T.C.
MANİSA CELAL BAYAR ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ
ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI
ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ BİLİM DALI

PARMAK HAREKETLERİNİN DERİN ÖĞRENME İLE SINIFLANMASI

Yasin ALI

Danışman
DOÇ. DR. Yücel KOÇYİĞİT

MANİSA–2024

Yasin ALI	**PARMAK HAREKETLERİNİN DERİN ÖĞRENME İLE SINIFLANMASI**	**2024**

TAAHHÜTNAME

Bu tezin Manisa Celal Bayar Üniversitesi Mühendislik Elektrik Elektronik Mühendisliği Bölümü'nde, akademik ve etik kurallara uygun olarak yazıldığını ve kullanılan tüm literatür bilgilerinin referans gösterilerek tezde yer aldığını beyan ederim.

Yasin ALI

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

Simgeler:

C_t	Mevcut zaman adımındaki hücre durumu
C_{t-1}	Önceki zaman adımındaki hücre durumu
f_t	Unutma kapısının çıktısı
i_t	Giriş kapısının çıktısı
W_f	Unutma kapısının ağırlık matrisi
W_i	Giriş kapısının ağırlık matrisi
W_f	Çıkış kapısının ağırlık matrisi
σ	Sigmoid aktivasyon fonksiyonu
tanh	Hiperbolik tanjant aktivasyon fonksiyonu
o_t	Çıkış kapısının çıktısı
h_t	Yeni gizli durum
x_t	Mevcut zaman adımındaki giriş verisi
b_f	Unutma kapısının bias terimi
b_i	Giriş kapısının bias terimi
b_o	Çıkış kapısının bias terimi
S_t	Evrişim katmanından gelen sinyal
W_t	Evrişim filtresi

Kısaltmalar

CNN	Convolutional Neural Network
KNN	K-Nearest Neighbors
SVM	Support Vector Machine
LDA	Linear Discriminant Analysis
DT	Decision Tree
RF	Random Forest
LR	Logistic Regression
FCNN	Fully-Connected Neural Network
LSTM	Long Short-Term Memory

MI	Mutual Information
TDD	Time Domain Descriptors
sEMG	Surface Electromyography
RMSE	Root Mean Square Error
F1-Score	Harmonik ortalama metrik değeri
ICA	Independent Component Analysis
R-Square	Determination Coefficient

ŞEKİLLER

TABLOLAR

TEŞEKKÜR

Bu araştırmada bana büyük destek sağlayan, tavsiyeleri ve yönlendirmeleriyle bana çok şey katan, değerli bilgilerinden faydalandığım Doç. Dr. Yücel KOÇYİĞİT'e, her zaman yanımda olan ve zor zamanlarımda bana destek veren annem, babam, kardeşlerim ve arkadaşlarıma, zor anlarımda her daim yanımda olan Yasemin'e, lisansüstü eğitimime devam etmem için beni teşvik eden arkadaşım Yusuf'a ve bana yardım eden herkese en içten teşekkürlerimi sunuyorum.

Yasin ALI

Manisa, 2024

ÖZET

Yüksek Lisans Tezi

Yasin ALI

Manisa Celal Bayar Üniversitesi

Lisansüstü Eğitim Enstitüsü

Elektrik-Elektronik Mühendisliği Anabilim Dalı

Danışman: Doç.Dr. Yücel KOÇYİĞİT

Bu çalışma, EMG sinyalleri ve derin öğrenme modeli kullanılarak parmak hareketlerinin sınıflandırılmasına odaklanmaktadır. Elektromiyografi sinyalleri, kasların elektriksel aktivitesini kaydeden biyomedikal sinyallerdir ve bu sinyaller analiz ederek, protez uzuvların üretilmesi, koşulların teşhis edilmesi ve hatta protez uzuvların kontrol edilmesi gibi çeşitli alanlarda kullanılır. Bu çalışmada kullanılan veriler, on sağlıklı gönüllüden toplanan verilerdir. Her gönüllü, parmak hareketiyle ilgili yedi görevi gerçekleştirdi ve her görev beş kez tamamlandı; bu, tüm gönüllülerden alınan toplam deney sayısının 350 olduğu anlamına geliyor. Elektromiyografi sinyalleri kullanılarak verileri kaydedildi ve daha sonra bu veriler işlenerek kullanılabilir hale getirildi. Bu çalışmada veriler işlendikten sonra sınıflandırmaya hazır hale getirilmiştir. Öncelikle her bir parmağa ait veriler ayrılınmıştır ve bazı parmağa bir kanaldan fazla ait olduğu için FastICA yöntemi kullanarak bu kanalların verileri teke düşürmüştür. Daha sonra, zaman alanı tanımlayıcıları (TDD), evrişimli sinir ağı (CNN) ve uzun kısa süreli bellek (LSTM) kullanılarak belirli öznitelikler çıkarıldı. Bu öznitelikler çıkarıldıktan sonra sınıflandırma işlemi için en önemli öznitelikler karşılıklı bilgi yöntemi (MI) kullanılarak seçilmiştir. Bu sürecin ardından tam bağlantılı sinir ağı modelinin (FCNN) seçilerek sınıflandırma gerçeklenirmiştir. Modelin etkili bir şekilde öğrenilmesi ve yüksek doğruluk oranlarının elde edilmesi nedeniyle sınıflandırma işlemi başarılı olmuştur. Bu sonuçlar elde edildikten sonra derin öğrenme modellerinin yaşamsal belirtilerin ve EMG sinyallerinin sınıflandırılmasında etkili bir araç olabileceği sonucuna varılabilir. Bu çalışmadan, Derin öğrenme ile sınıflama ve analiz alanlarda etkili katkılar sağlayabileceği sonucuna varılabilir.

Anahtar Kelimeler: Elektromiyografi, parmak hareketleri sınıflandırması, derin öğrenme, konvolüsyonel sinir ağları, uzun kısa süreli bellek, gerçek zamanlı sinyal işleme, bağımsız bileşen analizi.

ABSTRACT

M.Sc.

Yasin ALI
Manisa Celal Bayar University

Graduate Education School

Department of Electrical and Electronic Engineering

Supervisor: Assoc. Prof. Dr. Yücel KOÇYİĞİT

This study focuses on the classification of finger movements using EMG signals and deep learning model. Electromyography signals are biomedical signals that record the electrical activity of muscles, and these signals are analyzed and used in various fields such as the production of prosthetic limbs, diagnosing conditions, and even controlling prosthetic limbs. The data used in this study is the data collected from ten healthy volunteers. Each volunteer performed seven tasks related to finger movements, and each task was completed five times; this means that the total number of experiments from all volunteers is 350. The data was recorded using electromyography signals, and then these data were processed and made usable. In this study, the data was made ready for classification after being processed. First, the data belonging to each finger was separated, and since more than one channel belonged to some fingers, the FastICA method was used to reduce the data of these channels to one. Then, specific features were extracted using time domain descriptors (TDD), convolutional neural network (CNN), and long short-term memory (LSTM). After extracting these features, the most important features for the classification process were selected using the mutual information method (MI). After this process, the classification was performed by selecting the fully connected neural network model (FCNN). The classification process was successful due to the effective learning of the model and the high accuracy rates. After these results were obtained, it can be concluded that deep learning models can be an effective tool in the classification of vital signs and EMG signals. It can be concluded from this study that deep learning can provide effective contributions to the fields of classification and analysis.

Keywords: Electromyography, finger movement classification, deep learning, convolutional neural networks, long short-term memory, real-time signal processing, independent component analysis

1. GİRİŞ

Son yıllarda, öznitelikle derin öğrenme tekniklerine erişimle birlikte yapay zekâ alanında gelişmeler ilerleme kaydetti. Derin öğrenme, basit ve küçük veri kümelerinde öne çıkan makine öğreniminin aksine, büyük veri kümelerinden önemli kalıpları çıkarma konusunda üstündür. Derin öğrenme, zamansal veri analizi alanı, hayati sinyal işleme alanı ve görüntü işleme alanı gibi birçok alanda ilerleme ve devrim getirmiştir [1]. Elektromiyografi (EMG) sinyalleri, insan kaslarının elektriksel aktivitesini kaydeden hayati tıbbi sinyallerdir. Bu sinyalleri, kasların ne zaman ve nasıl çalıştığını detaylı olarak kaydederler; sınıflandırma ve analiz süreçleri, teşhis alanları, protez uzuv üretimi ve rehabilitasyon süreçleri [2]. Derin öğrenme modelleri, sinyalleri işleme ve veri kümelerinden veri çıkarma konusunda mükemmel performansa sahiptir. Örneğin CNN modeli ve LSTM modeli önemli özniteliklerin çıkarılmasında üstünlük göstermektedir. CNN, sinyallerin uzaysal özniteliklerini öğrenirken, LSTM zaman içindeki bağımlılıkları yakalamada en etkili yöntemdir. Bu modeller büyük veri setlerinin işlenmesi, analiz edilmesi ve sınıflandırılmasında etkili ve güçlü performans göstermektedir [3]. Ayrıca bu çalışmada, 10 sağlıklı gönüllüden toplanan her parmak için EMG sinyalleri kullanıldıktan sonra sınıflama işleminin başarılı olması için derin öğrenme modelleri ve diğer araçlar kullanıldı. İlkinde verileri ayırıp kullanılabilir hale getirmek için Fast Independent Component Analysis (FastICA) kullanıldı. Daha sonra sınıflandırma süreci için önemli özniteliklerin çıkarılması amacıyla zaman alanı tanımlayıcıları (Time Domain Descriptors) TDD ve derin öğrenme modelleri Evrişimli Sinir Ağı (Convolutional Neural Networks) ve uzun kısa süreli bellek (Long Short-Term Memory) LSTM kullanıldı. Seçilen öznitelikler arasından en önemli öznitelikleri seçmek için karşılıklı bilgi (Mutual Information) MI kullanıldı. Yüksek sınıflandırma sonucu gösteren derin öğrenme modeli, tam bağlantılı sinir ağı (Fully Connected Neural Networks) FCNN kullanılarak sınıflandırma işlemi gerçeklenmiştir. Derin öğrenme modelleri, makine öğrenimi modelin aksine büyük veri setlerinden öğrenme yetenekleri sayesinde EMG sinyallerinin analizinde yüksek doğruluk sağlarken, sinyallerin daha geniş ölçekte kullanılmasına da olanak tanıyor [4]. Bu çalışma, EMG sinyallerini kullanarak parmak hareketlerini sınıflandırmada derin öğrenme teknikleri kullanılmaktadır. Bu biyomedikal mühendisliği, rehabilitasyon ve protez kontrol sistemleri gibi alanlara önemli katkılar sağlanır.

1.1. Literatür Taraması: EMG Sinyalleri ve Derin Öğrenme Yöntemleri

Kasların elektriksel aktivitesini izlemek için elektromiyografi sinyalleri kullanılır. Bu sinyal biyomedikal sinyaldir ve robotik, protez cihazlar, teşhis vakaları ve diğer alanlarda kullanılmaktadır. 2000'li yıllardan itibaren, Kas hareketleri doğru tahmin amacı ile kullanılan bir yöntemdir. Kas aktivasyonundan elde edilen sinyalin işlenmesi ve sınıflaması için en uygun yöntemler ve modeller seçilmektedir. Bu modelden biri derin öğrenme modelidir. Derin öğrenme modeli sınıflama ve öznitelikler çıkarmak için yaygın bir yoldur öznitelikle görsellerin sınıflaması ya da EMG sinyalin işlemesi ve sınıflaması. Derin öğrenme ile EMG sinyallerin sınıflaması, son yıllarda sıkça kullanılan modeldir çünkü elde edilen sonuçlar çok yüksek doğruluk gösterir ve öznitelik çıkarma alanında kendine bir yol çizerek şimdilik sıkça kullanılan yöntem oldu. Buda protez alanlarda, teşhislerde, hastalıklarda ve farklı biyomedikal çalışmalarda büyük fayda kattı.

EMG sinyallerin sınıflama alanında sıkça ve yaygın kullanılan modelden biri **Convolutional Neural Networks** (CNN) modelidir. Derin öğrenme algoritması CNN, çok geniş alanında kullanılmakta, özellikle görüntü ve sinyal işlemesinde sinyallerin uzaysal öznitelikleri öğrenme imkânı ile büyük başarı sağlar. Biyomedikal verilerde kullanınca, ham sinyallerden direkt anlamlı öznitelikler çıkarma yeteneği sahiptir [5].

2016 yılında Atzori ve arkadaşları, yüzey elektromiyografisi (sEMG) ile elde edilen verileri kullanarak protez uzuvları kontrol etmek için derin öğrenme yöntemlerini denediler. Atzori tarafından yürütülen bu çalışmada 67 sağlıklı kişi ve 11 ampute kişi kullanıldı. Sınıflandırmada evrişimsel sinir ağları kullanılmış ve bu sınıflandırma sonucunda sağlıklı bireylerde %66, ampute bireylerde ise %38 doğruluk oranına ulaşılmıştır. Dolayısıyla CNN'in sahada daha yüksek doğruluk elde etme yeteneğine sahip olduğu sonucuna varılmıştır [6].

2024 yılında **Triwiyanto et al.** Ve ark. Yüzey EMG sinyallerini kullanarak el hareketi sınıflandırma performansını artırmak için bir derin öğrenme modeli geliştirdiler. Bu veriler on sağlıklı kişiden elde edildi. Bu çalışmada sınıflandırma işleminde doğruluk oranı yaklaşık %97'ye ulaşan evrişimli sinir ağları kullanılmış olup bu doğruluk diğer yöntemlere göre daha yüksektir. Dolayısıyla bu çalışmada CNN'in ham sEMG verilerini dahi işlemede iyi olduğu sonucuna varılabilir. Öznitelik çıkarma olmadan [7].

Reza Azhiri ve ark. “EMG Sinyallerinin Tekrarlayan Sinir Ağları (RNN'ler) Aracılığıyla Gerçek Zamanlı Sınıflandırılması” adlı bir çalışma yapmışlar ve EMG sinyallerinin gerçek zamanlı sınıflandırılmasında tekrarlayan sinir ağlarını kullanmışlardır. Bu çalışmada, zaman-frekans alanı boyunca özniteliklerin çıkarılması için ayrık bir dalgacık dönüşümü kullanılmıştır. Öznitelikler çıkarıldıktan sonra RNN'ler kullanılarak sınıflandırılmış ve 600 milisaniye içerisinde %96'luk doğruluk oranı elde edilmiştir. Bu da RNN modellerinin hız ve doğruluk açısından daha iyi sonuçlar verebileceğini doğrulamaktadır [8].

EMG sinyallerin sınıflamasında sadece derin öğrenme algoritmaları kullanılmıyor, farklı algoritmalarda kullanılır ve bu algoritmalarda makine öğrenme algoritmaları ve modelleridir. Bu algoritmalardan Destek Vektör Makineleri (SVM), Yapay sinir ağları (ANN) ve Rastgele Ormanlar (Random Forest) gibi geleneksel algoritmalar, küçük verileri için daha uygun ve yüksek sonuçları verirler.

Lee ve ark. (2022) tarafından gerçekleştirilen çalışmada, **elektromiyogram (EMG)** sinyallerinin kullanıldığı el ve parmak hareketlerinin yapay sinir ağları (**ANN**) ile sınıflandırılması incelenmiştir. Çalışmada, 10 sağlıklı katılımcıdan üç farklı EMG kanalı üzerinden veriler toplanmış ve katılımcılar 10 farklı el ve parmak hareketi gerçekleştirmiştir. Elde edilen verilerden **zaman-domain (TD)** öznitelikleri çıkarılarak, ANN, destek vektör makineleri (**SVM**), rastgele orman (**RF**) ve lojistik regresyon (**LR**) algoritmaları ile sınıflandırma yapılmıştır [9].

Sonuçlara göre ANN, %94 doğruluk oranıyla en iyi performansı gösterirken, SVM %87,6, RF %83,1 ve LR %53,9 doğruluk oranlarına ulaşmıştır. Çalışmada Ann’ının, diğer yöntemlere kıyasla bireysel farklılıklardan en az etkilendiği ve katılımcılar arası doğruluk farkının daha az olduğu vurgulanmıştır. Ayrıca, sadece TD özniteliklerinin kullanılmasıyla diğer çalışmalara kıyasla kanal başına daha fazla hareket tanımlanmış ve bu yöntemle sistemin kullanım kolaylığının ve hesaplama yükünün azaltıldığı belirtilmiştir.

2020 yılında Sanjay Kumar ve ark. yüzey EMG sinyalleri ile parmak hareketleri arasındaki doğrusal olmayan ilişkileri inceleyen ve bu ilişkileri belirli uygulamalarda kullanılabilecek şekilde modelleyen bir çalışma yürüttüler. Toplanıp bir veri seti haline getirilen veriler, araştırmacıların geliştirdikleri modelde daha sonra protez uzuv geliştirme ve daha verimli hale getirme alanlarında kullanılmak üzere parmak

hareketlerini sınıflandırmak için kullanıldı [10]. Bu çalışmada (derin öğrenme kullanılarak parmak hareketlerinin sınıflandırılması) Sanjay ve arkadaşlarının yaptığı çalışmadan elde edilen veriler kullanılmıştır.

Kaslar ve parmak hareketleri arasındaki doğrusal olmayan ilişkileri öğrenmek için MRD modeli eğitilmiştir. Veri setlerindeki modelin sürekli iyileştirilmesi ile eğitim süreci gerçekleştirilir. Bu grupta SGD gibi optimizasyon algoritmaları ve model parametremizin güncellenmesi her defasında eğitim sürecinin sonunda tekrarlanır. Eğitim sürecinin sonunda kas aktivasyonu ile parmak hareketleri arasındaki ilişkileri öğrenen bir model yapılmıştır [10].

Test sürecinde modelin performansı çeşitli istatistiksel ölçümler kullanılarak analiz edilmiş ve modelin başarısı ortalama karesel hata, ortalama mutlak hata ve korelasyon katsayısı gibi çeşitli ölçümler kullanılarak değerlendirilmiştir. Model sonuçları, modelin parmak hareketlerini %91'e kadar korelasyon katmanı tahmin edebildiğini ve ayrıca bu modelin doğrusal olmayan sinerjileri çıkarmanın etkili bir yolu olduğunu kanıtladığını gösterdi [10].

Derin öğrenme modelleri yalnızca sınıflama için değil, aynı zamanda öznitelik çıkarımında da yüksek katkı sağlamaktadır. CNN ve LSTM gibi modeller, daha yüksek sınıflama başarısı elde edebilmek için anlamlı öznitelikler çıkarabilmektedir.

Ali Arı'nın 2023 yılında "Derin Öğrenme ve Zaman Alanı Tanımlayıcılarına Dayalı Özellik Çıkartımı Kullanılarak EMG Sinyali Sınıflandırması ve El Kavrama Hareketi Tanıma" başlıklı çalışmasında TDD, CNN ve LSTM modelleri kullanılarak öznitelikler çıkarılmış, bu öznitelikler MRMR yöntemiyle anlamlı olanları seçilmiş ve SVM ile sınıflandırma yapılmıştır. Yapılan sınıflandırmanın doğruluğu ise %98.34 olarak elde edilmiştir [11].

2. GENEL BİLGİLER

2.1. Yüzey Elektromiyografi

Yüzey Elektromiyografi (Surface Electromyography, sEMG), kasların elektriksel aktivitesini invaziv olmayan bir yöntemle ölçmek amacıyla kullanılan nörofizyolojik bir tekniktir. Bu yöntemde, elektrotlar cilt üzerine yerleştirilir ve altta bulunan kasların elektriksel sinyalleri kaydedilir. sEMG, kasların motor birimlerinden kaynaklanan aksiyon potansiyellerini toplar ve bu sinyaller, kas aktivitesinin zamanlaması, şiddeti ve koordinasyonu hakkında bilgi verir. Klinik ortamlarda, spor bilimlerinde ve rehabilitasyon süreçlerinde sıkça kullanılan sEMG, kas fonksiyonlarının değerlendirilmesi ve kas-iskelet sistemi hastalıklarının teşhisi için önemli bir araçtır [12].

