

**ANKARA ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**



BLM 4062 PROJE RAPORU

Birkaç Örnek Öğrenme ile El İmzası Tanıma

Erdi Küçük

18290039

Dr.Öğr.Üyesi Özgür TANRIÖVER

MAYIS 2022

ÖZET

El yazısı imzalar, bilim camiasında tartışmanın merkezinde yer alan biyometrik özelliklerdir. Bunun sebebi, imzanın; bir kişinin kimliğini doğrulamak için her yerde kullanımı nedeniyle, önemli bir biyometrik özellik türü olmasından kaynaklıdır.

El yazısı imza doğrulaması ise, verilen bir imzanın sahte veya orijinal olup olmadığını tespit etmeyi amaçlayan önemli bir biyometrik tekniktir. Yasal, mali ve idari alanlarda, banka kontrolü ödemesi gibi birçok ticari senaryoda, sahteciliğe karşı koyabilmek adına imza doğrulama işlemi bilinen tek bir örneğin insan incelemesine dayanmaktadır. Ancak son yıllarda otomatik el imzası doğrulama sistemleri geliştirilmiştir.

İmza doğrulama sistemlerinde, diğer doğrulama problemlerinden farklı olarak, gerçek ve sahte imzalar arasındaki küçük ama kritik detayları modellemesi gerekir, çünkü yetenekli bir taklitçi gerçek bir imzadan yalnızca bazı belirli deformasyon türleriyle farklılık gösterebilir. Bu sebeple çevrimdışı imza doğrulama, biyometri ve adli belge alanındaki en zorlu görevlerden biridir.

Bu raporda; imza doğrulama sistemlerinin tarihçesi, açık kaynakları veri setleri, veri işleme adımları ve derin sinir ağları hakkında bilgiler verilmiştir. Çalışmada evrişimli bir Siyam ağı çevrimdışı imza doğrulama sistemini modellenmiştir. Eğitim ve test sırasında ICDAR 2011 SigComp veri seti kullanılmıştır. Geliştirmeler Python programlama dili üzerinde yapılmış olup, sinir ağı ise açık kaynaklı PyTorch kütüphanesi kullanılarak geliştirilmiştir.

İÇİNDEKİLER

1. GİRİŞ	1
2. VERİ SETLERİ	4
3. VERİ ÖN İŞLEME	6
3.1- Gri Renkli Görüntüye Çevirme	6
3.2 - Threshold Uygulama	7
3.3 - Alan Analizi	8
3.4 - Siyah ve Beyaz Piksellere Boyama	9
4. VERİLERİ DOSYADAN OKUMA	11
5. SİNİR AĞI MODELİ	13
5.1 – Meta Öğrenme	13
5.1.1 - Sıfır-atışlı (zero-shot) öğrenme	14
5.1.2 - Tek-adımlı (one-shot) ve birkaç adımlı (few-shot) öğrenme	14
5.1.3 – Eğitim Seti ve Destek Seti	16
5.1.4 – Prototipik Ağlar	16
6. ARAYÜZ TASARIMI	18
6.1 - Giriş Ekranı	18
6.2 - Ayarlar Ekranı	19
6.3 - Kullanıcı Yönetimi	19
6.4 - Ana Ekran ve Sonuçlar	20
7. PERFORMANS DEĞERLENDİRMELERİ	22
8. SONUÇ	23
9. KAYNAKÇA	24

1. GİRİŞ

Biyometri teknolojisi, çok çeşitli güvenlik uygulamalarında kullanılmaktadır. Bu tür sistemlerin amacı, bir kişiyi fizyolojik veya davranışsal özelliklere dayalı olarak tanımdır. İlk durumda, tanıma parmak izi, yüz, iris vb. gibi biyolojik özelliklerin ölçümlerine dayanır. Sonraki durumda ses ve el yazısı imza gibi davranışsal özelliklerle ilgilidir. Biyometrik sistemler esas olarak iki senaryoda kullanılır: doğrulama ve tanımlama. İlk durumda, sistemin bir kullanıcısı bir kimlik talep eder ve biyometrik numuneyi sağlar. Doğrulama sisteminin rolü, kullanıcının gerçekten iddia ettiği kişi olup olmadığını kontrol etmektir. Tanımlama durumunda, bir kullanıcı biyometrik bir örnek sağlar ve amaç, sisteme kayıtlı tüm kullanıcılar arasında onu tanımlamaktır. El yazısı imza, özellikle yasal, mali ve idari alanlarda bir kişinin kimliğini doğrulamak için her yerde kullanımı nedeniyle, biyometrik özelliğin özellikle önemli bir türüdür. Yaygın kullanımının nedenlerinden biri, el yazısı imza toplama sürecinin müdahale gerektirmemesi ve insanların günlük yaşamlarında imza kullanımına aşina olmalarıdır.

