysis of Different Aggregation and Hyperparameter Selection Methods for Federated Brain Tumor Segmentation, **(2021).**