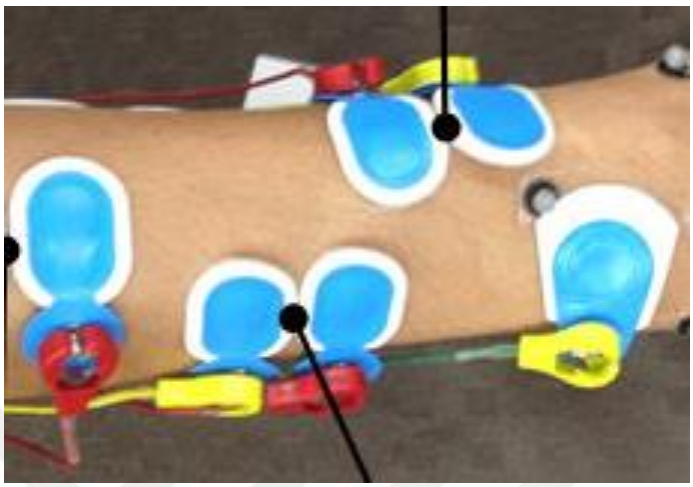
2.1.1. Yüzey Elektromiyografi'nin Tarihçesi ve Gelişimi

Elektromiyografi (EMG), 19. yüzyılın sonlarından itibaren kasların elektriksel aktivitesini incelemek için geliştirilen yöntemlerin başında gelir. İlk çalışmalar, kasların elektriksel potansiyellerinin kayıt altına alınması üzerine odaklanmış ve yüzey elektrotlarının kullanılmaya başlanmasıyla sEMG'nin temelleri atılmıştır. 20. yüzyılın ortalarında teknolojideki gelişmeler sayesinde sEMG daha hassas ve güvenilir bir teknik haline gelmiştir. Bugün, sEMG cihazları dijital sinyal işleme teknikleri ile çalışarak daha detaylı analizlere olanak tanımaktadır. Ayrıca, modern biyomedikal mühendislik uygulamaları sayesinde, sEMG sadece kas aktivitesinin ölçülmesinde değil, aynı zamanda insan-makine arayüzü ve protez kontrol sistemlerinde de kullanılır hale gelmiştir [12].

2.1.2. Yüzey Elektromiyografi'nin Temel İlkeleri

sEMG’nin temel prensibi, kaslar çalışırken motor birimlerden yayılan elektriksel sinyallerin cilt yüzeyine kadar ulaşması ve yüzey elektrotlarıyla bu sinyallerin kaydedilmesidir. Bir kasın kasılması sırasında motor nöronlar, kas liflerine elektriksel uyarılar gönderir ve bu elektriksel aktivite aksiyon potansiyelleri şeklinde cilde yansır. sEMG cihazları bu aksiyon potansiyellerini kaydeder ve analiz eder [13].

Ölçüm sırasında kullanılan elektrotların yerleşimi, cilt ve kas arasındaki dokuların öznitelikleri, sinyalin kalitesini ve doğruluğunu etkileyebilir. Bu nedenle, doğru elektrot yerleşimi ve sinyal işleme teknikleri, güvenilir sonuçlar elde etmek için kritik öneme sahiptir.

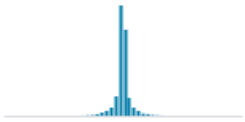
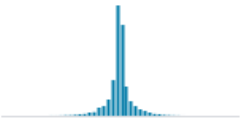


Şekil 2.1Yüzey Elektrotlar

2.2. Veri seti

Veri seti, belirli bir amaçla toplanan ve analiz edilmeye hazır olan organize verilerdir. Çeşitli değişkenler veya gözlemler içerebilir ve sayısal, kategorik ya da metinsel olabilir. Bilimsel araştırmalarda veri setinin kalitesi, boyutu ve çeşitliliği araştırmanın başarısında kritik rol oynar. Genellikle tablolar halinde düzenlenir; satırlar gözlemleri, sütunlar ise bu gözlemlerde ölçülen öznitelikleri temsil eder. sEMG gibi çalışmalarda veri setleri, kas aktivitesini ölçen sinyalleri içerebilir.

Veri seti oluşturma süreci, araştırma amacına uygun doğru verilerin toplanmasıyla başlar. sEMG çalışmalarında, elektrotların doğru yerleştirilmesi ve cihazların uygun kullanımı veri kalitesi için önemlidir. Ayrıca, denek seçimi ve örnekleme yöntemleri veri setinin temsil gücünü ve genellenebilirliğini etkiler. Veri analiz süreci ise ham verilerin işlenmesi, eksik verilerin tamamlanması ve ardından istatistiksel yöntemlerle anlamlı sonuçlar çıkarılması gibi adımları içerir. Veri setinin kalitesi, araştırma sonuçlarının doğruluğunu ve tekrarlanabilirliğini etkiler.

# channel1	# channel2	# channel3	# channel4
			
0 0	0 0	0 0	0 0
1e-05	-2e-05	-1e-05	-3e-05
1e-05	-2e-05	-1e-05	-3e-05
-1e-05	1e-05	2e-05	0
-1e-05	1e-05	2e-05	0
-1e-05	1e-05	2e-05	0
-1e-05	1e-05	2e-05	0
-1e-05	1e-05	2e-05	0
-1e-05	1e-05	2e-05	0
-1e-05	1e-05	2e-05	0
-1e-05	1e-05	2e-05	0

Şekil 2.2Veri Seti Örneği

2.3. Parmak hareketlerin sınıflaması

Parmak hareketlerinin sınıflandırılması, insan-makine etkileşimi, protez kontrolü, sanal gerçeklik uygulamaları ve nörobilim alanlarında önemli bir araştırma konusudur. Parmak hareketleri, öznitelikle el becerileri gerektiren görevlerde önemli rol oynar ve bu hareketlerin doğru bir şekilde sınıflandırılması, robotik sistemlerde veya biyomedikal cihazlarda başarılı bir insan-makine arayüzü sağlamak açısından kritik öneme sahiptir. Bu sınıflandırma, genellikle hareket verilerinin toplanması ve bu verilerin farklı algoritmalar kullanılarak analiz edilmesi ile gerçekleştirilir.

Parmak hareketlerinin sınıflandırılması süreci, hareket verilerinin toplanmasıyla başlar. Bu veriler genellikle elektromiyografi (EMG) sinyalleri, ivmeölçerler ve jiroskoplar gibi cihazlarla toplanır. EMG sinyalleri, parmakların kas aktivitesini kaydederken; ivmeölçer ve jiroskoplar, hareketin uzaydaki konumu ve hızı gibi verileri sağlar. Toplanan verilerden sinyal genliği, frekansı, ivme ve açı gibi öznitelikler çıkarılarak sınıflandırma algoritmalarına iletilir.

Sınıflandırma aşamasında çeşitli algoritmalar kullanılır. Destek Vektör Makineleri (SVM), doğrusal olmayan problemlerde başarılı olup hareket verilerini sınıflara ayırır. Yapay Sinir Ağları (ANN), öznitelikle derin öğrenme tabanlı algoritmalar, karmaşık hareketleri öğrenmede etkilidir. K-En Yakın Komşu (KNN) algoritması, hareketlerin sınıflandırılmasında en yakın komşuları temel alır. Bu algoritmalar, parmak hareketlerini tanıyarak sistemlerin daha doğru çalışmasını sağlar.

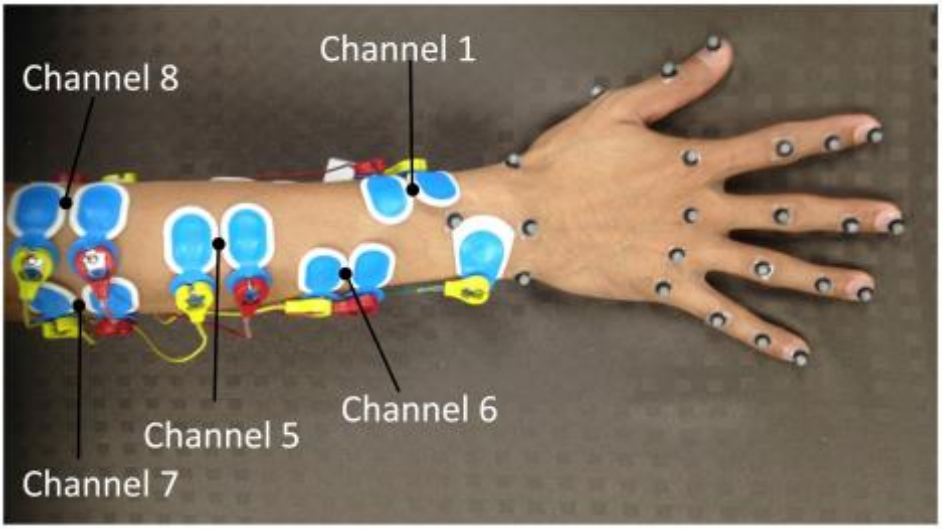
3. MATERYAL VE YÖNTEMLER

3.1. Materyal

3.1.1. Veri Toplama ve Ön İşlemesi Süreci

Sanjay Kumar Dwivedi ve ark. tarafında bir veri set toplanıldı. Çalışmada toplanan veriler, yaşları 26 ila 31 arasında değişen 10 sağlıklı gönüllüden kaydedilmiştir [10]. Gönüllüler her bir parmak hareketini yaparken, parmaklardaki kas aktivasyonu EMG sinyali olarak kaydedilmiştir. Aynı anda kinematik verileri toplamak için parmaklara 23 farklı eklem noktası yerleştirilmiştir. Böylece hem kas aktivasyonuna ait veriler hem de kinematik veriler toplandı.Parmaklara ait EMG sinyalleri Şekil 3.1 'de gösterildiği gibi elektrot bağlantıları yapılarak elde edilmiştir. Bu elektrotlar farklı kanalları oluşturmak üzere Tablo 3.1'de verilen el/parmak hareketlerine ait kas gruplarının üzerine yerleştirilmiştir.

Araştırmanın temelini oluşturan ve kaydedilen verilerin kalitesini doğrudan etkileyen kritik aşama veri toplama sürecidir. Bu süreçte elde edilen ve toplanan veriler, ileride modellerin eğitimi ve değerlendirilmesi için kullanılır.



Şekil 3.1 Veri Setin toplamasında kullanılan kanallar

Tablo 3.1Yüzey Elektrotların Hangi Kasların Üzerinde Yerleştirildiğini Belirlenmesi

Kanal	Hedef Kas	El/Parmak
1	Baş parmak uzun abdüktör kası (APL)	Baş parmak abdüksiyonu, ekstansiyonu
2	Radius bilek bükücü kası (FCR)	Bilek, el fleksiyonu
3	Yüzeysel parmak bükücü kas (FDS)	2-5. parmak PIP fleksiyonu
4	Derin parmak bükücü kas (FDP)	2-5. parmak DIP fleksiyonu
5	Parmak ekstansör kası (ED)	2-5. parmak ekstansiyonu
6	İşaret parmağı ekstansör kası (EI)	İşaret parmağı
7	Ulnar bilek ekstansör kası (ECU)	Bilek ekstansiyonu ve abdüksiyonu
8	Radius bilek ekstansör kası (ECR)	Bilek ve baş parmak

3.1.1.1. Kas Aktivasyonlarının Kaydedilmesi

Kas aktivasyonunu ölçmek için kullanılan en yaygın yöntem, yüzeysel elektromiyografi(sEMG)dir. Veri toplama çalışmasında, doğru ve kaliteli bir şekilde sEMG sinyallerini kaydetmek için Ag-AgCl bipolar aktif elektrotlar kullanılmasına karar verilmiştir. Bu elektrotlar, kas liflerinin doğrultusuna uygun bir şekilde önkol bölgesindeki 8 farklı kasın üzerine yerleştirilirken, her elektrotla diğer elektrot arasında 20 mm mesafe bırakılmıştır. 20 mm mesafesi sinyallerin kalitesini optimize etmek için seçilmiştir.

Veri toplama sırasında, Digitex Lab. Co. Ltd. tarafından üretilen BA1104 pre-amplifikatör ve TU-4 telemetri ünitesi kas aktivasyonlarını hassas bir şekilde kaydetmek için kullanılmıştır. BA1104 pre-amplifikatör, daha hassas verilerin toplanmasını sağlamak için sEMG sinyallerini kuvvetlendirir . Bundan sonra TU-4 telemetri ünitesi kuvvetlendirilen sinyalleri kablosuz olarak alıcıya aktarır ve elde edilen sEMG sinyaller, 1Hz ile 1kHz arasında bir frekans aralığında kaydedilir. Böylece kas aktivasyonlarının doğru bir şekilde izlenmesine yeterli frekans çözünürlüğü sağlanır. Daha sonra sinyaller 12bit çözürlüklü bir A/D dönüştürücü ile analog halden dijital hale gelir. Bu işlem, sEMG sinyallerinin ileride analiz edebilmesi ve işlenebilmesi için hazırlanmıştır.

3.1.2. Verilerin Ön İşlenmesi

Toplanan ham veriler, doğrudan modellemeye uygun hale getirilebilmesi için bir dizi ön işleme adımından geçirilmiştir. Bu adımlar, sEMG verilerinin işlenmesi ve kinematik verilerin işlenmesi olarak iki ana başlık altında incelenmiştir.

3.1.2.1. sEMG Verilerinin İşlenmesi

Bu adımda, ham sEMG verilerinin kas aktivasyonlarını doğru bir şekilde temsil edebilmesi için çeşitli ön işleme adımları uygulanmıştır. Uygulanan ön işleme adımları, sinyallerin filtrelenmesi, normalizasyonu ve düzleştirilmesini içermektedir.

Bant geçiren filtrelemede, 10 Hz ile 500 Hz arasında çalışan 4. dereceden Butterworth filtresi kullanılarak ham sEMG sinyalleri filtrelendi. Bu filtreleme işlemi, sinyaldeki düşük frekanslı gürültüleri ve yüksek frekanslı parazitleri ortadan kaldırarak kas aktivasyonlarının daha net bir şekilde elde edilmesini sağlar. Yüksek frekanslı gürültülerin yanı sıra hareketlerden kaynaklanan düşük frekans bileşenlerini de yok ederek, kas aktivasyonlarının doğru bir şekilde analiz edilmesine olanak tanır.[10]

Düzleştirme ve normalize etme adımında, doğrultma (rectifier) işlemi ile önceki adımdan filtrelenmiş sinyaller pozitif değerlere çevrilmiş ve her bir sinyal, ilgili kas için elde edilen maksimum sEMG değeri ile normalize edilmiştir. Kaslardan elde edilen sinyallerin karşılaştırabilir hale gelmesi ve kas aktivasyon seviyelerinin doğru bir şekilde değerlendirilmesi için normalize etme işlemi uygulanır. Kas aktivasyonlarının her bir kas için standart bir ölçümle ifade etmesi için bu işlem yapılır.

Normalize işleminden sonra sinyaller, alçak geçiren filtreleme işleminde, 4Hz kesim frekansı olan sıfır fazlı bir alçak geçiren filtre ile süzülmüştür. Kas aktivasyon sinyallerini daha düzleştirmek ve olası gürültüyü azaltmak için bu filtre yöntemi kullanılır. Yani bu filtreleme işlemi sinyaldeki ani dalgalar ve yüksek frekans gürültüyü kaldırır ve sinyal daha stabil hale gelir.

Ön işleme adımları, ham sEMG sinyallerinden kas aktivasyonunu başarılı bir şekilde tahmin etmek için gerekli olan işlenmiş verilerin elde edilmesine yönelik olarak gerçekleştirilir. Bu işlenmiş veriler, kas aktivasyonu modelinde kullanılacak verilerin temelini oluşturur. sEMG sinyalleri bu şekilde işlendiğinde, modelin doğruluk oranı artar ve kinematik verilerle olan ilişkiyi güçlendirir.

3.1.3. Sonuç Değerlendirme

Veri toplama ve ön işleme süreci, bu çalışmanın temel bileşenlerinden biridir. sEMG sinyalleri ve kinematik verilerin doğru bir şekilde toplanması ve işlenmesi, modelin doğruluğunu ve güvenilirliğini doğrudan etkileyen faktörlerdir. Elde edilen veriler, yüksek boyutlu sinyallerin düşük boyutlu temsillerinin oluşturulmasında ve bu temsillerin, karmaşık motor kontrol görevlerinde kullanılmasında büyük rol oynamıştır. Bu süreç, kas aktivasyonları ile parmak kinematikleri arasındaki ilişkilerin doğru bir şekilde çıkarılmasını sağlamakta ve bu ilişkilerin daha ileri analizler için kullanılabilmesine olanak tanımaktadır.

3.2. Yöntemler

3.2.1. Veri Toplama ve Hazırlık

Çalışmamızda, 10 farklı katılımcı (9 erkek, 1 kadın) üzerinde gerçekleştirilen bir deneyde elde edilen veri seti kullanılmıştır. Her katılımcı, 7 farklı görevi 5 kez tekrarlayarak toplam 35 deneme gerçekleştirmiştir. Veri seti, katılımcıların hareketlerini tanımlayan kinematik veriler ve kas aktivitelerini gösteren elektromiyografi (EMG) sinyallerinden oluşmaktadır. Bu değişken, her bir deneme ve görev için 5x7 boyutunda bir matris şeklinde organize edilmiştir. Çalışmamızda, parmak hareketleriyle ilgili 5 göreve odaklanıldığından, bu matris 5x5 boyutuna indirgenmiştir. Matrisin her bir hücresi, 8 farklı kas kanalından alınan 4000 örneklik bir sinyal içermektedir. Bu kanalların bazıları, sadece parmak hareketleriyle değil, elin diğer bölgelerindeki hareketlerle de ilişkilidir.

Tablo 3.2 Parmaklara Ait Kanalların Belirlemesi

Parmak	Kanal1	Kanal2	Kanal3	Kanal4	Kanal5	Kanal6	Kanal7	Kanal8
Başparmak								
İşaret								
Orta								
Yüzük								
Serçe								

Araştırmanın odak noktası parmak hareketleri olduğu için, veri analizini kolaylaştırmak ve sonuçların daha güvenilir olması amacıyla toplam veri setinden sadece parmaklara ait 5 kanal seçilmiştir. Bu kanallar, Tablo 3.2'de listelenmiştir.

Veri analizinde, her parmağın hareketini ayrı ayrı takip etmek için farklı bir yaklaşım benimsenmiştir. İşaret parmağı için özel bir kanal oluşturulurken, orta, yüzük ve serçe parmakların sinyalleri birleştirilerek tek bir kanalda toplanmıştır. Elde edilen bu kanallar, FastICA yöntemi ile bağımsız bileşenlere ayrıştırılarak 4000x1 boyutunda tek boyutlu sinyaller elde edilmiştir.

3.2.1.1. Bağımsız Bileşen Analizi (Fast Independent Component Analysis, FastICA)

Çalışmada, çok kanallı gözlemlenen parmak hareket verilerinden anlamlı bileşenleri çıkarmak için FastICA algoritması kullanılmıştır. FastICA, diğer ICA algoritmalarına göre daha hızlı ve verimli olduğu için tercih edilmiştir. Bu algoritma, karışık sinyallerden, parmak hareketleriyle doğrudan ilişkili olan fizyolojik sinyalleri temsil eden bağımsız bileşenleri başarılı bir şekilde çıkarır [14, 15]. Öznitelikle orta, yüzük ve serçe parmaklarından elde edilen 3, 4, 5 kanallarının birleştirilerek tek bir kanala indirgenmesi, farklı parmaklardan elde edilen sinyallerin farklı güç ve gürültü seviyelerine sahip olmasından kaynaklanan dengesizliği gidererek, sonraki analizler için daha homojen bir veri seti elde etmeyi amaçlar. Bu adım, veri ön işleme sürecinde önemli bir rol oynar.

$$\boldsymbol{X = AS} \qquad \textbf{(3.1)}$$

X matrisi, 4000 zaman örneği için 3 farklı sensörden alınan verileri içerir. Bu veriler, A matrisi adı verilen bir karışım matrisi ile karıştırılmış haldedir. A matrisi, orijinal sinyallerin nasıl bir araya gelerek gözlemlenen sinyalleri oluşturduğunu gösterir. FastICA algoritması, bu karışımı çözerek orijinal sinyallere karşılık gelen bağımsız bileşenleri (S matrisi) elde eder. S matrisi, 4000 zaman örneği ve tek bir bileşenden oluşur. Bu bileşenler, sinyallerin temel yapı taşlarını oluşturur ve sinyallerin kökeni hakkında önemli bilgiler verir.