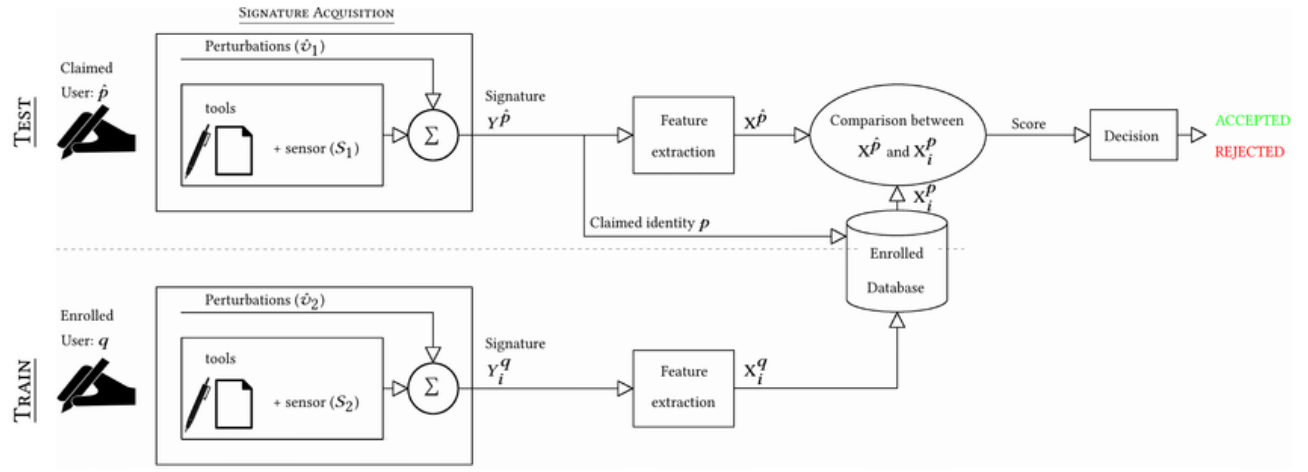
İmza doğrulama sistemleri, biyometrik örneğin gerçekten iddia edilen bir bireye ait olup olmadığını otomatik olarak ayırt etmeyi amaçlar. Başka bir deyişle, sorgu imzalarını gerçek veya sahte olarak sınıflandırmak için kullanılırlar. Sahtecilik genellikle üç türde sınıflandırılır:

Rastgele Sahtecilik: Bu, belirli bir imzanın daha önce bilgi sahibi olmadan, bir sahibinin kimliğini doğrulamaya çalıştığı durumdur. Taklitçi, kullanıcı veya imzası hakkında hiçbir bilgiye sahip değildir ve bunun yerine kendi hakiki imzasını kullanır. Bu durumda, sahtecilik, kullanıcıdan gelen orijinal imzalardan farklı bir anlamsal anlam içerir ve çok farklı bir genel şekil sunar.

Basit Sahtecilik: Bazı ülkelerde, birçok kişi sadece kendi adını ve soyadını yazarak imza atarlar, buna karşılık diğer ülkelerde ise imzacılar karmaşık süslemeler veya değerlendirme listeleri içeren okunaksız imzalar tasarlar. Basit sahtecilik söz konusu olduğunda, taklitçi ad ve soyad kullanılarak tasarlanmış bir imzayı taklit etmeye çalışır. Taklitçi kullanıcının adı ve soyadı hakkında bilgi sahibidir, ancak kullanıcının imzası hakkında bilgi sahibi değildir ve kullanıcının imzasına erişim imkanı da yoktur.

Bu durumda, özellikle tam adlarıyla veya imzanın bir kısmıyla imza atan kullanıcılar için sahtecilik, gerçek imzaya daha fazla benzerlik gösterebilir.

Yetenekli Sahtecilik: Bu durumda taklitçi, kullanıcının hem adına soyadına hem de imzasına erişim sağlamış durumdadır ve genellikle kullanıcının imzasını taklit etme uygulamaları yapar. Bu durum, gerçek imzaya daha fazla benzeyen ve bu nedenle tespit edilmesi daha zor olan sahteciliklerle sonuçlanır. Bu tür sahtecilik için yapılan test, imza doğrulamasıyla en alakalı olanıdır.



Şekil 1.1 – İmza doğrulama sistemlerinin çalışma algoritmasına yönelik bir şema.

Tipik bir imza doğrulama sisteminin çalışma şeması Şekil 1.1'de gösterilmektedir. Bir kişi bir imza doğrulama sistemine giriş yaptığında, belirli bir araç yardımıyla (örneğin, kalem ve kağıt) imzasını atar. Bu imzanın, imza doğrulama sistemine sokulabilmesi için sayısallaştırılması gerekir ve bu işlem çeşitli sensörler tarafından gerçekleştirilir. Örneğin bir tarayıcı, bir dijital tablet, bir cep telefonu veya bir stylus kalem.

Sayısallaştırma yöntemine bağlı olarak, imza doğrulama sistemleri iki kategoriye ayrılır:

Çevrimiçi (dinamik) imza toplama yöntemleri: Bu yöntemde kullanıcının imzasını kaydetmek için, sayısallaştırıcı tablet gibi bir cihaz gereklidir. Veriler, zaman içinde,

kalemin konumunu içeren ve bazı durumlarda kalem eğimi, basıncı gibi ek bilgileri içeren bir dizi halinde toplanır. Örneğin bir stylus kalem ile dijital bir cihaz üzerinde imza atılması.

Çevrimdışı (statik) imza toplama yöntemleri: Çevrimdışı imza doğrulamasında, imza, yazma işlemi tamamlandıktan sonra alınır. Bu durumda imza dijital bir görüntü olarak temsil edilir. Örneğin imzanın bir kalem ile kağıt üzerine atıldıktan sonra tarayıcı yardımıyla sayısallaştırılması.

Tüm imza doğrulama sistemleri, Şekil 1.1'de gösterilen genel şemaya göre çalışsa da, özelliklerin çıkarılması ve son sınıflandırmayı üretme stratejisi, çevrimiçi ve çevrimdışı sistemler arasında genellikle farklıdır. Bununla birlikte, bir imza doğrulama sistemini test etmek için her iki durumda da anlamlı bir istatistiksel imza veri seti gerekir.

2. VERİ SETLERİ

El yazısı imzalar, karmaşık bir insan sürecinin ürünüdür. Bu nedenle, pratik boyutta tek bir imza veritabanında temsil edilemeyen sayısız değişken ve parametre vardır. İmza doğrulama sistemleri tipik olarak belirli bir imza veritabanında bulunan imzalardaki özellikleri modeller ve bu onların uygulanabilirliğini sınırlar. Bu etki, veri kümelerinde tüm insan davranışları dikkate alınmadığından muhtemelen yanlış sonuçlara yol açmaktadır.

Özel veri kümeleriyle otomatik imza doğrulama konusunda büyük miktarda araştırma yapılmıştır. Bu, sınıflandırma performansındaki bir gelişme daha iyi bir yöntem veya basitçe daha temiz veya daha basit bir veri setine atfedilebileceğinden, ilgili çalışmaları karşılaştırmayı zorlaştırır. Bununla birlikte, son on yılda, araştırma topluluğu için bu boşluğu ele alan birkaç imza veri seti açık kaynaklı hale getirilmiştir. İmza görüntülerini alma işlemi, genel veri kümelerinin çoğu için benzer adımları takip eder. Orijinal imzalar bir veya daha fazla oturumda toplanır ve kullanıcının imzalarından birkaç örnek vermesini gerektirir. Kullanıcı, birçok hücre içeren bir form alır ve her hücrede imzasının bir örneğini sağlar. Hücreler genellikle banka çekleri ve kredi kartı fişleri gibi yaygın senaryolarla eşleşecek boyutlara sahiptir. Sahtelerin toplanması farklı bir süreç izler: kullanıcılar gerçek imzalardan örnekler alır ve imzayı bir veya daha fazla kez taklit etmeleri istenir. Sahteciliği sağlayan kullanıcıların sahtecilik konusunda uzman olmadıklarını belirtmekte fayda vardır. Formlar toplandıktan sonra taranır ve ön işleme tabi tutulur.

Bu çalışmada imza doğrulama algoritmasının değerlendirilmesi için, yaygın olarak kullanılan veri setlerinden biri olan ICDAR 2011 SigComp veri seti kullanılmıştır.

2.1 ICDAR 2011 SigComp

ICDAR 2011 SigComp veri seti 69 farklı kullanıcıya ait imzaları içerir. Her kullanıcıya ait 20'nin üzerinde orjinal ve sahte imza örneği yer almaktadır. Veri setinin toplam büyüklüğü 1649 görüntüdür.



Şekil 2.1.1 – ICDAR 2011 SigComp veri setine ait orjial (sol) ve sahte (sağ) imza örneği

ICDAR 2011 SigComp veri setine [bu adresten](#) ulaşılabilir.

	İmzalayan Sayısı	Gerçek İmza	Sahte İmza	Kayıt Formatı
ICDAR 2011	69	12-24	8 -16	.png

Tablo 2.1.1 – Veri setlerine ait bilgiler

3. VERİ ÖN İŞLEME

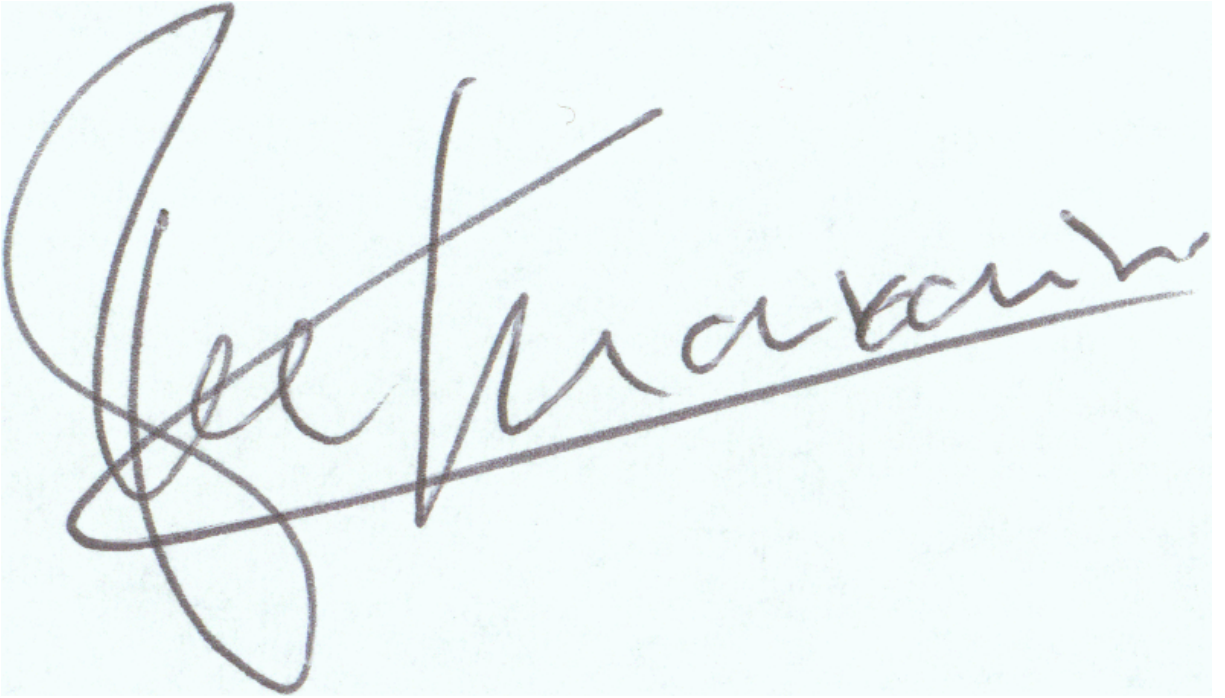
Kullanılan yapay sinir ağının daha başarılı sonuçlar verebilmesi için verinin işlenmesi ve aykırılıkların giderilmesi gerekmektedir.