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x_{11}} & \boldsymbol{x_{12}} & \boldsymbol{x_{13}} \\ \boldsymbol{x_{21}} & \boldsymbol{x_{22}} & \boldsymbol{x_{22}} \\ \vdots & \vdots & \vdots \\ \boldsymbol{x_{4000,1}} & \boldsymbol{x_{4000,2}} & \boldsymbol{x_{4000,3}} \end{pmatrix} \qquad \textbf{(3.2)}$$

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{a_{11}} \\ \boldsymbol{a_{21}} \\ \boldsymbol{a_{31}} \end{pmatrix} \tag{3.3}$$

bağımsız bileşen matrisini elde etmek için karıştırma matrisinin tersini bulmak gerekir. ve denklem şu şekilde olur.

$$\boldsymbol{S} = \boldsymbol{WX} \tag{3.4}$$

W, karıştırma matrisinin tersidir ve bağımsız bileşenleri elde etmek için kullanılır. Bu matris, 1x3 boyutundadır.

$$\boldsymbol{W} = \boldsymbol{A^{-1}} \tag{3.5}$$

$$\boldsymbol{W} = (\boldsymbol{w_{11}} \quad \boldsymbol{w_{12}} \quad \boldsymbol{w_{13}}) \tag{3.6}$$

$$\boldsymbol{S} = (\boldsymbol{w_{11}} \quad \boldsymbol{w_{12}} \quad \boldsymbol{w_{13}}) \begin{pmatrix} \boldsymbol{x_{11}} & \boldsymbol{x_{12}} & \boldsymbol{x_{13}} \\ \boldsymbol{x_{21}} & \boldsymbol{x_{22}} & \boldsymbol{x_{22}} \\ \vdots & \vdots & \vdots \\ \boldsymbol{x_{4000,1}} & \boldsymbol{x_{4000,2}} & \boldsymbol{x_{4000,3}} \end{pmatrix} \tag{3.7}$$

$$\boldsymbol{S} = \begin{pmatrix} \boldsymbol{s_{11}} \\ \boldsymbol{s_{21}} \\ \vdots \\ \boldsymbol{s_{4000,1}} \end{pmatrix} \tag{3.8}$$

S matris, parmak hareketlerinden elde edilen EMG sinyallerinin bağımsız bileşenlerini temsil etmektedir. FastICA algoritmasını kullandıktan sonra elde edilen bileşenler, daha sonra analiz edilmek ve kaydetmek için, orijinal sinyallerin karışımından bağımsız olarak ayrıştırılmıştır. Bu adımla, tüm parmakların verileri birbirine benzer boyutlu hale getirilmiştir, yani tüm parmakların verileri artık 4000x1boyutundadır. Bu işlem verileri ön işlemeden önce dengeli ve basit hale getirmiştir.

3.2.1.2. Eğitim ve Test Verilerinin Ayrılması

FastICA'dan sonra ön işlemeyi geçmeden önce, veri setini eğitim ve test olmak üzere ikiye ayırmaktır. Eğitim seti, modelin öğrenmesi için kullanılırken; test seti, modelin hiç görmediği verilerden oluşur. Bu, modelin sadece eğitim verilerini öğrenmesini değil, daha önce görmediği veriler üzerinde de doğru tahmin yapıp yapamayacağını test etmemizi sağlar. Veri setinin belirli bir oranı (%70 eğitim ve %30 test) bu şekilde eğitim ve test olarak ayrılmıştır.

3.2.2. Veri Ön İşleme

Veri ön işleme adımında, elde edilen ham veriler, daha anlamlı ve kullanışlı hale getirilmek üzere çeşitli yöntemlere tabi tutulur. Bu süreçte, öncelikle Zaman Alanı Tanımlayıcıları (Time Domain Descriptors), Evrişimli Sinir Ağı (Convolutional Neural Network, CNN) ve Uzun-Kısa Süreli Bellek (Long Short-Term Memory, LSTM) gibi derin öğrenme modelleri kullanılarak verilerden farklı öznitelikler çıkarılır. Bu öznitelikler, verinin zaman içindeki değişimlerini ve kalıplarını yansıtır. Daha sonra, Karşılıklı Bilgi (Mutual Information) yöntemi ile bu öznitelikler arasındaki ilişkiler incelenerek, sınıflandırma için en önemli ve ayırt edici öznitelikler seçilir. Bu sayede, seçilen öznitelikler kullanılarak yapılan sınıflandırma işlemlerinde daha başarılı sonuçlar elde edilir.

3.2.2.1. Zaman Alanı Tanımlayıcıları (Time Domain Descriptors, TDD)

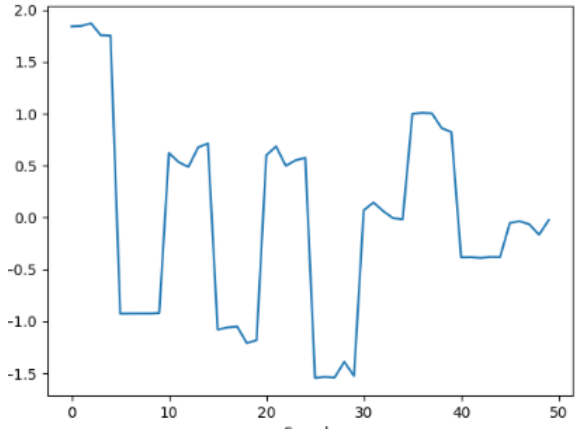
Parmak hareketlerinin karakteristik özniteliklerini ortaya çıkarmak için, her bir deneyden elde edilen veriler üzerinde Zaman Alanı Tanımlayıcıları (TDD) yöntemi uygulanmıştır. TDD, bir sinyalin ortalaması, varyansı, tepe değeri gibi zamanla ilgili istatistiksel özniteliklerini ifade eder [16]. Bu yöntem sayesinde, her bir deney için Tablo 3.3 'de belirlenen 22 farklı TDD özniteliği hesaplanmıştır. Bu öznitelikler, parmak hareketlerinin hızlanması, yavaşlaması, düzenliliği gibi farklı yönlerini temsil eder. Elde edilen bu öznitelikler, derin öğrenme modelinin eğitiminde giriş verisi olarak kullanılır. Böylece, model, parmak hareketlerinin bu farklı özniteliklerini öğrenerek daha doğru sınıflandırmalar yapabilir.

Tablo 3.3TDD ile Çıkarılacak 22 Öznitelik

Öznitelik Adı	**Açıklama**
Mean	Sinyalin ortalama değeri
Standard Deviation	Sinyalin standart sapması
Maximum Value	Sinyalin maksimum değeri
Minimum Value	Sinyalin minimum değeri
RMS (Root Mean Square)	Sinyalin karekök ortalaması
MAV (Mean Absolute Value)	Sinyalin ortalama mutlak değeri
Waveform Length	Sinyalin dalga boyu
Zero Crossings	Sinyalin sıfır geçişleri sayısı
Slope Sign Changes	Eğimin işaret değişikliklerinin sayısı
Skewness	Sinyalin çarpıklık değeri
Kurtosis	Sinyalin basıklık değeri
Variance	Sinyalin varyansı
Energy	Sinyalin enerji değeri
Entropy	Sinyalin entropisi
Mean Absolute Difference	Ortalama mutlak fark
Max Absolute Difference	Maksimum mutlak fark
Median	Sinyalin medyan değeri
Range	Sinyalin aralığı (maksimum-minimum)
P25 (Percentile 25)	Sinyalin 25. yüzdelik dilimi
P75 (Percentile 75)	Sinyalin 75. yüzdelik dilimi
IQR (Interquartile Range)	Çeyrekler arası aralık (P75- P25)
ZCR (Zero Crossing Rate)	Sıfır geçiş oranı

Mean: Sinyalin ortalama değeri

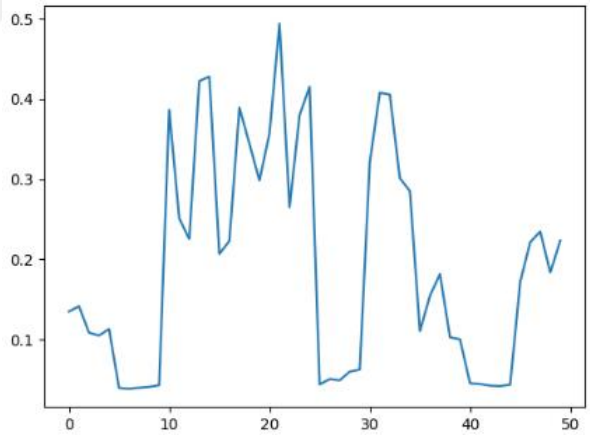
$$mean = \frac{1}{N}\sum_{i=1}^{N} x_i \quad (3.9)$$



Şekil 3.2TDD'nin Ortalama Sinyali

StandardDeviation: Sinyalin standart sapması

$$std = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - mean)^2} \quad (3.10)$$



Şekil 3.3TDD'nin Srandart Sapması

Tablo 3.4 TDD Özniteliklerin Denklemleri

Öznitelik	**Denklem**
Ortalama	$mean = \frac{1}{N}\sum_{i=1}^{N} x_i$
Standart Sapma	$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - mean)^2}$
Maksimum Değer	$Max = \max(x_i)$
Minimum Değer	$min = \min(x_i)$
Karekök Ortalaması	$RMS = \frac{1}{N}\sum_{i=1}^{N} x_i^2$
Ortalama Mutlak Değer	$MAV = \frac{1}{N}\sum_{i=1}^{N} \lvert x_i \rvert$
Dalga Formu Uzunluğu	$\sum_{i=1}^{N-1} \lvert x_{i+1} - x_i \rvert$
Sıfır Geçişleri	$\sum_{i=1}^{N-1} sign(x_i \cdot x_{i+1}) < 0$
Eğeri Yön Değişiklikleri	$\sum_{i=1}^{N-1} sign((x_i - x_{i-1}) \cdot (x_{i+1} - x_i))$
Çarpıklık	$\frac{\frac{1}{N}\sum_{i=1}^{N}(x_i - mean)^3}{std^3}$
Basıklık	$\frac{\frac{1}{N}\sum_{i=1}^{N}(x_i - mean)^4}{std^4}$

Tablo 3.5. TDD Özniteliklerin Denklemleri

Öznitelik	**Denklem**
Sinyalin varyansı	$\frac{1}{N}\sum_{i=1}^{N}(x_i - mean)^2$
Sinyalin enerji değeri	$Energy = \sum_{i=1}^{N} x_i^2$
Sinyalin entropisi	$-\sum_{i=1}^{N}(x_i^2)\cdot\log(x_i^2 + 1e - 12)$
Ortalama mutlak fark	$\frac{1}{N-1}\sum_{i=1}^{N-1}\lvert x_{i+1} - x_i\rvert$
Maksimum mutlak fark	$\max(\lvert x_{i+1} - x_i\rvert)$
Sinyalin medyan değeri	$Median = median(x_i)$
Sinyalin aralığı (maksimum-minimum)	$Range = max - min$
Sinyalin 25. yüzdelik dilimi	$P25 = percentile(x_i, 25)$
Sinyalin 75. yüzdelik dilimi	$P75 = percentile(x_i, 75)$
Çeyrekler arası aralık (P75-P25)	$percentile(x_i, 75) - percentile(x_i, 25)$

3.2.2.2. Evrişimli Sinir Ağı (Convolutional Neural Network, CNN)

3.2.2.2.1. CNN Mimarisi ve Temel İlkeler

CNN modeli, otomatik bir şekilde sinyal verilerinden öznitelikler çıkarır. Giriş katmanı, sinyal verilerini CNN modeli için uygun hale dönüştürür. Sonra da Conv1D katmanı kullanarak sinyallerdeki paternler öğrenilir ve en önemli öznitelikleri elde etmek ve çıkarmak için filtreler uygulanır. Veri boyutlarını küçültmek ve önemli bilgileri özetlemek için Havuzlama (Pooling) katmanı kullanılır. Evrişim ve Havuzlama işleminden elde edilen özniteliklerin haritalarını Düzleştirme katmanı (Flatten Layer) düzleştirir [17]. Çıkarılan özniteliklerden ileri düzeyde öznitelikler hesaplamak için Dense katmanları kullanılır. Bu adımlardan sonra CNN modeli anlamlı öznitelikler çıkarmış olur ve öznitelik seçimi ve sınıflama için hazırlanmış olur.

3.2.2.2.2. Giriş (Input) Katmanı

Giriş katmanında, CNN'in kullanılabilmesi ve çalışabilmesi için sinyaller uygun formata getirilir; bu format üç boyutludur. Dolayısıyla, sinyaller üç boyutlu formata dönüştürülür. Sinyallerin bu şekilde işlenmesi, CNN'in sinyaldeki zaman serisinden örüntüler (paternler) çıkarmasını sağlar.

$$\boldsymbol{X = \{x_1, x_2, \dots, x_n\}} \qquad \textbf{(3.11)}$$

x_i, sinyalin i. zaman dilimindeki değerini ifade eder ve n sinyalin toplam uzunluğudur.

3.2.2.2.3. Evrişimli Katman (Convolutional Layer)

Evrişimli katman, CNN'in en önemli bölümlerinden biridir. Bu katman, sinyaldeki örüntüleri analiz etmek ve tanımak için küçük filtreler uygular. Bu filtreler, sinyallerin üzerinde kayarak küçük bölümlerini inceler [18]. Böylece, filtreleme işleminde kullanılan filtreler, sinyalleri analiz ederek önemli öznitelikleri başarılı bir şekilde çıkarır. Evrişim işlemi sırasında, filtre ile sinyal arasında çarpma işlemi gerçekleştirilir ve her seferinde yeni bir öznitelik haritası elde edilir. Bu haritalar, sinyaldeki farklı örüntülerin ve frekans bileşenlerinin görselleştirilmesini sağlar.

$$\boldsymbol{S(t) = (X * W)(t) = \sum_{i=1}^{k} W(i) \cdot X(t + i - 1)} \qquad \textbf{(3.12)}$$

$X(t)$, giriş sinyalini, $W(i)$, filtrenin (i). ağırlığını ifade eder. " $*$ " sembolü, evrişim işlemini temsil eder.

3.2.2.2.4. Aktivasyon Fonksiyonu (ReLU)

Doğrusal olmayan aktivasyon fonksiyonu, CNN'de evrişim katmanından sonra uygulanır. Bu çalışmada kullanılan aktivasyon fonksiyonu ReLU fonksiyonudur. ReLU sıkça ve yaygın bir şekilde kullanılan fonksiyondur. ReLU, sinyallerdeki sadece negatif değerler üzerinde çalışarak tüm negatif değerleri sıfırlar ve pozitif değerleri olduğu gibi bırakır.

$$\boldsymbol{f(x) = \max(0, x)} \tag{3.13}$$

$\boldsymbol{x}$, evrişim katmanından gelen bir değeri ifade eder. Eğer x negatifse, ReLU çıktısı sıfır olur. Eğer x pozitifse, ReLU çıktısı x ile aynı olur.

3.2.2.2.5. Maksimum Havuzlama (Max Pooling) Katmanı

Evrişim katmanından sonra gelen havuzlama katmanı, boyutları küçültmek ve modelin daha verimli bir şekilde çalışmasını sağlamak için kullanılır. Havuzlama işlemi, sinyalin belirli bölgelerden en yüksek değeri alarak sinyali özetler. Bu sayede model, hesaplamada fazla yük harcamaz ve önemli bilgileri koruyarak aşırı öğrenme (overfitting) riskini azaltır. Havuzlama tekniği olarak en yaygın kullanılan yöntem max pooling'dir. Max pooling, yüksek frekanslı bileşenleri daha iyi koruduğu için tercih edilen bir yöntemdir.

$$\boldsymbol{y(t) = max\{S(t + i) | i = 0, \dots, p - 1\}} \tag{3.14}$$

$\boldsymbol{p}$, havuzlama penceresinin boyutunu ifade eder.

3.2.2.2.6. Düzleştirme Katmanı (Flatten Layer)

Genellikle havuzlama ve evrişim katmanlarından çıkan veriler çok boyutlu veri yapılarıdır. Bu çıkışların bağlantı katmanlarına geçmeden önce düzleştirilmesi gerekir. Çok boyutlu verilerin tek boyutlu hale dönüştürülmesi için düzleştirme katmanı uygulanır. Bu işlem, bir sinyalin öznitelik haritasının tek boyutlu bir vektöre

dönüştürülmesini sağlar. Sinyalde n tane öznitelik haritası bulunur ve her bir haritada m tane değer vardır. Bu işlemden elde edilen vektör, öznitelik haritası ile değer sayısının çarpımıyla elde edilir; dolayısıyla m*n boyutunda bir vektör oluşturulur.

3.2.2.2.7. Tam Bağlantılı Katman (Fully Connected Layer)

Tam bağlantılı katman, her bir nöronun giriş verilerini ağırlıklandırarak toplar ve ReLU aktivasyon fonksiyonu ile işleyerek ileri düzeyde bilgi çıkarır [19]. Bu çalışmada tam bağlı katman sadece öznitelik çıkarımı yapmaktadır. Sınıflama işlemi olsaydı, eğitim süreci gerçekleşecekti; ancak burada yalnızca öznitelik çıkarımı gerçekleştirilmektedir.

$$\boldsymbol{z} = \sum_{i=1}^{n} \boldsymbol{w_i} \cdot \boldsymbol{x_i} + \boldsymbol{b} \tag{3.15}$$

w_i, ağırlıkları; x_i, giriş değerlerini ve $\boldsymbol{b}$, bias terimini ifade eder. Bu toplama işlemi sonrası, aktivasyon fonksiyonu uygulanır.

$$\boldsymbol{a} = \boldsymbol{ReLU}(\boldsymbol{z}) \tag{3.16}$$

3.2.2.2.8. Çıkış Katmanı (Output Layer)

Öznitelik çıkarma işlemi, sinyallerin farklı yönlerinin analiz etmesine yardımcı olur. Çıkarılan öznitelikler ise sinyalin farklı paternlerini ve bileşenlerini temsil etmektedir.

$$\boldsymbol{P(y = j|x)} = \frac{\exp(z_j)}{\sum_{k=1}^{K} \exp(z_k)} \tag{3.17}$$

$\exp(\boldsymbol{z_j})$, sınıf $\boldsymbol{j}$ için hesaplanan değeri ifade eder ve toplam $\boldsymbol{K}$ sınıf vardır. Bu işlem, her sınıfın olasılığını hesaplanır ve sınıflandırma problemleri için kullanılır.

3.2.2.3. Uzun-Kısa Süreli Bellek (Long Short-Term Memory, LSTM)

Yöntemi ve yapısı

Zaman serisi verilerinde uzun vadeli bağımlılıkları modellemek için kullanılan LSTM, tekrarlayan sinir ağlarının (RNN) bir türüdür. Bu model, gereksiz bilgileri

unutmak ve önemli bilgileri bellekte tutmak için hücre durumu ile giriş kapısı, unutma kapısı ve çıkış kapısı kullanır [20, 21].

3.2.2.3.1. Hücre Durumu

Hücre durumu, bu modelin en önemli bileşenlerinden biridir; çünkü yalnızca zaman serisindeki taşınan bilgileri temsil etmekle kalmaz, aynı zamanda LSTM'nin hafızası görevini de üstlenir. Geçmiş bilgilerin sürekli güncel kalmasını sağlamak için, zamanın her ilerlemesinde hücre durumu güncellenir ve bu sayede bilgileri korur.

$$\boldsymbol{C_t = f_t \cdot C_{t-1} + i_t \cdot \widetilde{C}_t} \quad \textbf{(3.18)}$$

C_t, mevcut zaman adımındaki hücre durumu, C_{t-1}, önceki zaman adımındaki hücre durumu, C_{t-1}, unutma kapısının çıktısı, i_t, giriş kapısının çıktısı, $\tilde{C}_t$, hücre durumu.