Bunun için de veri setimizdeki görüntüler üzerinde işlemler yapılarak modele verilecek nihai görüntü elde edilmiştir.

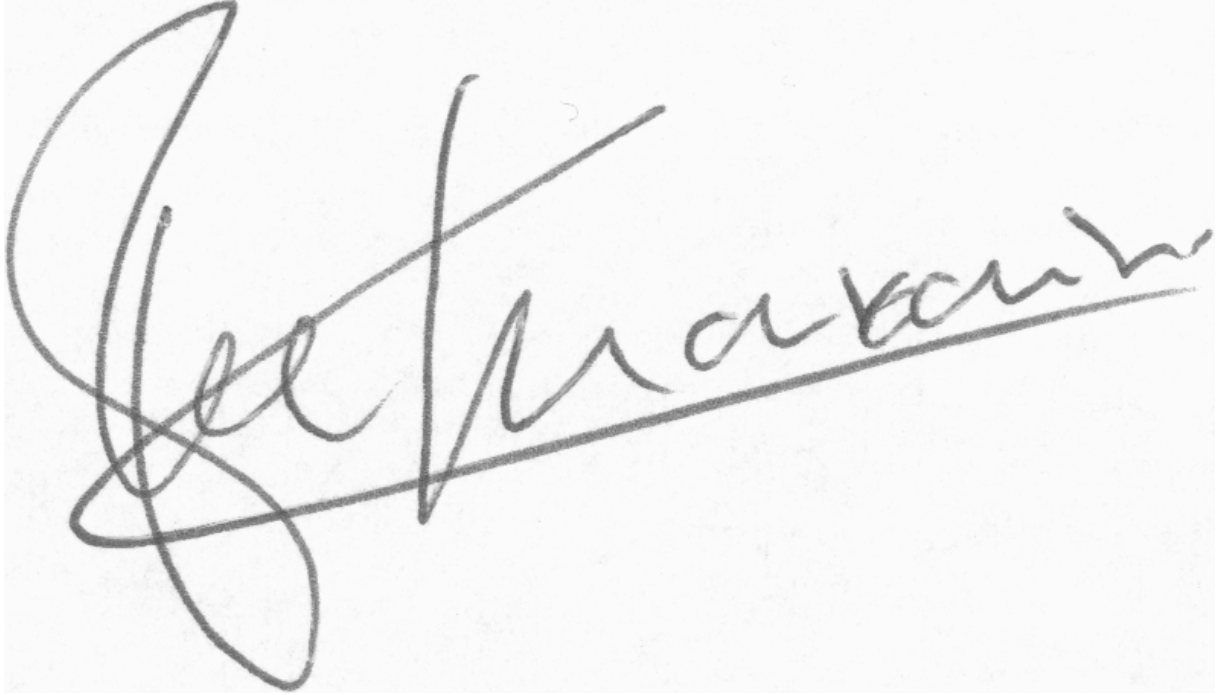
Aşağıda bu işlemler adımlar halinde gösterilmiştir.

3.1 - Gri Renkli Görüntüye Çevirme

Modelin sınıflandırma yaparken daha kesin sonuçlar verebilmesi için bütün görüntüler sadece siyah ve beyaz pikseller içerecek şekilde ayarlanmalıdır, Bunun ilk adımı görüntüleri gri görüntüye çevirmektir. Şekil 3.1.1 ve Şekil 3.1.2’de ham görüntü ve gri renkli görüntü gösterilmiştir.



Şekil 3.1.1 – Ham (işlenmemiş) görüntü



Şekil 3.1.2 – Gri renkli görüntü

3.2 - Threshold Uygulama

Gri renkli görüntü üzerinde siyah ve beyaz piksel ayrımı yapabilmek için bir threshold uygulanmıştır. Threshold değeri olarak 210 değeri seçilmiştir. Bunun anlamı renk değeri 210 altında kalan pikseller siyah bölge üstünde kalan pikseller beyaz bölge olarak seçilmiştir.