3.2.2.3.2. Unutma Kapısı (Forget Gate)

Hücre durumundaki bilgilerin ne kadar kaybedileceği ve ne kadar korunacağı, unutma kapısı tarafından belirlenir.

$$\boldsymbol{f_t = \sigma\left(W_f \cdot \left[h_{\{t-1\}}, x_t\right] + b_f\right)} \quad \textbf{(3.19)}$$

f_t, unutma kapısının çıktısı, W_f, unutma kapısının ağırlık matrisi, $h_{\{t-1\}}$, bir önceki zaman adımındaki gizli durum, x_t, mevcut zaman adımındaki giriş, b_f, unutma kapısının kutuplama terimi, σ, sigmoid aktivasyon fonksiyonudur.

3.2.2.3.3. Giriş Kapısı (Input Gate)

Bu kapı, yeni bilgilerden hangilerinin hücre durumuna ekleneceğini belirler. Bu süreç, her zaman adımında bilgilerin nasıl işlendiğini kontrol eder.

Giriş kapının çıktısı.

$$\boldsymbol{i_t = \sigma\left(W_i \cdot \left[h_{\{t-1\}}, x_t\right] + b_i\right)} \quad \textbf{(3.20)}$$

$$\widehat{C_t} = \tanh\left(W_C \cdot \left[h_{\{t-1\}}, x_t\right] + b_C\right) \quad (3.21)$$

Bu işlem, mevcut hücre durumunu giriş kapısının çıktısı ile günceller ve hücreye eklenecek yeni hücre durumu adayını hesaplar.

i_t, giriş kapısının çıktısı, $\widehat{C_t}$, hücre durumu adayı, W_i ve W_C, giriş kapısının ağırlık matrisleri, b_i ve b_C, giriş kapısının kutuplama terimleri, $tanh$, hiperbolik tanjant aktivasyon fonksiyonudur.

3.2.2.3.4. Hücre Durumunun Güncellenmesi

Güncellenmiş hücre durumu

$$C_t = f_t \cdot C_{\{t-1\}} + i_t \cdot \widehat{C_t} \quad (3.22)$$

C_t, yeni hücre durumu, f_t, unutma kapısının çıktısı, $C_{\{t-1\}}$, bir önceki hücre durumu, i_t, giriş kapısının çıktısı, $\widehat{C_t}$, yeni hücre durumu adayıdır.

3.2.2.3.5. Çıkış Kapısı (Output Gate)

Bu kapı, hücrede depolanan bilgilerin ne kadarının dışarıya aktarılacağını belirler. Çıkış kapısı, modelin tahmin işlemini gerçekleştirmek için kullanılacak bilgileri kontrol eder. Modelin çıkışı, çıkış kapısının çıktısı, hücre durumu ve gizli durumdan oluşur.

$$o_t = \sigma\left(W_o \cdot \left[h_{\{t-1\}}, x_t\right] + b_o\right) \quad (3.23)$$

$$h_t = o_t \cdot \tanh(C_t) \quad (3.24)$$

o_t, çıkış kapısının çıktısı, h_t, yeni gizli durumdur.

3.2.2.3.6. Öznitelik çıkarma sonucu

Yukarıdaki modelleri ve yöntemleri kullanarak gerekli öznitelikler başarılı bir şekilde seçilir ve verileilerin sayısı azaltılmış olur. Sonuç olarak toplamda 118 özellik elde edilir: 22 TDD, 64 CNN ve 32 LSTM.

3.2.2.4. Karşılıklı Bilgi (Mutual Information, MI)

Ön işleme aşamasında, TDD, CNN ve LSTM kullanılarak toplamda çok sayıda öznitelik elde edilmiştir. Bu sayı sınıflama için yeterli olsa da, daha net ve yüksek sonuçlar elde etmek amacıyla Karşılıklı Bilgi yöntemi uygulanmaktadır. Bu yöntem, en önemli özniteliklerin seçilmesini sağlayarak, öznitelik sayısını azaltır. Böylece, sınıflama işlemi daha kolay hale gelir ve performans artışı sağlanır; ayrıca aşırı öğrenme (overfitting) riskinden kaçınılmış olur.

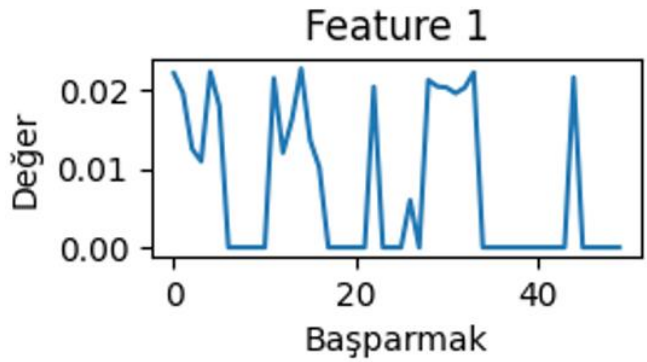
3.2.2.4.1. Karşılıklı Bilgi Hesaplaması

Bu hesaplama, bir özelliğin hedef değişken üzerindeki etkisini göstermek için o özniteliğin hedef değişkenle olan karşılıklı bilgi miktarını hesaplar. Bu sayede, hangi özniteliklerin hedef değişkenle daha fazla ilişkili olduğu belirlenir ve en önemli öznitelikler seçilerek sınıflama performansı artırılabilir.

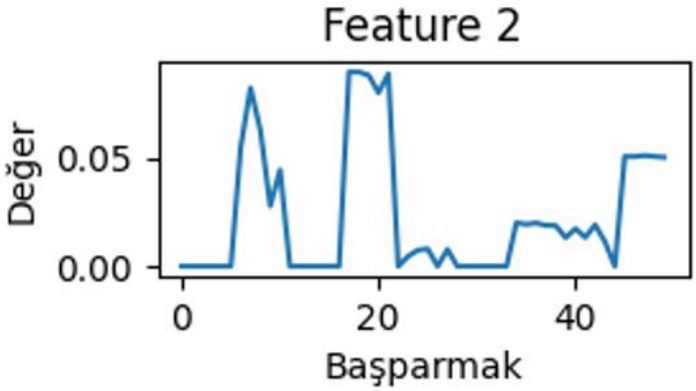
$$\boldsymbol{I(X_i;Y) = \sum_{x_i \in X_i} \sum_{y \in Y} P(x_i, y) \log \left(\frac{P(x_i,y)}{P(x_i)P(y)} \right)} \quad \textbf{(3.25)}$$

X öznitelik, Y hedef değişken, $p(x, y)$ ortak olasılık dağılımı, $p(x)$ ve $p(y)$ ise marjinal olasılık dağılımlarıdır.

MI yöntemi kullanıldığında, hesaplama sonucuna göre öznitelikler sıralanır ve en yüksek değere sahip olanlar seçilir. Bu çalışmada, fazla sayıda öznitelik seçmemek amacıyla 30 özellik belirlenmiştir. Bu seçim, sınıflama işlemi için daha uygun hale gelir ve daha yüksek performans gösterir.



Şekil 3.4 MI ile Seçilen İlk Öznitelik



Şekil 3.5 MI ile Seçilen İkinci Öznitelik

3.3. Derin Öğrenme Modeli Tasarımı

Son yıllarda yapay zekâ ve makine öğrenimi alanında yapılan gelişmeler, verilerin işlenmesi ve analizi konusunda büyük katkılar sağlamıştır. Bu alandaki yenilikler devam ederken, dikkat çekici yöntemler arasında derin öğrenme öne çıkmaktadır. Derin öğrenme, makine öğreniminden önemli ölçüde farklıdır; çünkü çok katmanlı sinir ağları kullanarak büyük ve karmaşık veri kümeleriyle başa çıkabilmektedir. Oysa makine öğrenimi, bazen büyük veri işlemede zorluk ve gecikmelerle karşılaşabilir.

Derin öğrenme, verilerdeki doğrusal olmayan yapıların modellenmesinde çok katmanlı bir yapı kullanarak, karmaşık ve büyük veri setleri üzerinde olumlu ve başarılı sonuçlar elde edilmesine yardımcı olur. Bu nedenle birçok araştırmacı, büyük veri setlerine sahip olduklarında derin öğrenme yöntemlerini tercih etmektedir.

Derin öğrenme, özellikle görüntü ve ses işleme gibi analiz alanlarında yaygın bir şekilde kullanılmaktadır. Makine öğrenimiyle karşılaştırıldığında, derin öğrenme, tıbbi sinyallerin analizi gibi zaman serisi verileriyle çalışırken de benzersiz sonuçlar ortaya koymaktadır. Bu çalışmada, parmak hareketlerinin sınıflandırılması için EMG sinyalleri kullanılarak derin öğrenme uygulanmıştır.

Öznitelik çıkarma adımında TDD ile birlikte derin öğrenmenin iki modeli olan CNN ve LSTM kullanılmıştır. Deneysel sınıflama aşamasında ise SVM ve KNN algoritmalarıyla yüksek sonuçlar elde edilmiştir. Bu durum, derin öğrenmenin pozitif etkisini ve başarılı bir şekilde özniteliklerin çıkarıldığını göstermektedir.

3.3.1. Tam Bağlantılı Sinir Ağı (Fully Connected Neural Networks FCNN) nın Diğer Modellerle Karşılaştırılması

Evrişimli Sinir Ağı **(CNN)** mimarisi, genellikle görüntü işleme ve iki boyutlu veri setlerinde yaygın olarak kullanılır. CNN, verinin uzaysal paternlerini öğrenmek için filtreler kullanır ve sinir ağının her katmanı, verideki belirli öznitelikleri çıkarmaya odaklanır. Ancak CNN, zaman serisi veriler üzerinde çok etkili değildir; çünkü zamanla değişen sinyallerde uzaysal paternlerden çok, ardışık verilerin birbiriyle olan ilişkileri önemlidir.

Yinelemeli Sinir Ağı **(Recurrent Neural Network, RNN)** ise zaman serisi verilerinin analiz edilmesinde, özellikle uzun vadeli sonuçların öğrenilmesinde başarılı sonuçlar vermekte ve oldukça uygun olup göz alıcı bir performans sergilemektedir. Ancak eğitim sırasında uzun serilerde doğru sonuçlara ulaşmanın zorluk derecesini etkileyebilecek bir sorun (gardiyan kaybı) yaşayabilir. RNN modelleri içinde bu sorunu çözen veya önemli ölçüde sorunu azaltan ancak maliyeti genellikle yüksek olan LSTM modeli bulunmaktadır.

FCNN ise her bir nöronun bir önceki katmandaki tüm nöronlara bağlı olduğu bir yapıya sahip olması ve sinyallerdeki doğrusal olmayan kalıpları öğrenebilmesi nedeniyle öğrenmede daha iyi sonuçlar vermektedir. Bu modeli en çok farklılaştıran özellik, özellikle bu çalışmanın temelini oluşturan EMG verileri gibi zaman serisi verilerindeki hesaplamaların büyük hızıdır. Bu modelin yeteneği, hesaplama hızı ve doğrusal olmayan ilişkileri öğrenme yeteneğinin yüksek olması, EMG verilerinin

sınıflandırılmasında daha iyi sonuçlar elde edilmesine yardımcı olmaktadır. Bu nedenle, her bir nöronun önceki tüm hücrelere bağlı olması avantajı ve hesaplama sürecinin basitliği ve hızı, bu modeli bu çalışmada kullanılacak en iyi seçim haline getirmektedir [22].

3.3.2. FCNN'nin Bu Çalışmada Seçilme Nedeni

Genel olarak EMG sinyalleri kasların elektriksel aktivitelerini kaydeder ve bu sinyaller çoğu zaman elektriksel aktiviteleri kaydederken kasların yaptığı harekete bağlı olarak farklı paternler ve gürültüler içerir. Bu verinin doğasında doğrusal olmayan ve karmaşık paternler bulunması nedeniyle, bu çalışmada en çok tercih edilen model FCNN olmuştur. Çünkü FCNN, doğrusal olmayan paternler ve karmaşık verilerle fazla zaman harcamadan işlem yapabilme yeteneğine sahiptir. Ayrıca, bu model, karmaşık verilerin daha hızlı ve verimli bir şekilde öğrenilmesini sağlayan, mükemmel sonuçlar elde edilmesini mümkün kılan katman yapısına sahiptir.

FCNN'nin bu çalışmaya büyük ölçüde yardımcı olacak birçok önemli özelliği bulunmaktadır. Özellikle, parmak veya kas hareketlerinden elde edilen EMG sinyallerindeki zamana bağlı değişiklikleri analiz etmek için, zaman serisi verilerindeki doğrusal olmayan ilişkilerin öğrenilmesi gibi kritik bir rol oynamaktadır. Ayrıca, verilerin boyutu ve karmaşıklığı da göz önünde bulundurulduğunda, FCNN bu tür zorluklarla başa çıkmak için etkili bir modeldir.

Bu çalışmada kullanılan veriler, on gönüllüden elde edilmiştir ve FCNN, daha az karmaşık olmasına rağmen, bu verileri karmaşık modellere kıyasla daha iyi işleyebilmektedir. FCNN'nin maliyet ve hız açısından avantajları da önemli bir faktördür: Bu model, hesaplamaları hızlı bir şekilde gerçekleştirmek için yüksek enerji gereksinimi duymaz. Böylece eğitim ve doğrulama süreçlerini daha hızlı tamamlayabilir ve büyük hesaplama maliyetleri olmadan daha iyi performans sergileyebilir.

Bu nedenle, FCNN'nin sahip olduğu bu özellikler göz önünde bulundurularak, son işlem olan parmak hareketlerini sınıflandırma işlemine karar verilmiştir. EMG sinyallerindeki karmaşık zamansal veriler ve doğrusal olmama durumlarıyla etkili bir şekilde başa çıkabilen bu modelin seçilmesi, sınıflandırma işleminin daha hızlı tamamlanmasını sağlayacak, daha iyi performans ve daha yüksek doğrulukla sonuç verecektir.

3.3.3. Katmanların Yapısı

Derin öğrenme modellerinde modele düzgün öğrenme yeteneği kazandıran ana bileşen, içindeki katmanlardır. Bu çalışmada kullandığımız FCNN modeli birkaç katmandan oluşmaktadır ve her katman verilerin sınıflandırılması, analiz edilmesi ve işlenmesinde belirli bir rol oynamaktadır. FCNN modeli üç bölümden oluşur: giriş katmanı, gizli katmanlar ve çıkış katmanı.

3.3.3.1. Giriş Katmanı (Input Layer)

Yapay sinir ağı modeli olarak kabul edilen Tam Evrişimli Sinir Ağı (FCNN) yapısının ilk katmanı olan giriş katmanı, modele sunulan verilerle bağlıdır. Bu bölümde, giriş verisi olarak EMG sinyallerinden elde edilen özellikler kullanılmıştır. Özniteliklerin elde edilmesinde kullanılan TDD, CNN, LSTM ve MI yöntemleriyle 30 farklı öznitelik ortaya çıkmıştır.

Giriş katmanının görevi, 30 adet özniteliğiöğrenme sürecine uygun bir şekilde FCNN modelinin gizli katmanlarına aktarmaktır. Böylece modelin gizli katmanlarına veri girilmesine yönelik bir yöntem sağlanmaktadır.

$$\boldsymbol{X = x_1 + x_2 + \cdots + x_{30}} \qquad \textbf{(3.26)}$$

3.3.3.1.1. Giriş Katmanının Yapısı

Giriş katmanında kullanılan yapı, veriyi işlemek için 128 nörondan oluşmaktadır. Her nöron, giriş verilerinin tamamıyla bağlantılıdır. Yani, 30 farklı özniteliğe sahip bir veri seti düşünüldüğünde, her bir öznitelik, 128 nörondan her birine bağlıdır. Bu sayede, tüm öznitelikler, nöronlar aracılığıyla işlenerek modelin öğrenme sürecine katkıda bulunur.

Matematiksel olarak bu işlem şöyle ifade edilir: Her nöron, tüm giriş özniteliklerini toplar ve her bir özniteliğe karşılık gelen bir w ağırlık ile çarpar. Sonra bu toplama bir de kutuplama (bias) adı verilen sabit bir sayı eklenir. Yani her nöron için bu işlem şu şekilde yapılır:

$$\boldsymbol{z_j = \sum_{i=1}^{30} w_{ji} x_i + b_j} \qquad \textbf{(3.27)}$$

w_{ji}, giriş öznitelikleri ile nöron arasındaki bağlantının ağırlığıdır. Bu ağırlıklar, her giriş özniteliğinin önemini belirler. x_i, giriş verisinin i'inci özniteliğidir. Yani veri setinizdeki her öznitelik (örneğin, EMG sinyali verisi) bu işlemde yer alır. b_j, her nöron için eklenen sabit bir terimdir. Bu terim, nöronun verdiği cevabı ayarlamak için kullanılır.

Yani her nöron, 30 özniteliği birer birer alır, onları belirli ağırlıklarla çarpar, toplar ve sonra bu toplama bir sabit ekler. Bu işlem, giriş verilerini daha anlamlı bir hale getirmek için yapılır.

3.3.3.1.2. Aktivasyon Fonksiyonu: ReLU

Her nöron, aldığı bu toplam değeri doğrudan kullanmaz. Bunun yerine, bir aktivasyon fonksiyonu kullanarak bu değeri işler. Burada kullanılan aktivasyon fonksiyonu ReLU (Rectified Linear Unit) adını taşır. Bu fonksiyon, nöronun aldığı değerin pozitif mi negatif mi olduğunu kontrol eder.

Eğer nöronun aldığı değer sıfırdan büyükse, bu değer olduğu gibi kullanılır. Eğer sıfırdan küçükse, sonuç sıfır olur.

$$\boldsymbol{f(z_j) = \max(0, z_j)} \tag{3.28}$$

Bu formül, nörona gelen değerin sıfırdan büyük olup olmadığını kontrol eder. Sıfırdan büyükse, değeri değiştirmeden alır. Ama sıfır veya daha küçük bir sayı geldiyse, bu değeri sıfıra çevirir. Bu, sinir ağının öğrenme sürecini kolaylaştıran bir tekniktir, çünkü sadece önemli olan verilerle çalışır.

3.3.3.1.3. Giriş Katmanında Öğrenme Süreci

Her nöronun öğrenme sürecinde, ağırlıklar w_{ji} ve bias b_j adı verilen iki temel parametre sürekli olarak güncellenir. Bu güncellemeler, sinir ağının hata yapmasını en aza indirmek için geri yayılım (backpropagation) algoritmasıyla yapılır.

Ağırlıkların güncellenmesi

$$\boldsymbol{w_{ji} \leftarrow w_{ji} - \eta \frac{\partial L}{\partial w_{ji}}} \tag{3.29}$$

Bias terimin güncellenmesi

$$b_j \leftarrow b_j - \eta \frac{\partial L}{\partial b_j} \tag{3.30}$$

η , öğrenme oranıdır; bu, modelin öğrenme hızını belirler. Küçük değerler daha yavaş öğrenmeye, büyük değerler ise daha hızlı öğrenmeye neden olur. L, modelin hata miktarını ölçen kayıp fonksiyonudur. Bu fonksiyon, modelin yaptığı hataları belirler. ∂w_{ji}, ∂b_j ağırlıklar ve bias terimleri üzerindeki türevlerdir. Yani bu türevler, modelin hatayı nasıl en aza indireceğini belirler.

3.3.3.1.4. Giriş Katmanının Önemi

Giriş katmanı, modelin ham veriyi alıp anlamlı hale getirdiği ilk aşamadır. Bu katmanda sinir ağı, verideki tüm öznitelikleri analiz ederek bunları işlenebilir bir yapıya dönüştürür. Bu çalışmada giriş katmanının temel ihtiyacı, EMG sinyallerinden elde edilen özniteliklerin modelin derin katmanlarına aktarılmasıdır. Giriş katmanındaki tam bağlantılı mimarinin özelliği, her giriş özniteliğini diğer özniteliklere bağlayabilmesidir. Bu özellikle biyomedikal sinyal analizinde önemlidir; çünkü bu verilerdeki önemli modeller girdi katmanı tarafından modelin sonraki aşamalarına taşınır. Bunu yaparken girdi katmanı, veriyi işleyerek anlamlı hale getirerek sinir ağının derin katmanlarına ilettiği için modelin en önemli parçasını oluşturur. Buna göre sinir ağı, verilerdeki karmaşık ilişkileri anlamaya başlar.

3.3.3.2. Gizli Katmanlar (Hidden Layers)

Gizli katmanlar, sinir ağı modelinin öğrenme yeteneğini artıran katmanlardır. Giriş katmanından gelen veriler bu katmanlarda işlenir ve daha karmaşık bilgiler çıkarılabilir.

3.3.3.2.1. Gizli katman yapısı

Birinci gizli katman, tamamı giriş katmanındaki tüm nöronlara bağlı olan 128 nörondan oluşur. İkinci gizli katman ise bu yapı sayesinde her aşamada verileri işler, daha gelişmiş öznitelikler çıkarır. Verilerdeki önemli bilgileri son katmana aktarır

Gizli katmanlar, sinir ağı modelinin öğrenme kapasitesini artıran katmanlardır. Giriş katmanından gelen veriler, bu katmanlarda işlenir ve daha karmaşık bilgiler

çıkarılabilir hale gelir. Bu modelde iki gizli katman bulunur ve her ikisi de ReLU aktivasyon fonksiyonuyla donatılmıştır. Bu fonksiyon, modelin doğrusal olmayan ilişkileri öğrenmesine yardımcı olur. Gizli katmanlar sayesinde model, daha derin ve karmaşık yapıları öğrenebilir.

Gizli Katman Yapısında, ilk gizli katman 128 nörondan oluşur ve giriş katmanındaki tüm nöronlarla tam bağlantılıdır. İkinci gizli katman ise 64 nöron içerir. Bu yapı sayesinde model, her aşamada veriyi işleyip daha ileri seviyede öznitelikler çıkarır ve verinin içindeki önemli bilgileri son katmana taşır.

$$\boldsymbol{z_j{}^l = \sum_{i=1}^{n_{l-1}} w_{ji}{}^l a_i{}^{l-1} + b_j{}^l} \quad \textbf{(3.31)}$$

$z_j{}^l$, l. katmandaki j. nöronun, aktivasyon fonksiyonuna uygulanmadan önceki toplam girdisidir. $w_{ji}{}^l$, l. katmandaki j. nöron ile bir önceki $l-1$. katmandaki i. nöron arasındaki bağlantının ağırlığıdır. $a_i{}^{l-1}$, bir önceki katmandaki i. nöronun aktivasyon değeridir.

$b_j{}^l$, l. katmandaki j. nöronun bias terimidir. Bu yapıyla, her katman bir önceki katmandan gelen verileri işler ve bu bilgiyi daha ileri seviyede kullanır.

3.3.3.2.2. Aktivasyon Fonksiyonu: ReLU

Her gizli katmanda, ReLU (Rectified Linear Unit) aktivasyon fonksiyonu kullanılır. ReLU, modelin doğrusal olmayan ilişkileri öğrenmesini sağlar. ReLU şu şekilde çalışır:

$$\boldsymbol{f(z_j{}^l) = \max(0, z_j{}^l)} \quad \textbf{(3.32)}$$

Yani, bir nöronun aldığı değer pozitifse olduğu gibi kullanılır, negatifse sıfıra dönüştürülür. Bu basit ama güçlü fonksiyon, karmaşık yapıları öğrenmek için çok etkilidir.

ReLU'nun avantajları:

1. Doğrusal Olmayan İlişkileri Öğrenme: ReLU, modelin doğrusal olmayan yapıları daha iyi öğrenmesine olanak tanır.
2. Gradyan Sönmesi Probleminin Azaltılması: Diğer aktivasyon fonksiyonlarına göre, ReLU daha derin sinir ağlarında gradyan sönmesi problemini azaltır ve modelin öğrenme kapasitesini artırır.

3.3.3.2.3. Düğüm Seyreltme (Dropout) Tekniği

Gizli katmanlarda aşırı öğrenmeyi (uyumu) (overfitting) önlemek için **Düğüm Seyreltme** tekniği kullanılmıştır. **Düğüm Seyreltme** , eğitim sırasında rastgele bazı nöronları devre dışı bırakarak modelin daha genel öğrenme yapmasını sağlar. Bu teknik sayesinde model, aşırı öğrenme eğiliminden kurtularak daha iyi genelleme kapasitesi kazanır.

$$a_j{}^l = \begin{cases} 0, & r_j \leq p \\ f(z_j{}^l), & r_j > p \end{cases} \tag{3.33}$$

r_j, 0 ile 1 arasında rastgele bir sayıdır. p, düğüm seyreltme oranını ifade eder. Bu modelde, %30 düğüm seyreltme uygulanmıştır, yani eğitim sırasında gizli katmandaki nöronların yarısı devre dışı bırakılmaktadır. $a_j{}^l$, gizli katmandaki j. nöronun düğüm seyreltme uygulandıktan sonraki aktivasyon değeridir.

Bu yöntem, modelin her eğitim adımında farklı nöron kombinasyonlarıyla çalışmasını sağlar ve bu sayede aşırı öğrenmenin önüne geçilir.

3.3.3.2.4. Parametre Öğrenimi ve Geri Yayılım

Gizli katmanlardaki her bir nöron, aldığı bilgileri işledikten sonra ağırlıklarını günceller. Bu işlem, geri yayılım (backpropagation) algoritmasıyla yapılır. Modelin kayıp fonksiyonu (örneğin, categorical crossentropy), her nöronun ağırlık ve bias terimlerine göre türev alınarak optimize edilir. Bu güncelleme işlemi şu şekilde yapılır:

Ağırlık güncellemesi:

$$w_{ji}^{\,l} \leftarrow w_{ji}^{\,l} - \eta \frac{\partial L}{\partial w_{ji}^{\,l}} \quad \textbf{(3.34)}$$

Bias güncellemesi:

$$b_j^{\,l} \leftarrow b_j^{\,l} - \eta \frac{\partial L}{\partial b_j^{\,l}} \quad \textbf{(3.35)}$$

L, kayıp fonksiyonudur. η, öğrenme oranıdır ve modelin ne kadar hızlı öğrendiğini belirler. Türevler, modelin hatasını minimize etmek için ağırlıklar ve biaslar üzerinde yapılan güncellemeleri gösterir.

3.3.3.2.5. Gizli Katmanların Rolü

Gizli katmanlar, modelin verilerdeki gizli paternleri öğrenmesine yardımcı olur. İlk gizli katman, giriş katmanından gelen bilgileri işler ve veriyi daha anlamlı hale getirir. İkinci gizli katman ise bu bilgileri daha derin seviyede işleyerek sınıflandırma için gerekli olan bilgilere ulaşır.

İlk Gizli Katman: 128 nörondan oluşur. Giriş katmanındaki bilgileri alıp işler. ReLU aktivasyon fonksiyonu, bu nöronların doğrusal olmayan yapıları öğrenmesini sağlar. Ayrıca %30 düğüm seyreltme kullanılarak modelin aşırı öğrenme yapması engellenir.

İkinci Gizli Katman: 64 nörondan oluşur ve ilk katmandan gelen bilgileri daha detaylı bir şekilde işler. Bu katmanda da ReLU aktivasyon fonksiyonu kullanılır ve %30 düğüm seyreltme uygulanır.

Gizli katmanlar, modelin daha karmaşık paternleri öğrenmesini sağlayan önemli bileşenlerdir. ReLU aktivasyon fonksiyonları, doğrusal olmayan ilişkileri öğrenmede etkilidir. Dropout tekniği ile modelin aşırı öğrenme eğilimleri engellenir ve daha genel bir öğrenme sağlanır. Sonuç olarak, gizli katmanlar, modelin eğitim ve test verilerinde daha iyi performans göstermesine katkıda bulunur.

3.3.3.3. Çıkış Katmanı (Output Layer)

Çıkış katmanı, modelin son aşamasıdır ve burada model, veriyi işleyip nihai kararını verir. Bu çalışmada, 5 farklı parmak hareketini tahmin etmek amacıyla çıkış katmanında 5 nöron bulunmaktadır. Her nöron bir parmak hareketini temsil eder ve model, bu hareketlerden her biri için bir olasılık değeri tahmin eder. Bu tahminler Softmax aktivasyon fonksiyonu kullanılarak normalize edilir. Softmax sayesinde, bu olasılıkların toplamı her zaman 1 olur.

3.3.3.3.1. Çıkış Katmanının Yapısı

Çıkış katmanındaki her nöron, modelin sınıflandırması gereken 5 farklı parmak hareketine karşılık gelen olasılıkları hesaplar. Bu nöronlar, gizli katmanlardan gelen bilgileri kullanarak tahminler yapar. Her bir nöron şu matematiksel formülle çalışır:

$$\boldsymbol{o_j = \sum_{i=1}^{n_{L-1}} w_{ji}{}^{L} a_i{}^{L-1} + b_j{}^{L}} \quad \textbf{(3.36)}$$

o_j, j. inci sınıfa (parmak hareketine) ait aktivasyon değeridir; yani bu, ilgili nöronun çıkardığı sonuçtur. $w_{ji}{}^{L}$, j. nöron ile önceki katmandaki i. nöron arasındaki bağlantının ağırlığıdır. $a_i{}^{L-1}$, bir önceki katmandan gelen aktivasyon değeridir. $b_j{}^{L}$, j. nöronun bias terimidir.

Bu hesaplanan değerler, her parmak hareketine karşılık gelen bir skor gibi düşünülebilir. Bu skorlar, Softmax fonksiyonu aracılığıyla olasılıklara dönüştürülür.

3.3.3.3.2. Softmax Aktivasyon Fonksiyonu

Çıkış katmanında, modelin tahmin ettiği her sınıf için bir olasılık değeri döndürülür ve bu olasılıklar Softmax fonksiyonu ile hesaplanır. Softmax fonksiyonu, her bir sınıfın olasılığını hesaplayarak, tüm sınıfların olasılıklarının toplamını 1 olacak şekilde normalize eder. Bu, modelin her sınıf için olasılıkları belirlemesini sağlar. Yani, model her bir sınıfın (örneğin, her bir parmak hareketinin) ne kadar olası olduğunu öğrenir ve en yüksek olasılığa sahip sınıfı tahmin olarak verir. Bu şekilde model, hangi sınıfın daha olası olduğunu belirleyerek, sınıflandırma kararını verir. Softmax matematiksel olarak şöyle tanımlanır:

$$\boldsymbol{P(y = j|x) = \frac{e^{o_j}}{\sum_{k=1}^{C} e^{o_k}}} \quad \textbf{(3.37)}$$

$P(y = j|x)$, modelin giriş verisinin j. sınıfa ait olma olasılığını gösterir. o_j, j. sınıf için hesaplanan skor (lojit) değeridir. C, toplam sınıf sayısıdır (bu çalışmada 5 sınıf vardır).

3.3.3.3.3. Çıkış Katmanının Hesaplanması

Çıkış katmanında, gizli katmanlardan gelen veriler işlenir ve ardından Softmax fonksiyonuna uygulanır. Örneğin, model sinyalleri aldıktan sonra şu şekilde skorlar üretir:

$$\boldsymbol{o_1, o_2, o_3, o_4, o_5} \tag{3.38}$$

Bu skorlar Softmax fonksiyonuna girerek her sınıf için bir olasılık oluşturulur.

- $P(\,y = 1 \mid x\,) =$ (Başparmak hareketi)
- $P(\,y = 2 \mid x\,) =$ (İşaret parmağı hareketi)
- $P(\,y = 3 \mid x\,) =$ (Orta parmak hareketi)
- $P(\,y = 4 \mid x\,) =$ (Yüzük parmağı hareketi)
- $P(\,y = 5 \mid x\,) =$ (Serçe parmak hareketi)

Bu tahminlere göre, model en yüksek olasılığı seçer.

3.3.3.3.4. Kayıp Fonksiyonu: Categorical Crossentropy

Modelin tahmin ettiği olasılıkların ne kadar doğru olduğunu anlamak için Categorical Crossentropy kayıp fonksiyonu kullanılır. Bu fonksiyon, modelin tahmin ettiği olasılıklar ile gerçek değerler arasındaki farkı ölçer. Eğer model, doğru sınıfı yüksek bir olasılıkla tahmin ederse kayıp düşük olur; yanlış tahmin ederse kayıp artar.

$$\boldsymbol{L = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)} \tag{3.39}$$

L, kayıp fonksiyonudur. y_i, doğru sınıfa ait etiket (bir-hot encoded, yani doğru sınıf için 1, diğerleri için 0 olur). $\hat{y}_i$, modelin tahmin ettiği olasılıktır.

Model, bu kayıp fonksiyonu ile hatalarını hesaplayarak geri yayılım süreciyle öğrenmeye devam eder.

3.3.3.3.5. Geri Yayılım ve Ağırlık Güncellemeleri

Çıkış katmanında hesaplanan kayıp, modelin tüm katmanlarındaki ağırlıkların güncellenmesini sağlar. Geri yayılım algoritması şu şekilde işler:

1. Çıkış katmanında hesaplanan kayıp, her nöronun ağırlıkları ve bias terimleri üzerinden türevi alınarak hesaplanır.
2. Bu türev, gizli katmanlara geri iletilir ve tüm katmanlarda ağırlık güncellemeleri yapılır.
3. Bu süreç modelin doğru tahminler yapmasını sağlar.

3.3.3.3.6. Çıkış Katmanın Sonuç ve Değerlendirme

Çıkış katmanı, modelin nihai kararlarını verdiği yerdir. Softmax aktivasyon fonksiyonu, modelin tahmin ettiği skorları olasılıklara çevirir ve her sinyal için hangi sınıfın daha olası olduğunu belirler. Modelin tahmin ettiği olasılıkların doğruluğu, Categorical Crossentropy kayıp fonksiyonu ile ölçülür ve geri yayılım algoritmasıyla hatalar düzeltilir.

Sonuç olarak, 5 nöronlu çıkış katmanı, her parmak hareketi için bir tahmin üretir. Softmax aktivasyon fonksiyonu, tahmin edilen skorları olasılık olarak normalize eder. Categorical Crossentropy kayıp fonksiyonu, modelin hatalarını minimize etmeye çalışır.

3.4. Modelin Derlenmesi (Compile)

Modelin derlenmesi, modelin nasıl çalışacağını belirleyen önemli bir adımdır. Bu aşamada, modelin hataları nasıl hesaplayacağını, nasıl öğrenip optimize edileceğini ve performansının nasıl değerlendirileceğini tanımlıyoruz. Bu çalışmada, derleme sürecinde üç ana bileşen kullanılmıştır: kayıp fonksiyonu (loss), optimizasyon algoritması (optimizer).

3.4.1. Kayıp Fonksiyonu: Categorical Crossentropy

Kayıp fonksiyonu, modelin tahminleri ile gerçek sonuçlar arasındaki farkı ölçer. Bu farkı minimize ederek modelin daha doğru öğrenmesini sağlar. Çok sınıflı bir sınıflandırma problemi üzerinde çalıştığımız için Categorical Crossentropy kayıp fonksiyonu kullanılır.

$$L = -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic} \log(\hat{y}_{ic}) \quad (3.40)$$

N, eğitimdeki örneklerin sayısını gösterir. C, sınıf sayısını ifade eder. y_{ic}, gerçek sınıf etiketini gösterir (bir-hot kodlaması kullanılarak; doğru sınıf için 1, diğerleri için 0 olur). $\hat{y}_{ic}$, modelin sınıf ccc için tahmin ettiği olasılık değeridir.

Bu fonksiyon, modelin her sınıf için tahmin ettiği olasılık ile gerçek sınıf arasında bir fark hesaplar. Amaç, bu farkı tüm sınıflar için mümkün olduğunca azaltmaktır.

3.4.2. Optimizasyon Algoritması: Adam

Optimizasyon algoritması, modelin ağırlıklarının her adımda nasıl güncelleneceğini belirler. Bu çalışmada Adam optimizasyon algoritması kullanılmıştır. Adam hem gradyan inişi hem de momentum tabanlı bir algoritmadır ve modelin öğrenme hızını dinamik olarak ayarlar.

Adam algoritmasının temel mantığı, modelin hatalarını en aza indirmek için ağırlıkları güncellemek üzerine kuruludur. Gradyanlar ve moment hesaplamaları yaparak ağırlıkları günceller ve bu süreç, şu adımlarla gerçekleştirilir:

3.4.2.1. Gradyanın momentum hesaplaması

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \quad (3.41)$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \quad (3.42)$$

g_t, modelin her adımda hesapladığı gradyandır. m_t, gradyanın momentumu (ilk moment tahmini). v_t, gradyanın karesel momentumu (ikinci moment tahmini). $\beta_1$ve β_2, genellikle sırasıyla 0.9 ve 0.999 olarak seçilen hiper parametrelerdir.

3.4.2.2. Ağırlıkların güncellenmesi

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \quad (3.43)$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \quad \textbf{(3.44)}$$

$$\theta_t = \theta_{t-1} - \frac{\eta \cdot \hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon} \quad \textbf{(3.45)}$$

θ_t, modelin ağırlıklarıdır. η, öğrenme oranıdır. ϵ, sıfıra bölme hatasından kaçınmak için kullanılan küçük bir sabittir.

Adam, ağırlıkları bu şekilde optimize ederek modelin hızlı ve stabil bir şekilde öğrenmesini sağlar, özellikle büyük veri setlerinde oldukça etkilidir.

3.5. Modelin Eğitimi ve Doğrulaması

Modelin eğitim ve doğrulaması, yapay sinir ağının verilerle öğrenme işlemi yaptığı ve bu öğrenme sürecinin ardından modelin ne kadar başarılı olduğunu anlamaya çalıştığımız aşamalardır. Bu süreçlerde, model önce verilerle eğitilir ve sonra öğrenilen bilgilerin test verileri üzerinde ne kadar başarılı olduğuna bakılır. Eğitim ve doğrulama işlemleri, modelin başarısını anlamak için hayati öneme sahiptir.

3.5.1. Etiketlerin Kategorik Hale Getirilmesi

Modelin çok sınıflı bir sınıflandırma problemi ile karşılaştığında, etiketlerin bir-hot kodlama yöntemi ile kategorik hale getirilmesi gerekir. Bir-hot kodlama, her sınıfı bir vektör olarak temsil eder. Örneğin, 5 sınıflı bir problemde birinci sınıf şu şekilde kodlanır:

$$y = [1, 0, 0, 0, 0] \quad \textbf{(3.46)}$$

Bu, modelin her sınıf için olasılık tahmin etmesini sağlar. Her bir sınıfın 1 ya da 0 olarak gösterildiği bu yapıyla, model doğru sınıfı tahmin etmeye çalışır.

3.5.2. Modelin Doğrulaması

Doğrulama, modelin hiç görmediği test verileri üzerinde performansını test etme aşamasıdır. Eğitim sırasında model, verileri öğrenir ve bu bilgileri test setine uygulayarak ne kadar doğru tahmin yaptığını gösterir. Doğrulama aşaması, modelin genelleme yapma yeteneğini ölçer. Eğer model, eğitim verilerine çok fazla uyum sağladıysa (aşırı öğrenme), test verileri üzerinde düşük performans gösterebilir.