Şekil 3.2.1 – Threshold işlemi sonrası seçilmiş bölgeler

3.3 - Alan Analizi

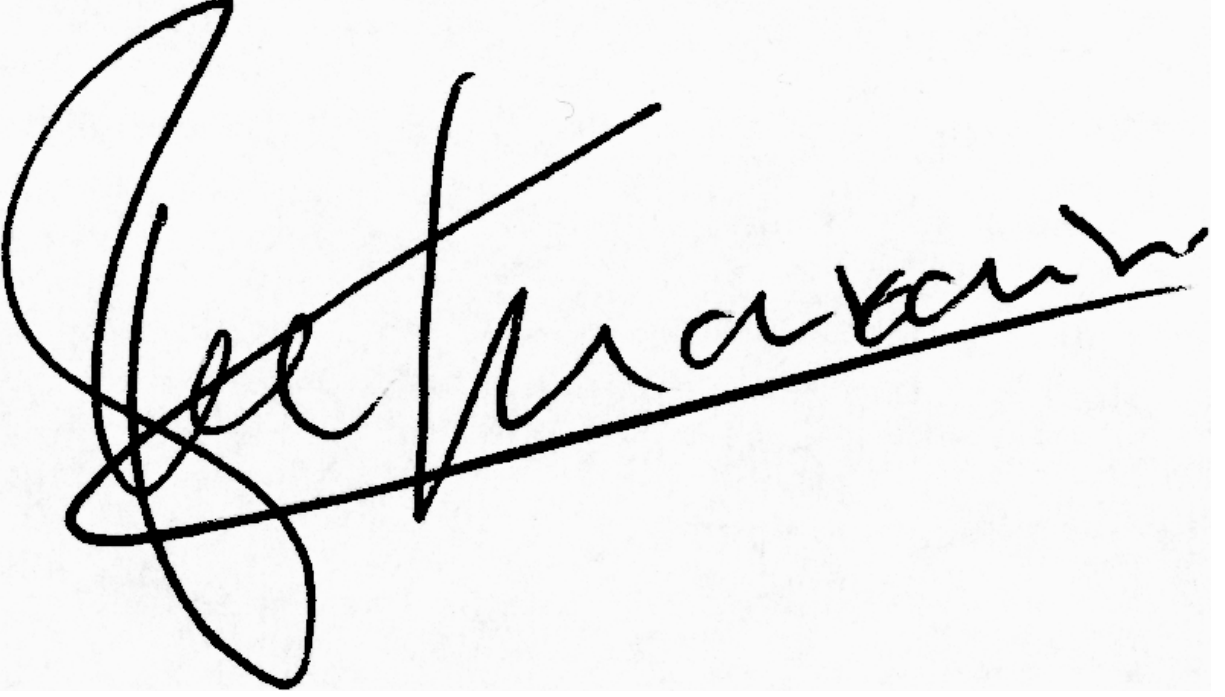
Threshold işlemi bize sonuç olarak farklı alan büyüklüğündeki bölgeler (region) döner. Şekil 3.1'deki imza için 15 adet bölge elde edilmiştir. Ancak bu bölgelerin hepsinin imzaya dahil bir bölge olarak seçilmesi bazı yanlış sonuçlara sebep olabilir. Örneğin kullanıcının imza atarken kağıt üzerinde bıraktığı çok ufak ama imzaya dahil olmayan mürekkep izleri veya kağıt üzerindeki çok ufak boyuttaki kirler modelin doğru sonuç vermesini zorlaştırabilir. Bu durumun önüne geçebilmek adına bölgeler üzerinde alan analizi yapılmıştır. Alanı $5px^2$ den büyük bölgeler seçilmiştir. Kalan bölgeler kullanılmıştır. Bu işlem sonucu bölge sayısı 15'den 6'ya düşmüştür.



Şekil 3.3.1 – Alan analizi sonucu elde edilen görüntü

3.4 - Siyah ve Beyaz Piksellere Boyama

Son adım olarak alan analizi sonucu elde edilen görüntüde threshold altında kalan pikseller siyah renge, üstünde kalan bölgeler beyaz renge boyanır. Bu işlem sonucu modele verilecek nihai görüntü elde edilir.



Şekil 3.4.1 – Piksel boyama işlemi ve elde edilen nihai görüntü

4. VERİLERİ DOSYADAN OKUMA

ICDAR 2011 veri seti 69 farklı kullanıcıya ait 1649 adet imza içerir. Bu imzaların bir kısmı eğitim bir kısmı ise test verisi olarak ayrılmıştır. Eğitim verisi 1149 adet, test verisi ise 500 adet imza içerir.

Bu imzaların modelin anlayabileceği ve işleyebileceği bir format olan tensörlere dönüştürülmesi gerekmektedir.