Doğrulama performansı, eğitim sırasında modelin öğrendiği bilgilerin yeni veriler üzerinde ne kadar başarılı olduğunu anlamamızı sağlar

3.5.3. Performans Metrikleri: Doğruluk, Kesinlik, Anma ve F1 Skoru

Modelin performansını değerlendirmek için doğruluk (accuracy) metriği kullanılır. Doğruluk, modelin yaptığı tahminlerin ne kadarının doğru olduğunu gösterir ve şu şekilde hesaplanır:

$$\textbf{Doğruluk} = \frac{\textbf{DP} + \textbf{DN}}{\textbf{DP} + \textbf{DN} + \textbf{YP} + \textbf{YN}} \quad \textbf{(3.47)}$$

Bu metrik, modelin tahmin ettiği örneklerin ne kadarının doğru olduğunu ölçer. Çok sınıflı sınıflandırma problemlerinde doğruluk, modelin genel başarısını değerlendirmenin temel yollarından biridir.

DP (Doğru Pozitif): Modelin pozitif olarak doğru tahmin ettiği örnekler.

DN (Doğru Negatif): Modelin negatif olarak doğru tahmin ettiği örnekler.

YP (Yanlış Pozitif): Modelin pozitif tahmin yaptığı ama aslında negatif olan örnekler.

YN (Yanlış Negatif): Modelin negatif tahmin yaptığı ama aslında pozitif olan örnekler.

Modelin başarısını sadece doğruluk (accuracy) metriği ile ölçmek yeterli değildir. Öznitelikle dengesiz veri setlerinde doğruluk metriği, sınıflar arasındaki dengesizlikleri gizleyebilir. Bu nedenle, modelin her sınıf için ne kadar iyi performans gösterdiğini anlamak için kesinlik (precision), duyarlılık (recall) ve F1-skor gibi metrikler kullanılır.

Precision (Kesinlik): Modelin belirli bir sınıfı pozitif olarak tahmin ettiğinde, bu tahminlerin ne kadar doğru olduğunu gösterir.

$$\textbf{Kesinlik} = \frac{\textbf{DP}}{\textbf{DP}+\textbf{YP}} \quad \textbf{(3.48)}$$

Modelin tahmin ettiği pozitif sınıfların ne kadarının gerçekten doğru olduğunu gösterir.

Duyarlılık: Modelin gerçekten pozitif olan sınıfların ne kadarını doğru tahmin ettiğini gösterir.

$$\textbf{Duyarlılık} = \frac{\textbf{DP}}{\textbf{DP + YN}} \quad \textbf{(3.49)}$$

Modelin ne kadar doğru pozitif bulduğunu gösterir.

F1-Skor: Kesinlik ve Duyarlılığ'ın harmonik ortalamasıdır. Öznitelikle kesinlik ve duyarlılık arasında bir denge kurmak istendiğinde faydalıdır.

$$\textbf{F1 - Skor} = \textbf{2} \cdot \frac{\textbf{Kesinlik} \cdot \textbf{Duyarlılık}}{\textbf{Kesinlik} + \textbf{Duyarlılık}} \quad \textbf{(3.50)}$$

Bu metrikler, modelin her sınıf için ne kadar başarılı olduğunu ve genel performansını anlamak için önemlidir.

3.5.4. Sonuç ve Değerlendirme

Modelin eğitim ve doğrulama süreci tamamlandıktan sonra, eğitim seti üzerindeki performans ve doğrulama seti üzerindeki performans karşılaştırılır. Eğer model eğitim verileri üzerinde çok başarılı, ancak test verileri üzerinde düşük performans gösteriyorsa, model aşırı öğrenme yapmış olabilir. Bu durumda model, yalnızca eğitim verilerini ezberlemiş ve genelleme yeteneği kazanamamış demektir.

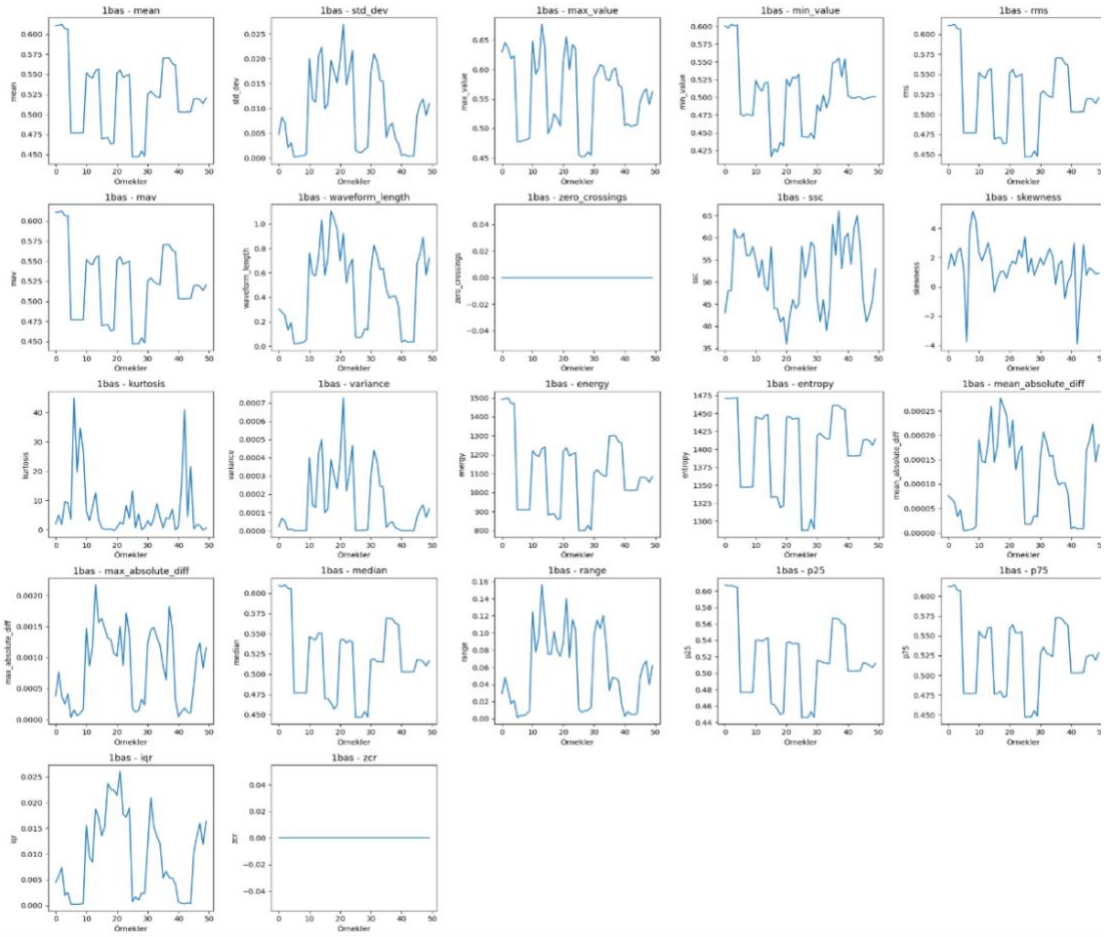
Doğruluk ile, kesinlik, anma ve F1-skor gibi metrikler de analiz edilmelidir. Bu metrikler, modelin her sınıfta ne kadar başarılı olduğunu ve sınıflar arasındaki dengesizlikleri nasıl yönettiğini gösterir. Sonuç olarak, eğitim ve doğrulama süreçleri modelin genel başarısını ve yeni veriler üzerindeki performansını anlamamıza yardımcı olur.

4. ARAŞTIRMA BULGULARI VE TARTIŞMA

4.1. Ön İşleme Sonuçları

4.1.1. TDD Öznitelikleri

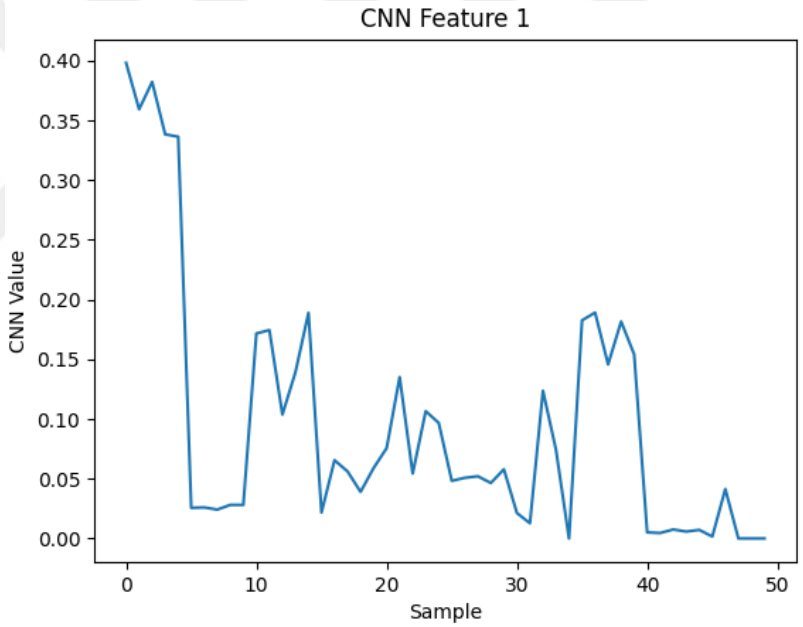
Tablo 3.3 ‘de sunulan TDD öznitelikleri, literatürde sıklıkla kullanılan ve parmak hareketlerinin karakteristiklerini etkili bir şekilde tanımlayan ölçütlerdir. Başparmağın tek bir denemesi için hesaplanan TDD özniteliklerinin görsel olarak sunulduğu Şekil 4.1, bu özniteliklerin parmak hareketinin farklı aşamalarında nasıl değiştiğini göstermektedir. Bu sayede, seçilen özniteliklerin parmak hareketinin hangi yönlerini temsil ettiği daha iyi anlaşılmaktadır. Çalışma kapsamında tüm parmaklar için benzer analizler yapılmış olup, elde edilen sonuçlar genel olarak parmak hareketlerinin karakteristik özniteliklerini temsil etmektedir.



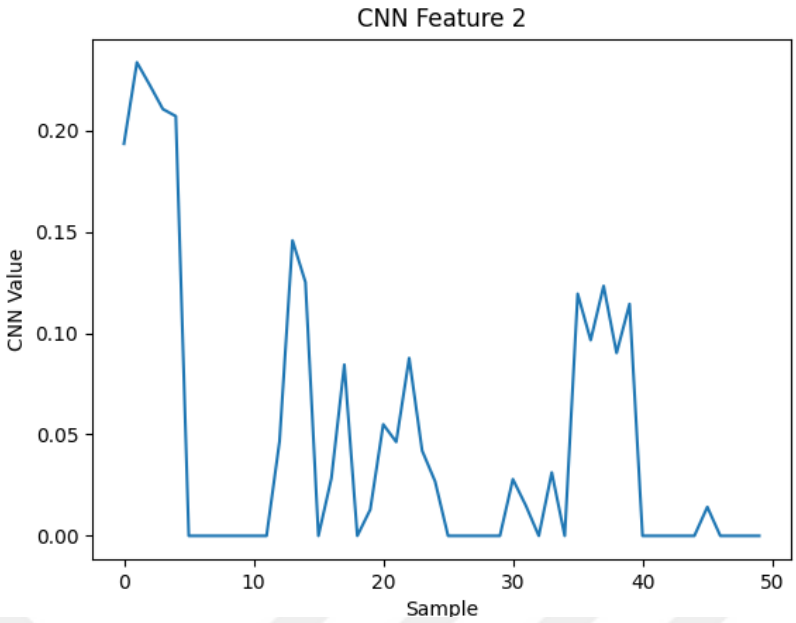
Şekil 4.1 TDD Öznitelikler

4.1.2. CNN Öznitelikler

Genellikle, CNN modeli öğrenme aşamasında filtre ağırlıklarını verileri kullanarak optimize eder. Bu optimizasyon süreci, modelin daha doğru ve genel özellikler öğrenmesine olanak tanır. Fakat bu çalışmada CNN modeli sinyalden önemli bilgiler çıkarma işlemi yapar ve bu bilgileri anlamlı vektör haline getirir. Sinyalin üzerinde uygulanan bu işlem, belirli örüntülerin, zaman aralıklarının ve frekans bileşenlerinin tanımasını sağlar. CNN modelinin işlenmesi sonunda, her bir deneme için 64 adet öznitelik çıkarılmış olur. Bu, ileride gerçekleştirilecek öznitelik seçimi, analiz ve sınıflama işlemleri için daha faydalı ve düzenli bir veri seti elde edilmesini sağlar. Baş parmak için elde edilen ilk iki öznitelik Şekil 4.2 ve Şekil 4.3, CNN modelin başarılı bir şekilde öznitelikleri çıkardığını gösterir.



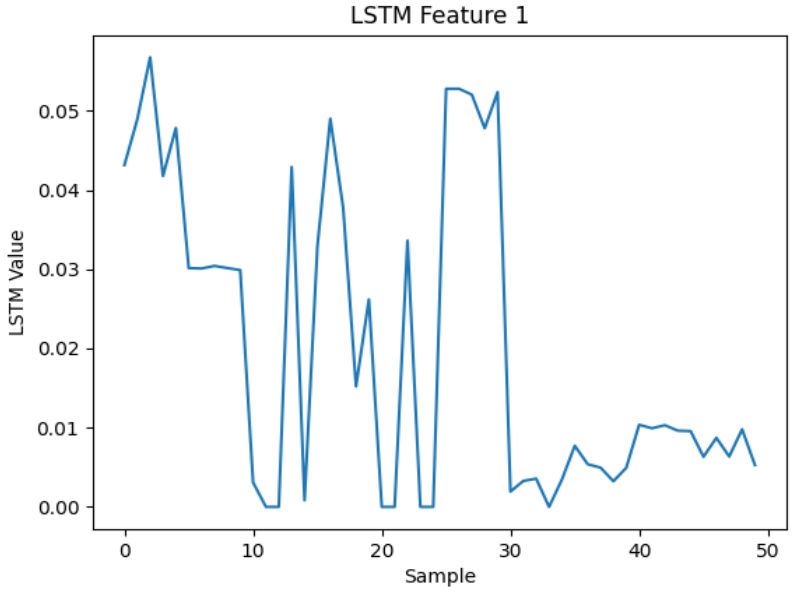
Şekil 4.2CNN'nin Çıkardığı İlk Öznitelik



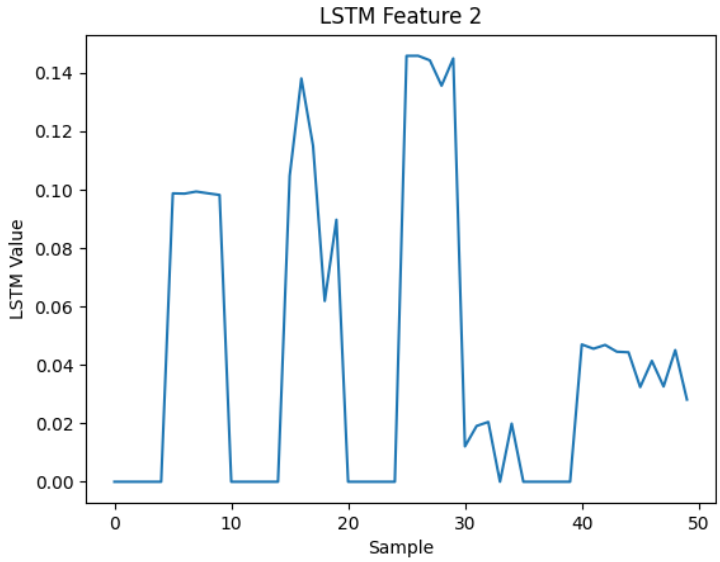
Şekil 4.3 CNN'nin Çıkardığı İkinci Öznitelik

4.1.3. LSTM Öznitelikler

LSTM modeli ile elde edilen özniteliklerin sayısı 32'dir. Bu model, her sinyalin zaman içindeki paternlerini temsil eden öznitelik vektörlerini oluşturur. Elde edilen öznitelikler, sınıflama aşamasında büyük katkı sağlayacaktır.



Şekil 4.4 LSTM'in Çıkardığı İlk Öznitelik

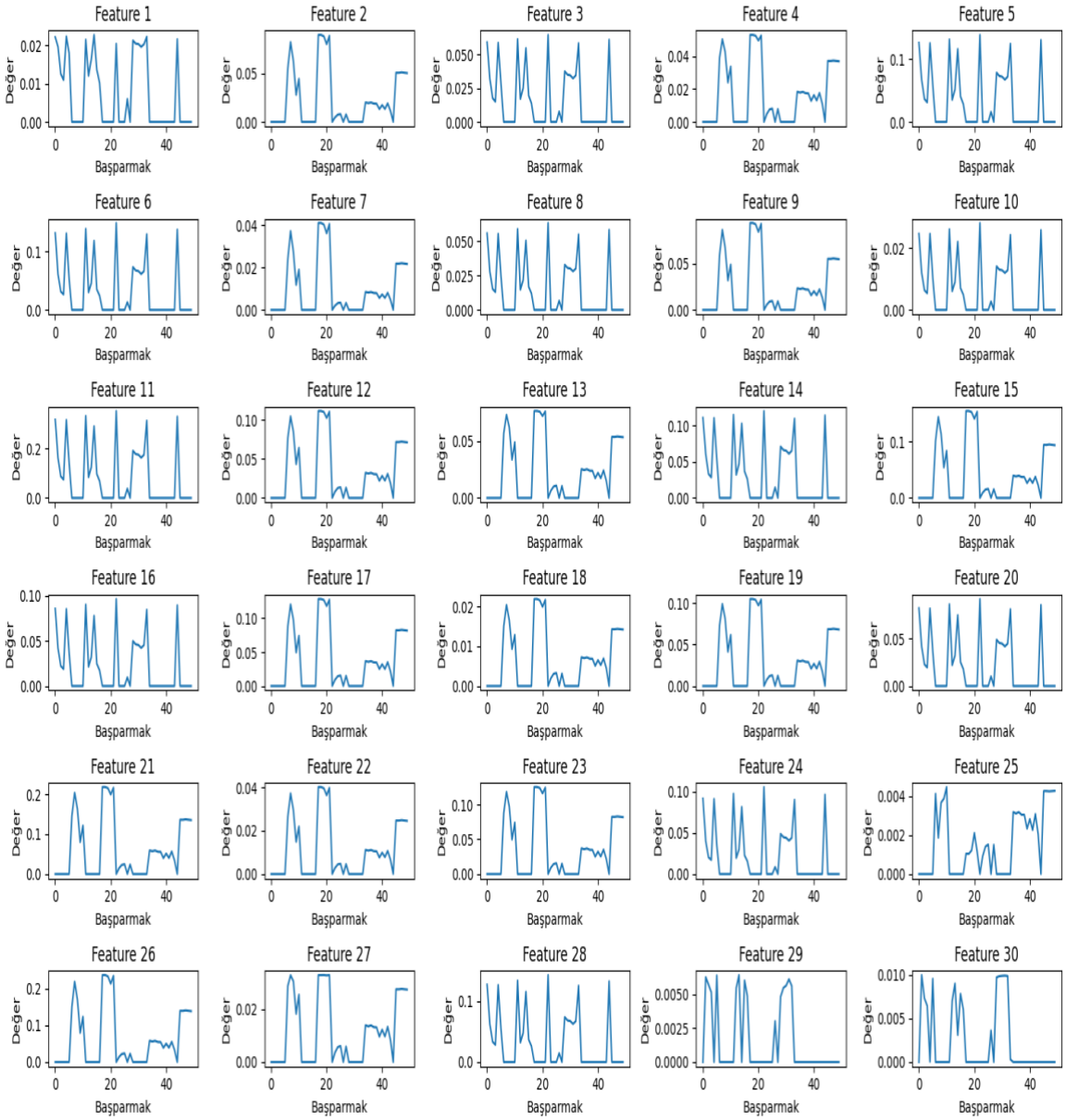


Şekil 4.5 LSTM'in Çıkardığı İkinci Öznitelik

LSTM modeli ile elde edilen özellikler, sinyal verilerindeki zaman içindeki bağımlılıkları başarılı bir şekilde yakalayarak, her sinyal için anlamlı ve yoğunlaştırılmış özellikler sunar. Bu, karmaşık ilişkilerin ve dinamiklerin daha iyi anlaşılmasına yardımcı olur.

4.1.4. MI ile Öznitelik seçimi

TDD, CNN ve LSTM ile her parmağın denemesinden çıkarılan 118 adet öznitelikten, daha anlamlı öznitelikleri elde etmek için Mutual information yöntemi kullanarak Şekil 4.6'da gösterilen sadece 30 öznitelik seçilir. Bu yöntem, sınıflama işlemini kolaylaştırır ve yüksek doğruluğa elde etmekte yardımcı olur.



Şekil 4.6 MI ile Seçilen İlk 30 Öznitelik

4.2. Makine Öğrenmesi Modelleri ile Sınıflandırma

Bu adımda, sınıflama modeli oluşturmadan ön işleme adımdan elde edilen öznitelikleri farklı yöntemler ile sınıflamayı gerçekleşerek, Burada kullanılacak modeller, Destek Vektör Makineleri (SVM) ve K-En Yakın Komşu Algoritması (KNN). Sınıflama eğitimi ve testi daha başarılı sonuç vermek ve iyice eğitmek için, eğitim %70 test ise %30 olarak ayrıldı.

Tablo 4.1 SVM Eğitim Performansı

Parmak	Kesinlik	Duyarlılık	F1-skor	Doğruluk
Başparmak	83	100	91	95
İşaret	100	97	99	
Orta	100	91	96	
Yüzük	97	91	94	
Serçe	100	97	99	

Tablo 4.1 de SVM eğitim performansı verilmektedir. Eğitim doğruluk oranı %95 olup başparmak hariç diğer parmaklarda kesinlik oranları oldukça yüksektir. Duyarlılık oranları da orta ve yüzük parmaklarda nispeten düşük olup genelde yüksektir.

Tablo 4.2 SVM Test Performansı

Parmak	**Kesinlik**	**Duyarlılık**	**F1-skor**	**Doğruluk**
Başparmak	79	100	88	95
İşaret	100	100	100	
Orta	100	93	97	
Yüzük	100	87	93	
Serçe	100	93	97	

Tablo 4.2 de ise SVM test performansı verilmektedir. Test doğruluk oranı %95 olup başparmak hariç diğer parmaklarda kesinlik oranlarının %100 olduğu görülmektedir. Duyarlılık oranları da orta, yüzük ve serçe parmaklarda nispeten düşük olup başparmak ve işaret parmakta %100 oranındadır.

Tablo 4.3 KNN Eğitim Performansı

Parmak	Kesinlik	Duyarlılık	F1-skor	Doğruluk
Başparmak	90	100	95	97
İşaret	100	97	99	
Orta	100	97	99	
Yüzük	97	94	96	
Serçe	100	97	99	

Tablo 4.4 KNN Test Performansı

Parmak	Kesinlik	Duyarlılık	F1-skor	Doğruluk
Başparmak	83	100	91	93
İşaret	94	100	97	
Orta	100	87	93	
Yüzük	93	87	90	
Serçe	100	93	97	

Tablo 4.3 ve Tablo 4.4'te KNN ile sınıflama doğru bir şekilde gerçekleşmiş ve sonuçla eğitim %97 iken test %93 olarak belirlendi.

Bu sonuçlara göre SVM ile KNN modelleri kullanarak çok yüksek doğruluk oranları elde ederek özniteliklerin ön işlemesinin doğru ve başarılı bir şekilde elde ettiğimizi göstermektedir.

4.3. FCNN ile Sınıflama Sonuçları

Çalışmamızda, FCNN modeli ve eklenen yöntemleri kullanarak eğitim doğruluğu %100 olarak elde edilirken, test doğruluk oranı %99 olarak elde edildi ve bu çok yüksek bir doğruluk orandır. Her beş parmağın performansı değerlendirmek için doğruluk, kesinlik, Duyarlılık ve F1 skor kullanılmıştır. Eğitim performansı da kesinlik, duyarlılık ve F1 skoru, beş parmak için %100 olarak elde edilmiştir.

Tablo 4.5 FCNN Eğitim Performansı

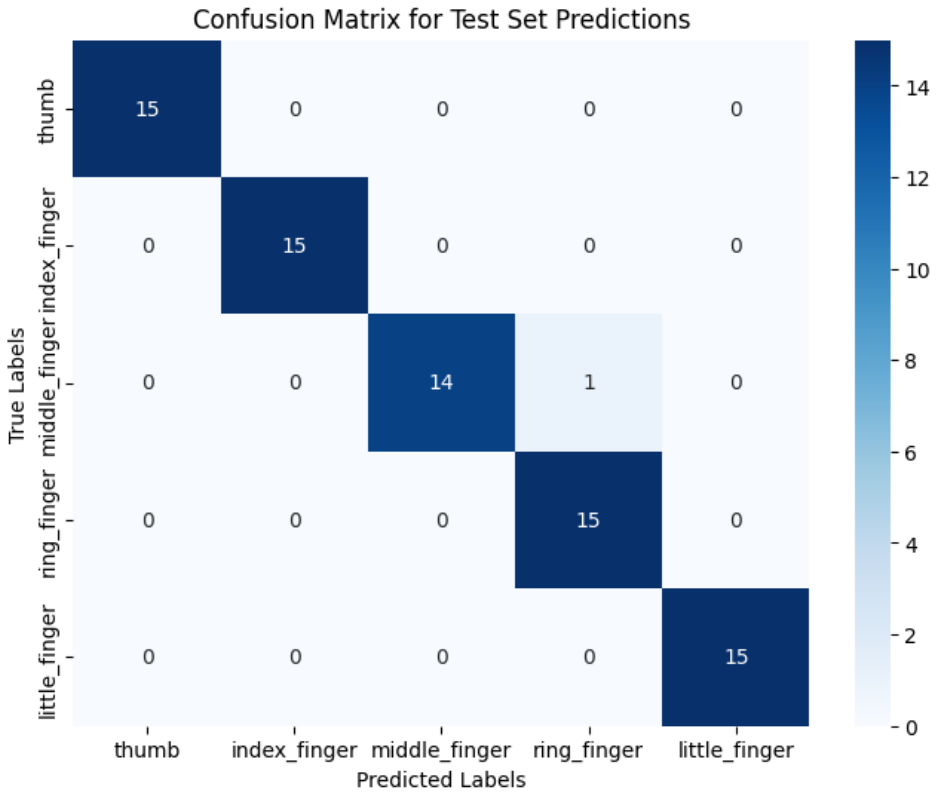
Parmak	Kesinlik	Duyarlılık	F1-skor	Doğruluk
Başparmak	100	100	100	100
İşaret	100	100	100	
Orta	100	100	100	
Yüzük	100	100	100	
Serçe	100	100	100	

Modelin eğitim sonuçlarından model iyice öğrenmiştir. Tablo 4.5'te FCNN Eğitim Performansı doğruluk oranı %100 olarak bulunurken Tablo 4.6'da FCNN Test Performansı doğruluk oranı %99 olarak belirlenmiştir.

Tablo 4.6 FCNN Test Sınıflama Performansı

Parmak	Kesinlik	Duyarlılık	F1-skor	Doğruluk
Başparmak	100	100	100	99
İşaret	100	100	100	
Orta	100	93	97	
Yüzük	94	100	97	
Serçe	100	100	100	

Modelin sınıflama sonuçları oldukça iyi ve yüksek görünmektedir. Test verileri doğru tahmin yaptığını incelemek için Şekil 4.7'de Karışıklık matrisi, modelin her sınıf için ne kadar doğru veya yanlış tahmin yaptığını tablo halinde özetleyen bir görseldir.

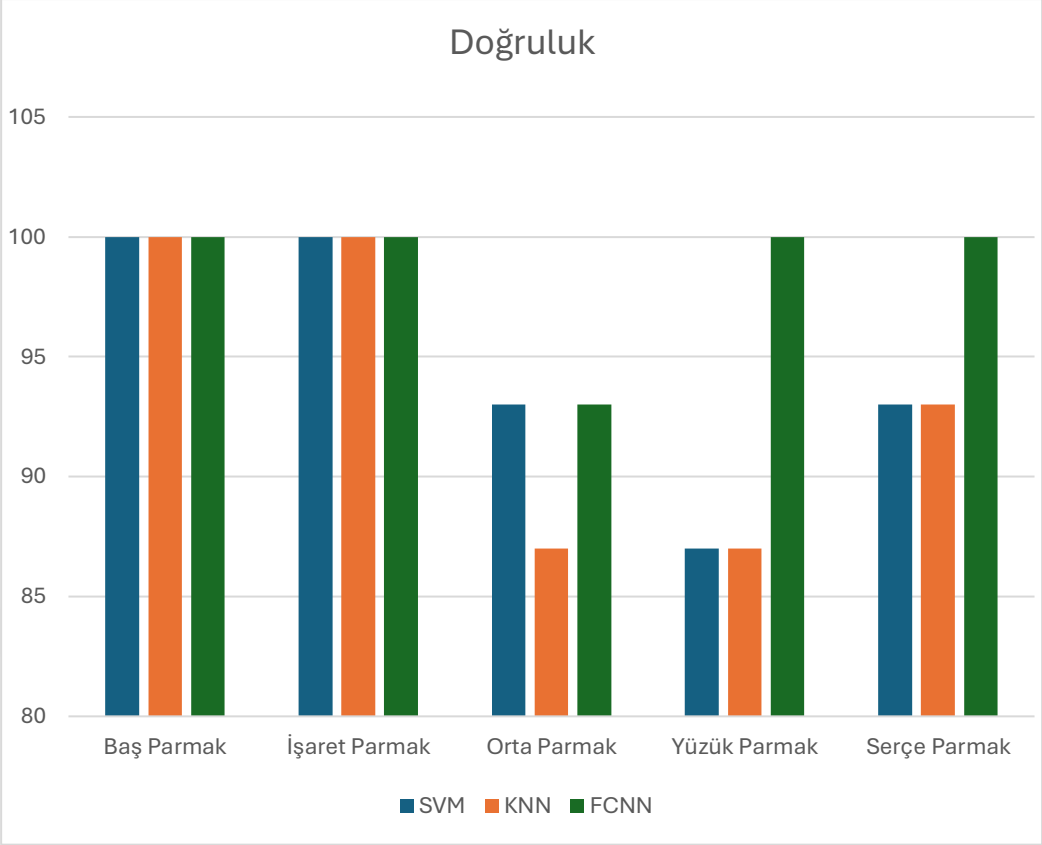


Şekil 4.7 Karışıklık Matrisi

Bu matrise göre, model test verisinde oldukça başarılı. Sadece orta parmak sınıfındaki bir örnek yanlış tahmin edilmiş, diğer tüm örnekler doğru sınıflandırılmış. Bu durumda modelin doğruluğu oldukça yüksek ve sınıflar arasında karışıklık minimal düzeyde.

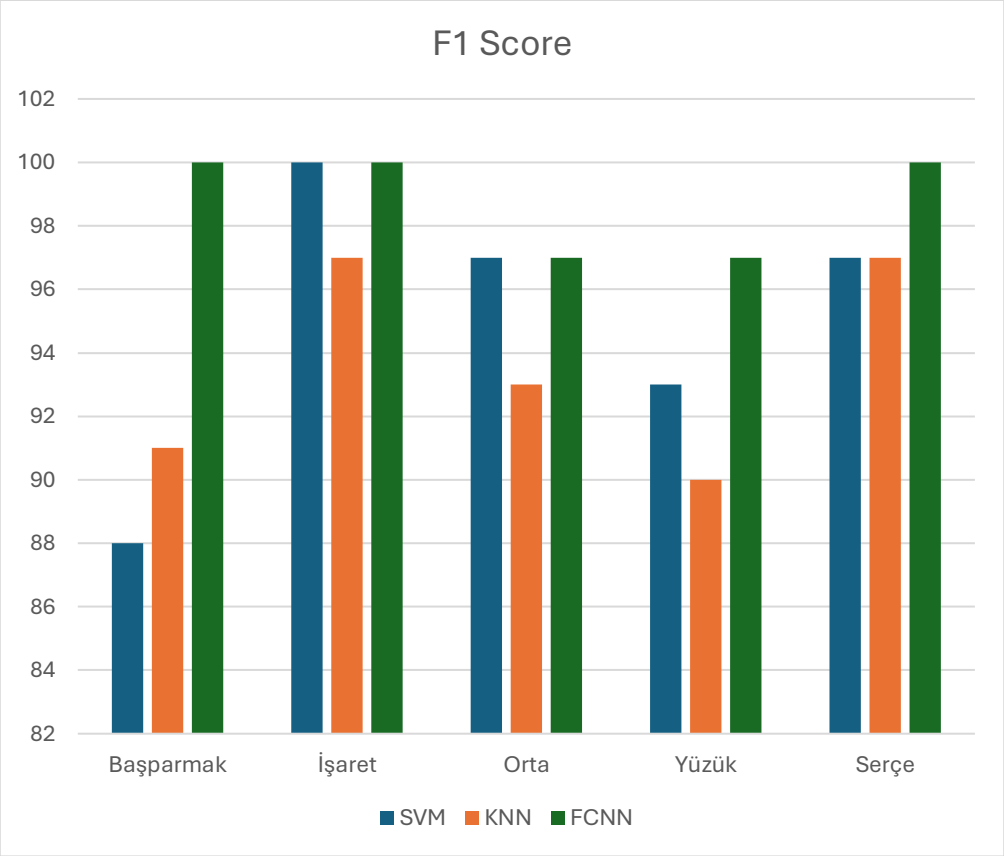
4.4. Modellerin Karşılaştırması

FCNN modeli uygulamadan, öznitelik seçiminden sonra kullanılan SVM ve KNN modeller, yine yüksek doğruluk oranları verilmişler fakat FCNN gibi değil. SVM ile sınıflama test doğruluk oranı %95 iken, KNN ile sınıflama test oranı %93 olarak elde edilmiştir. Eğitim oranları ise FCNN ile model eğitimi 100% iken, SVM ile %95 ve KNN ile model eğitim oranı %97 olarak eğitilmiştir.



Şekil 4.7 Modellerin Doğruluk Karşılaştırılması

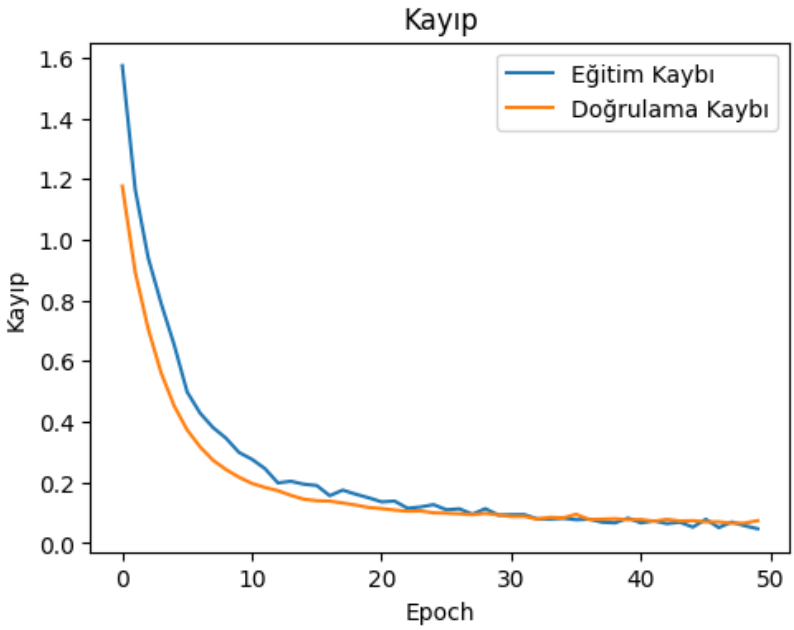
FCNN modelimiz hem eğitim hem de test aşamasında diğer sınıflama modellerden daha yüksek sonuç verilmiştir. En yüksek sonuç İşaret parmağın sonucuydu ve kullanılan üç modelden FCNN %100, SVM %100 ve KNN %97 F1-Score olarak elde edilmiştir, fakat diğer parmaklar için karışık sonuçlar verilmiştir ve en düşük başarı oranı, SVM'de %88 baş parmağı , KNN'de %90 yüzük parmağı ve FCNN' de %93 orta parmağın nasibinden çıktı. Bu sonuçlar hiç düşük değildir sadece aynı modelin F1-Score oranında parmakların arasında en düşüktü. Böylece FCNN diğer iki sınıflama modellerden bütün 5 parmakların sınıflamasında daha iyi sonuç ve daha yüksek doğruluk ve performans elde etmektedir. FCNN diğer modellerden eğitimde daha fazla süre alırsa'da sonuçları daha yüksek vermektedir. Ve daha fazla süre alma sebebi FCNN bir derin öğrenme modelidir yani çok katmanlı ve derin yapıya sahip ve öznitelikleri otomatik olarak öğretir ve büyük veri setleri için daha uygun.



Şekil 4.8 FCNN, SVM ve KNN Performans Karşılaştırılması

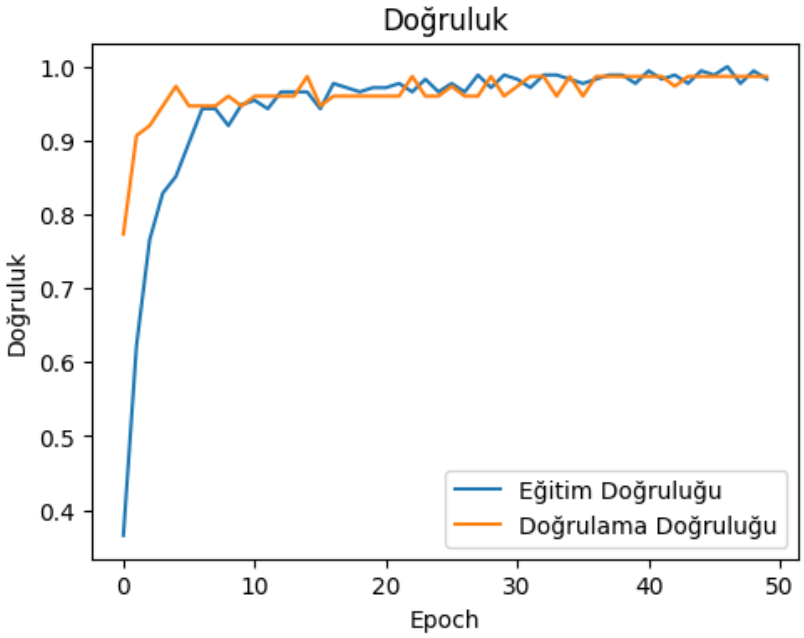
4.5. Aşırı Öğrenme Durumunun İrdelenmesi

Modelin aşırı öğrenme yapmadığını anlamanın iki yolu vardır. İlk yol, eğitim ve doğrulama kayıplarını incelemektir. Eğitim ve doğrulama kayıpları benzer bir şekilde azalıyorsa ve birbirine yakınsa, modelin aşırı öğrenme yapmadığını demektir. Şekil 4.10 aşırı öğrenme olmadığını gösterir. Fakat doğrulama kaybı bir noktadan sonra yükselmeye başlarsa, bu durum modelin eğitim verisini aşırı öğrenmeye başladığını gösterebilir.



Şekil 4.9 Eğitim ve Doğrulama Kayıpları

İkinci yol ise eğitim ve doğrulama doğruluklarını karşılaştırmaktır. Şekil 4.11, Eğitim ve doğrulama doğruluklarının yüksek ve birbirine yakın olması, modelin veriyi aşırı öğrenmeden genelleme yapabildiğini ifade eder. Ancak eğitim doğruluğu çok yüksekken doğrulama doğruluğu düşükse, modelin aşırı öğrenme yaptığı düşünülebilir.