Şekil 4.1 ve Şekil 4.2’de görüntülerin bulunduğu klasörlere ulaşan ve görüntüleri okuyup numpy dizilerine dönüştüren Python fonksiyonları yer almaktadır.

```
41 def read_images(base_directory):
42     """
43     Reads all the images from the base_directory
44     Uses multithreading to decrease the reading time drastically
45     """
46     datax = None
47     datay = None
48     pool = Pool(mp.cpu_count())
49     results = [pool.apply(read_classes,
50                          args=(
51                              base_directory + '/' + directory + '/', directory,
52                              )) for directory in os.listdir(base_directory)]
53     pool.close()
54     for result in results:
55         if datax is None:
56             datax = result[0]
57             datay = result[1]
58         else:
59             datax = np.vstack([datax, result[0]])
60             datay = np.concatenate([datay, result[1]])
61     return datax, datay
```

Şekil 4.1 – Görüntülerin bulunduğu klasörlere erişen Python fonksiyonu

‘read_images’ fonksiyonu parametre olarak dosya yolunu alır. Bu parametreyi kullanarak her bir sınıf klasörünün dosya yolu liste halinde elde edilir ve bu dosya yolları ‘read_classes’ fonksiyonuna gönderilir. Bu işlem yapılırken çoklu işlem parçacığı (multithreading) kullanılır. Bunun nedeni eş zamanlı çalışmayı sağlayarak görüntülerin okunma süresinin azaltılmak istenmesidir.

Fonksiyonun dönüş değerleri her bir görüntünün ve sınıf isimlerinin yer aldığı numpy dizileridir.

```
16 def read_classes(class_directory_path, class_directory_name):
17     """
18     Reads all the signatures from a given class_directory
19     """
20     datax = []
21     datay = []
22
23     images = os.listdir(class_directory_path)
24     for img in images:
25         # print(alphabet_directory_path + img)
26         image = cv2.resize(cv2.imread(class_directory_path + img), (105, 105))
27
28         # rotations of image
29         rotated_90 = ndimage.rotate(image, 90)
30         rotated_180 = ndimage.rotate(image, 180)
31         rotated_270 = ndimage.rotate(image, 270)
32         datax.extend((image, rotated_90, rotated_180, rotated_270))
33         datay.extend((
34             class_directory_name + '_0',
35             class_directory_name + '_90',
36             class_directory_name + '_180',
37             class_directory_name + '_270'))
38
39     return np.array(datax), np.array(datay)
```

Şekil 4.2 – Görüntüleri numpy dizisine çeviren ‘read_classes’ fonksiyonu

‘read_classes’ fonksiyonu her bir görüntüyü okur, istenilen yükseklik ve genişlik değerlerine yeniden boyutlandırır.

Eğitim veri setinin oldukça az sayıda imza içermesi modelin başarısını olumsuz etkileyebilir. Buna çözüm olarak her bir imza 90, 180 ve 270 derece döndürülerek yeni görüntüler elde edilmiştir. Bu sayede toplam görüntü sayısı 1149’dan 4596’ya yükselmiştir.

Döndürülmüş görüntüler ve bunlara ait sınıf isimleri numpy dizisi olarak return edilir.

5. SİNİR AĞI MODELİ

İlk evrişimli sinir ağı (CNN) algoritmaları oluşturulduğundan beri, bilgisayarlı görme (computer vision) görevlerinde derin öğrenme performansını büyük ölçüde iyileştirdiler.

Günümüzde, belirli bir görevi çözmek için milyarlarca görüntünün kullanılması söz konusu olduğunda bilgisayarlar oldukça başarılıdır. Yine de, gerçek dünyada, bu kadar çok örnek içeren bir veri seti oluşturmak veya bulmak nadir bir durumdur.

Peki bu kadar az veriyle başarılı bir model geliştirme problemini nasıl aşarız? Bir bilgisayarlı görü görevinden bahsediyorsak, veri artırmayı (data augmentation) kullanabilir veya ek verileri toplayıp etiketleyebiliriz.

Veri artırma güçlü bir araçtır ve sorunu büyük ölçüde çözebilir. Ek verileri toplamak ve etiketlemek zaman alan ve maliyeti yüksek bir iştir, ancak yine de iyi sonuçlar verir.

Eğer veri kümesi gerçekten küçükse, veri artırma tekniği gerçekten başarılı olamayabilir. Sınıf başına yalnızca bir veya iki örnekle bir sınıflandırma oluşturmamız gereken bir problem düşünüldüğünde yenilikçi yaklaşımlara ihtiyaç duyulur. Meta öğrenme metodu (meta learning) bunlardan biridir.

5.1 – Meta Öğrenme

Meta öğrenme, makine öğreniminin alt dallarından biridir. Yalnızca birkaç eğitim örneğiniz olduğunda etiketlenmemiş yeni verileri sınıflandırmakla ilgilidir.

Meta öğrenme, daha fazla araştırma ve iyileştirme gerektiren oldukça genç bir alandır. Günümüz itibarıyla bilgisayarlı görü problemlerinde kullanılabilir. Bu bilgisayarlı görme modeli, nispeten az sayıda eğitim örneğiyle oldukça iyi çalışabilir.

Meta öğrenme metodu varyasyonları sıfır atışlı öğrenme ve birkaç adımlı öğrenmeyi içerir.

5.1.1 - Sıfır-atışlı (zero-shot) öğrenme

Sıfır atışlı öğrenmenin amacı, görünmeyen sınıfları herhangi bir eğitim örneği olmadan sınıflandırmaktır.

Nesneleri daha önce görülmemiş sınıflardan doğru bir şekilde kategorize etmek, gerçek anlamda herhangi bir nesne tespit sisteminde önemli bir gereksinimdir.

Sıfır atışlı öğrenme, yeni bir örnek kategorisi üzerinde genelleme yapmak için önceden eğitilmiş bir derin öğrenme modelinin yapıldığı, yani eğitim ve test seti sınıflarının ayırık olduğu bir makine öğrenimi paradigmasıdır.

5.1.2 - Tek-adımlı (one-shot) ve birkaç adımlı (few-shot) öğrenme

Birkaç adımlı öğrenme, sınırlı sayıda örneğe dayalı tahminler yapma sorunudur. Birkaç adımlı öğrenme, standart denetimli öğrenmeden farklıdır. Birkaç adımlı öğrenmenin amacı, modelin eğitim setindeki görüntüleri tanımasına ve ardından test setine genelleme yapmasına izin vermemektir. Bunun yerine amaç öğrenmektir. “Öğrenmeyi öğrenmek” anlaşılması zor bir kavram olarak gelebilir. Bunu şu şekilde düşünebiliriz:

Model birden fazla sınıfa ait verileri içeren bir eğitim setinde eğitilir. Eğitimin amacı test verilerinin hangi sınıfa ait olduğunu bilmek değildir. Bunun yerine amaç, nesneler arasındaki benzerliği ve farkı bilmektir.

Eğitimden sonra iki görüntüyü modele gösterip ikisinin aynı sınıf olup olmadığını sorabilirsiniz. Model, nesneler arasında benzerlikler ve farklılıklara sahiptir. Böylece model, iki görüntüdeki içeriklerin aynı tür nesneler olduğunu söyleyebilir.

a) Birkaç Adımlı Öğrenme Terminolojileri

k-yol (k-way), destek kümesinin k sınıfı olduğu anlamına gelir.

n-adım (n-shot), her sınıfın n örneği olduğu anlamına gelir.

Destek seti k-yol ve n-adım olarak adlandırılır.

b) Birkaç adımda öğrenmenin tahmin doğruluğu

Birkaç vuruşlu öğrenme gerçekleştirirken, tahmin doğruluğu, yol sayısına ve atış sayısına bağlıdır.

Yol sayısı arttıkça tahmin doğruluğu düşer.

Bu durumun sebebi şu şekilde açıklanabilir: Daha önce görmediğiniz bir imzayı size verilen 3 farklı kullanıcıya ait 3 adet imzayı kullanarak sınıflandırmaya çalışın. Bu 3 yönlü 1 atışlı öğrenmedir. Peki ya 6 kullanıcı olursa? Bu 6 yönlü 1 atışlı öğrenme olacaktır. Verilen imzayı sınıflandırırken 3-yol 6-yoldan daha kolaydır. 3'ten birini seçmek, 6'dan birini seçmekten daha kolaydır.

Bu nedenle, 3-yollu öğrenme, 6-yolludan daha yüksek doğruluğa sahiptir.

Çekim sayısı arttıkça tahmin doğruluğu da artar. Daha fazla örnekle tahmin daha kolay hale gelir. Bu nedenle, 2 atış, 1 atıştan daha kolaydır.

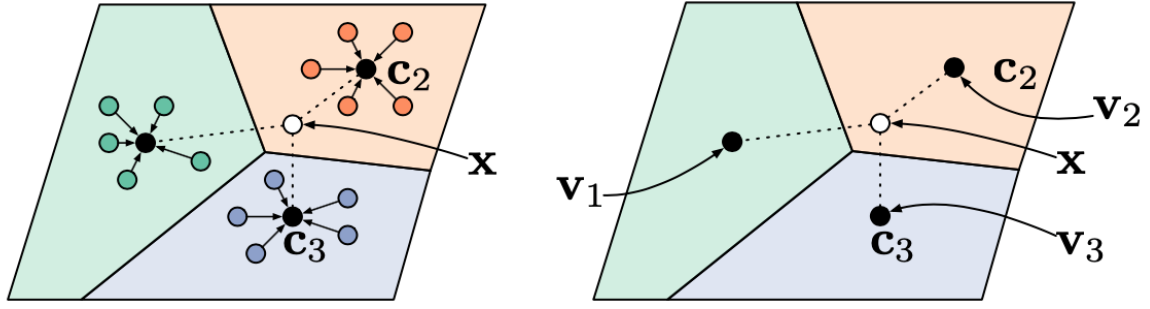
b) Birkaç adımlı öğrenmenin ardındaki temel fikir

Birkaç adımlı öğrenmenin temel fikri, benzerliği tahmin eden bir işlevi eğitmektir.

Benzerlik fonksiyonunu $\text{sim}(x, x')$ ile gösteriniz. İki örnek, x ve x' arasındaki benzerliği ölçer. İki örnek tamamen aynıysa, benzerlik işlevi 1 döndürür, yani $\text{sim}(x, x') = 1$

Örnekler oldukça farklıysa 0'a yakın bir değer döndürülür. yani $\text{sim}(x, x') \sim 0$

Eğitimden sonra, öğrenilen benzerlik işlevi, görünmeyen sorgular için tahminler yapmak için kullanılabilir. Destek kümesindeki her örnekle sorguyu karşılaştırmak ve benzerlik puanlarını hesaplamak için benzerlik fonksiyonunu kullanılır. Ardından, en yüksek benzerlik puanına sahip örnek tahmin sınıfı olarak isimlendirilir.