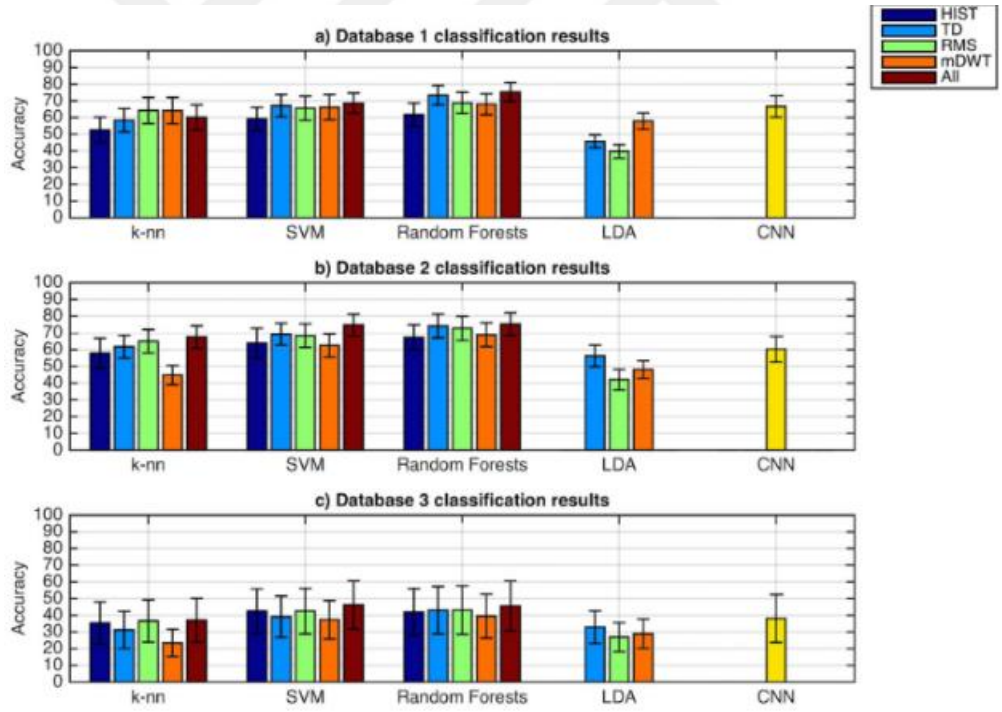
Şekil 4.10 Eğitim ve Doğrulama Doğrulukları

4.5.1. Literatür ile Karşılaştırma

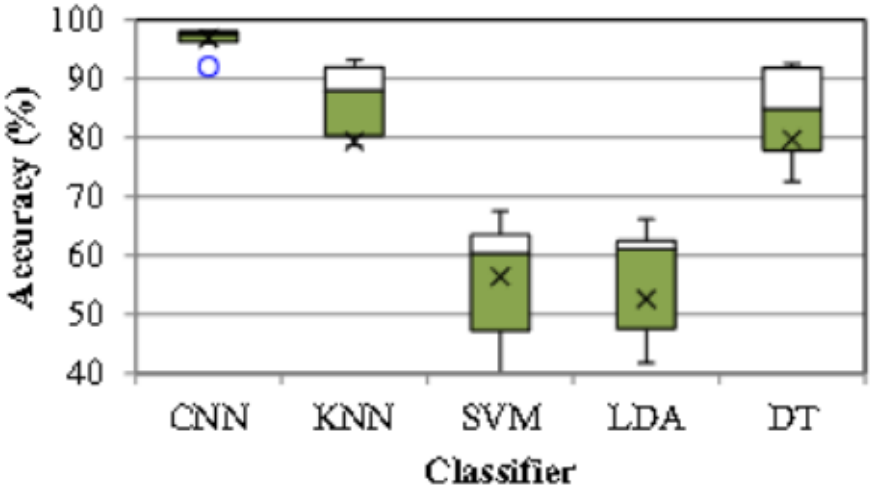
Literatürde etiketlenen çalışmalarda farklı veri setler kullanılmıştır. Bu çalışmada kullanılan veri setini toplayanlar, veri seti üzerinde farklı yöntemleri kullanarak sınıflama işlemini gerçekleştirilmişler. Literatürdeki çalışmaların farklı derin öğrenme ve makine öğrenme modelleri kullanarak çok iyi ve yüksek sonuçları elde etmişler.

4.5.2. Farklı veri seti

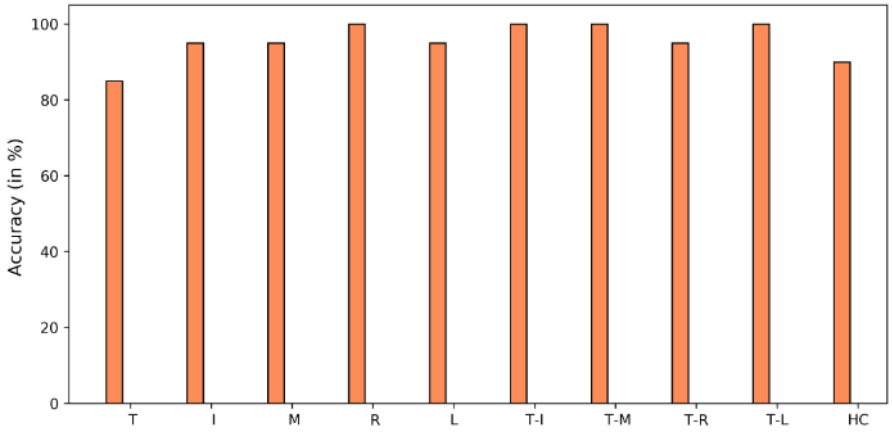
Atzori (2016) çalışmasında CNN yöntemi kullanarak en yüksek doğruluk oranı %75 olarak elde edilmiştir (Şekil 4.12). Triwiyanto et al. (2024) çalışmasında CNN kullanarak yaklaşık %97 doğruluk elde ederken(Şekil 4.13), Reza Bagherian Azhiri çalışmasında RNN yöntemi (Şekil 4.14) kullanarak oda %96 sonuç elde ederek iyi sonuçlar elde etmiştir. Makine öğrenme yöntemleri kullanarak çalışmayı gerçekleştiren Kyung Hyun Lee ve arkadaşları en son ANN ile %94 olarak elde edilmiştir [6, 7, 8, 9].



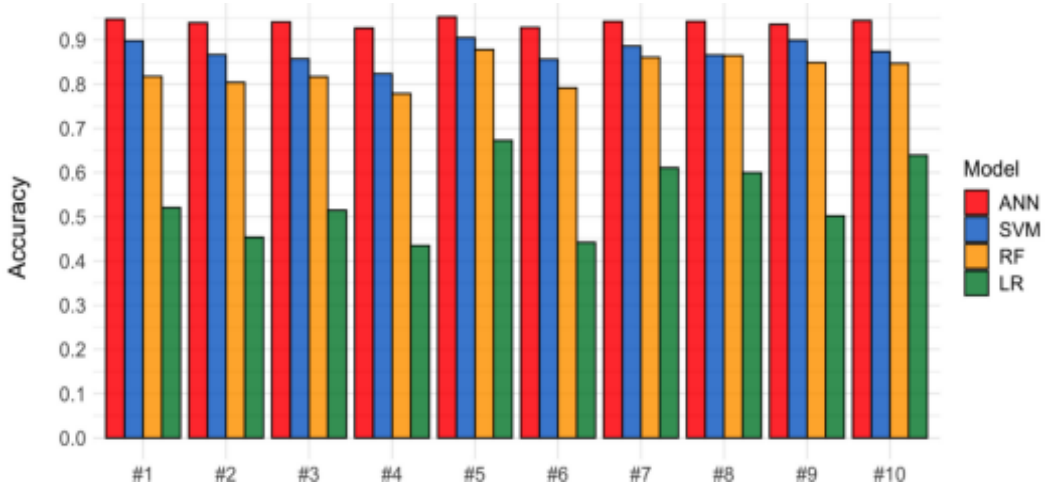
Şekil 4.11 CNN ile 3 Data setin Doğruluğu



Şekil 4.12 CNN, KNN, SVM, LDA ve DT Sınıflama Doğruluğu



Şekil 4.13 RNN Çalışmanın Sınıflama Sonuçları



Şekil 4.14 ANN, SVM, RF ve LR Sınıflama Doğruluğu

Bu sonuçlar başka veri setlerine ait olsa bile sonuçları bu çalışma için önemlidir; çünkü bu çalışmalarda görülen sonuçlar çok iyi olmasına rağmen bizim çalışmada kullandığımız FCNN modeli daha yüksek sonuçlar vermiştir.

4.5.3. Aynı Veri Seti ile Yapılan Çalışmalar

Bu çalışmada kullanılan veri setini toplayan Sanjay Kumar Dwivedi, MRD modeli ile sınıflama gerçekleştirmiştir. Çalışmada, toplanan veri seti iki tür veriden oluşmaktadır: kas aktivasyonu verileri ve kinematik hareket verileri. Araştırmacılar, bu iki veri setini birleştirerek MRD modeli ile Tablo 4.7'de görülen %91 doğruluk oranına ulaşmışlardır.

Tablo 4.7 MRD Çalışmasının Sınıflama Sonuçları

Regression Methods	RMSE	Correlation coefficient(*p*)	R-Square	Time(ms)
LR- RD	8.08 ±1.28	0.50 ±0.08	0.31 ±0.13	0.000,09 ±0.000,1
LR- Full	7.66 ±1.22	0.57 ±0.07	0.38 ±0.12	0.000,43 ±0.000,10
ANN-RD	6.64± 1.19	0.68± 0.04	0.46 ±0.10	0.06 ±0.002
ANN- Full	5.07 ±0.90	0.82± 0.04	0.68 ±0.07	0.09 ±0.010
Prosed Method	3.4 ±0.89	0.91 ±0.03	0.84 ±0.05	2.6 ±0.79

FCNN yöntemiyle, yalnızca kas aktivasyonu verileri kullanılarak sınıflama gerçekleştirilmiş ve bu yöntemle daha yüksek bir sınıflama doğruluğu elde edilmiştir; başarı oranı %99 olarak belirlenmiştir. Kas aktivasyonunun seçilmesinin sebebi, dünyada sınıflama, robotik uygulamalar, teşhis ve testler üzerine yapılan çalışmaların çoğunun EMG verileri üzerinde yoğunlaşmasıdır. Bu nedenle, gerçekleştirilen çalışmanın daha kullanışlı ve faydalı olması beklenmektedir.

4.6. Analiz ve Yorumlar

Bu çalışmanın amacı, derin öğrenme yöntemleri ile parmak hareketlerinin sınıflanmasıdır. Derin öğrenme ile en yüksek ve en iyi sonuçları elde edebilmek için üç farklı yöntem kullanılmıştır. Özniteliklerden yalnızca TDD 22 özniteliği yeterli

bulunmamış, bu nedenle CNN ve LSTM derin öğrenme modelleri de uygulanmıştır. Öznitelikler seçildikten sonra, sınıflama işlemi FCNN derin öğrenme modeli ile gerçekleştirilmiştir. Bu yaklaşım, daha başarılı ve yüksek doğruluklu sonuçlar elde edilmesini sağlamış ve çalışmanın amacına ulaşılmasına katkı sunmuştur. Elde edilen başarı, tüm parmaklar için geçerli olup, kesinlik, doğruluk ve modelin eğitim performansı açısından yüksek bir başarı oranı göstermektedir.

Modelin aşırı öğrenme (overfitting) riskinden korunması amacıyla Dropout, veriyi %70 ve %30 oranında ayırma, havuzlama (pooling), eğitim ve doğrulama kayıplarının karşılaştırılması, ayrıca eğitim ve doğrulama doğruluklarının izlenmesi gibi yöntemler kullanılmıştır. Bu yöntemler sayesinde, eğitim performansı aşırı öğrenme olmadan artmış ve model etkin bir şekilde öğrenmiştir. Eğitimde %100 doğruluk ve testte %99 doğruluk elde edilmesi arasındaki küçük fark, aşırı öğrenme olmadığını göstermektedir.

5. SONUÇ VE ÖNERİLER

Bu çalışmada, öznitelik seçimi ve sınıflama işlemleri Spyder uygulaması ve Python dili kullanılarak gerçekleştirilmiştir. Bu yöntem, aşırı öğrenme olmadan, diğer yöntemler ve modellere göre daha yüksek doğruluk ve performans sergilemiştir. Her bir parmak için performans ve doğruluk değerlendirildiğinde, FCNN modelinin eğitimi %100 doğruluk sağlamış ve toplam test doğruluğu %99 olarak kaydedilmiştir. Parmakların doğruluğu ise, baş, işaret, yüzük ve serçe %100 iken, orta parmak %93 ile en düşük doğruluk oranına sahip olarak sınıflanmıştır.

TDD zaman serisi ile öznitelikleri kolay ve basit bir şekilde elde edilebilir ve bunu yetmeden derin öğrenme modeli CNN ile başarılı 64 öznitelik elde edilmiştir dahada öznitelikleri kaçırmamak için yine derin öğrenme modeli LSTM'i kullanarak 32 öznitelik elde edildi toplam olarak 118 öznitelik yapar. Bu 118 özniteliklerden en önemli öznitelikleri seçmek için MI yöntemi kullanıldı ve 30 tane öznitelik seçilmiştir. Böylece sınıflama aşamasına geldiğinde özniteliklerin sayısı az fakat en önemlisi seçildi ve böyle FCNN modeli ile sınıflama yüksek doğruluk ve performans gösterildi.

Bu çalışmada çok zaman ve güç harç edildi ve bu başarı sonuçlara ulaştıkça yeni bilgiler öğrenildi ve çok hatalar ile karşılaştık. Burada aynı alanda çalışma yapacak araştırmacılar için birkaç önerilere bulunuyor.

Parmak hareketlerinin sınıflaması üzerinde çalışmak isteyenler, öncelikle veri setini dikkatlice araştırmalı ve en uygun veri setini seçmelidir. Veri seti seçildikten sonra, bu veri setini iyice anlamalı ve üzerinde yapılan önceki çalışmaları incelemelidir. Ayrıca, veri setinin toplanmasında hangi cihazların ve yöntemlerin kullanıldığını öğrenmek önemlidir. Bu adımlar çok kritik olup, yalnızca bu aşama bile zaman alıcı olabilir ve birkaç hafta sürebilir.

Sınıflama modelini seçmeden önce önemli özniteliklerin belirlenmesi, en kritik adımlardan biri olabilir. Özniteliklerin çıkarılması için kullanılan teknikleri ve modelleri aramak ve denemek, zaman alıcı ve yorucu bir süreçtir çünkü özniteliklerin belirlenmesi için pek çok farklı yöntem bulunmaktadır. Bu nedenle, bu tür araştırmalarda en yaygın kullanılan teknik ve yöntemleri öncelikli olarak denemek daha verimli olacaktır. Böylece, hızlı bir şekilde uygun öznitelik çıkarımı ve seçimi yapılabilir.Uygun sınıflama modelini arama aşamasında, her denenen modelin sonuçları kaydedilmelidir. Çünkü sonuçlar ne kadar düşük olursa olsun, bu veriler

önemli olabilir; bazen modelin başarısızlığı, modelden değil, veri setinden ya da öznitelik çıkarım yöntemlerinden kaynaklanabilir. Ayrıca, modeli kullanırken sonuçları iyileştirebilecek ek yöntemler uygulanmalıdır.

KAYNAKLAR

[1] M. M. Taye, "Understanding of Machine Learning with Deep Learning: Architectures, Workflow, Applications and Future Directions," 2023. doi: 10.3390/computers12050091.

[2] A. Ibrahim, V. R. Gannapathy, L. W. Chong, and I. S. M. Isa, "Analysis of electromyography (EMG) signal for human arm muscle: A review," in *Lecture Notes in Electrical Engineering*, 2016. doi: 10.1007/978-3-319-24584-3_49.

[3] M. Benzyane, M. Azrour, I. Zeroual, and S. Agoujil, "Exploring the Impact of Convolutions on LSTM Networks for Video Classification," in *Lecture Notes in Networks and Systems*, 2024. doi: 10.1007/978-3-031-48573-2_4.

[4] D. Xiong, D. Zhang, X. Zhao, and Y. Zhao, "Deep Learning for EMG-based Human-Machine Interaction: A Review," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 3, 2021, doi: 10.1109/JAS.2021.1003865.

[5] C. J. Swinney and J. C. Woods, "GPS Jamming Signal Classification with CNN Feature Extraction in low Signal-to-Noise Environments," *International Journal on Cyber Situational Awareness*, vol. 6, no. 1, 2022, doi: 10.22619/ijcsa.2021.100135.

[6] M. Atzori, M. Cognolato, and H. Müller, "Deep learning with convolutional neural networks: a resource for the control of robotic prosthetic hands via electromyography," *Front Neurorobot*, vol. 10, 2016.

[7] T. Triwiyanto, V. Abdullayev, and A. A. Ahmed, "Deep Convolution Neural Network to Improve Hand Motion Classification Performance Against Varying Orientation Using Electromyography Signal," *International Journal of Precision Engineering and Manufacturing*, vol. 25, no. 6, 2024, doi: 10.1007/s12541-024-00985-x.

[8] R. B. Azhiri, M. Esmaeili, and M. Nourani, "Real-Time EMG Signal Classification via Recurrent Neural Networks," in *Proceedings- 2021 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2021*, 2021. doi: 10.1109/BIBM52615.2021.9669872.

[9] K. H. Lee, J. Y. Min, and S. Byun, "Electromyogram-based classification of hand and finger gestures using artificial neural networks," *Sensors*, vol. 22, no. 1, 2022, doi: 10.3390/s22010225.

[10] S. K. Dwivedi, J. Ngeo, and T. Shibata, "Extraction of Nonlinear Synergies for Proportional and Simultaneous Estimation of Finger Kinematics," *IEEE Trans Biomed Eng*, vol. 67, no. 9, 2020, doi: 10.1109/TBME.2020.2967154.

[11] Ari, A. (2023). EMG signal classification using deep learning and time domain descriptors-based feature extraction for hand grip movement recognition. Traitement du Signal, Vol. 40, No. 3, pp. 949-960. https://doi.org/10.18280/ts.400311

[12] V. Medved, S. Medved, and I. Kovač, "Critical Appraisal of Surface Electromyography (sEMG) as a Taught Subject and Clinical Tool in Medicine and Kinesiology," 2020. doi: 10.3389/fneur.2020.560363.

[13] M. Shimura, A. Mizumoto, Y. Xia, and Y. Shimomura, "Multipoint surface electromyography measurement using bull's-eye electrodes for wide-area topographic analysis," *J Physiol Anthropol*, vol. 42, no. 1, 2023, doi: 10.1186/s40101-023-00342-3.

[14] A. Tharwat, "Independent component analysis: An introduction," 2018. doi: 10.1016/j.aci.2018.08.006.

[15] M. Chen and P. Zhou, "A novel framework based on FastICA for high density surface EMG decomposition," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 24, no. 1, 2016, doi: 10.1109/TNSRE.2015.2412038.

[16] R. N. Khushaba, A. Al-Ani, A. Al-Timemy, and A. Al-Jumaily, “A fusion of time-domain descriptors for improved myoelectric hand control,” in *2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016*, 2017. doi: 10.1109/SSCI.2016.7850064.

[17] W. Souai, "The secret to mastering feature extraction in Convolutional Neural Network," *Medium*, 2024. [Online]. Available: https://medium.com/ubiai-nlp/the-secret-to-mastering-feature-extraction-in-convolutional-neural-network-785ddedfb962. [Accessed: Nov. 17, 2024].

[18] X. Zhao, L. Wang, Y. Zhang, X. Han, M. Deveci, and M. Parmar, “A review of convolutional neural networks in computer vision,” *Artif Intell Rev*, vol. 57, no. 4, 2024, doi: 10.1007/s10462-024-10721-6.

[19] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, “Convolutional neural networks: an overview and application in radiology,” 2018. doi: 10.1007/s13244-018-0639-9.

[20] H. Sak, A. Senior, and F. Beaufays, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling,” in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2014. doi: 10.21437/interspeech.2014-80.

[21] K. L. says, “What is LSTM- Introduction to Long Short Term Memory,” *Intellipaat Blog*, 2020.

[22] P. Mahajan, "Fully connected vs Convolutional Neural Networks," *Medium*. [Online]. Available: https://medium.com/swlh/fully-connected-vs-convolutional-neural-networks-813ca7bc6ee5. [Accessed: Nov. 17, 2024].