Şekil 5.1.2.1 – Birkaç atış (few-shot) ve sıfır atış (zero-shot) senaryolarında prototip ağlar. **Sol:** Az sayıda prototip c_k , her sınıf için destek seti örneklerinin ortalaması olarak hesaplanır. **Sağ:** Sıfır-atış prototipleri c_k , sınıf meta-veri v_k gömülerek üretilir. Her iki durumda da, gömülü sorgu noktaları, sınıf prototiplerine olan mesafeler üzerinden bir softmax fonksiyonu aracılığıyla sınıflandırılır:

5.1.3 – Eğitim Seti ve Destek Seti

Destek seti, meta öğrenmenin jargonudur. Küçük etiketli görüntü kümesine destek kümesi denir. Eğitim seti ile destek seti arasında farklar vardır. Eğitim seti daha büyüktür ve her sınıf için birden çok örnek içerir. Eğitim seti, derin bir sinir ağını öğrenmek için yeterince büyüktür. Buna karşılık, destek seti küçüktür. Her sınıfın en fazla birkaç örneği vardır. Eğitim setinde her sınıfın sadece bir örneği varsa, derin bir sinir ağını eğitmek imkansızdır. Destek seti, yalnızca test zamanında ek bilgi

sağlayabilir.

5.1.4 – Prototipik Ağlar

a) Notasyon

Birkaç adımlı sınıflandırmada bize N etiketli küçük bir destek seti verilir.

$$S = \{(x_1, y_1), \dots, (x_N, y_N)\}$$

Burada her x_i , D boyutlu özellik vektörünün (\mathbb{R}^D) elemanıdır.

$y_i \in \{1, \dots, K\}$ karşılık gelen etiket değeridir. S_k , k sınıfı ile etiketlenmiş örnekler kümesini belirtir.

b) Model

Prototipik Ağlar, öğrenilebilir parametrelerle (ϕ) bir gömme fonksiyonu $f_\phi: \mathbb{R}^D \rightarrow \mathbb{R}^M$ aracılığıyla her sınıfın M boyutlu bir temsiliyi hesaplar.

$$C_k = 1/|S_k| * \sum_{(x_i, y_i) \in S_k} f_\phi(x_i)$$

Verilen bir $d: \mathbb{R}^M * \mathbb{R}^M \rightarrow [0, +\infty)$ fonksiyonu ile, Prototipik Ağlar, gömme alanındaki prototiplere olan mesafeler üzerinden softmax fonksiyonuna dayalı olarak bir sorgu noktası x için sınıflar üzerinde bir dağılım üretir:

$$p_\phi(y = k|x) = \exp(-d(f_\phi(x), c_k)) / \sum_{k'} \exp(-d(f_\phi(x), c_{k'}))$$

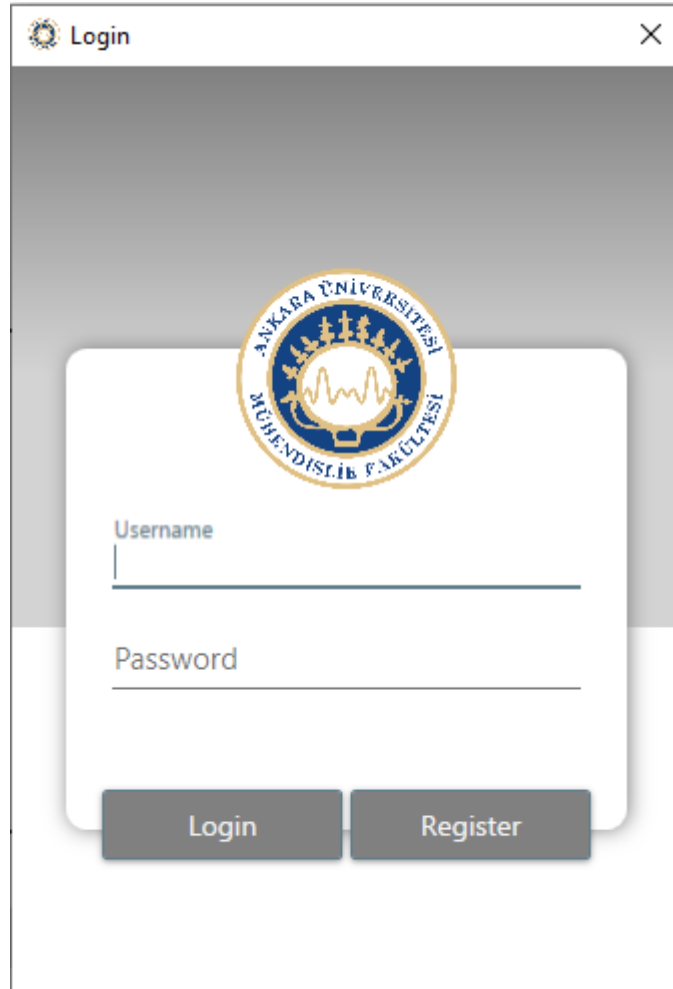
6. ARAYÜZ TASARIMI

Eğitilen modellerin kullanıma sunulabilmesi için bir arayüz tasarlanmıştır. Kullanıcılar arayüz üzerinden istedikleri imzaya ait destek ve sorgu seti verilerini yükleyerek sınıflandırma yapabilirler.

Arayüz tasarımında .NET Framework tabanlı WPF geliştirme platformu kullanılmıştır.

6.1 - Giriş Ekranı

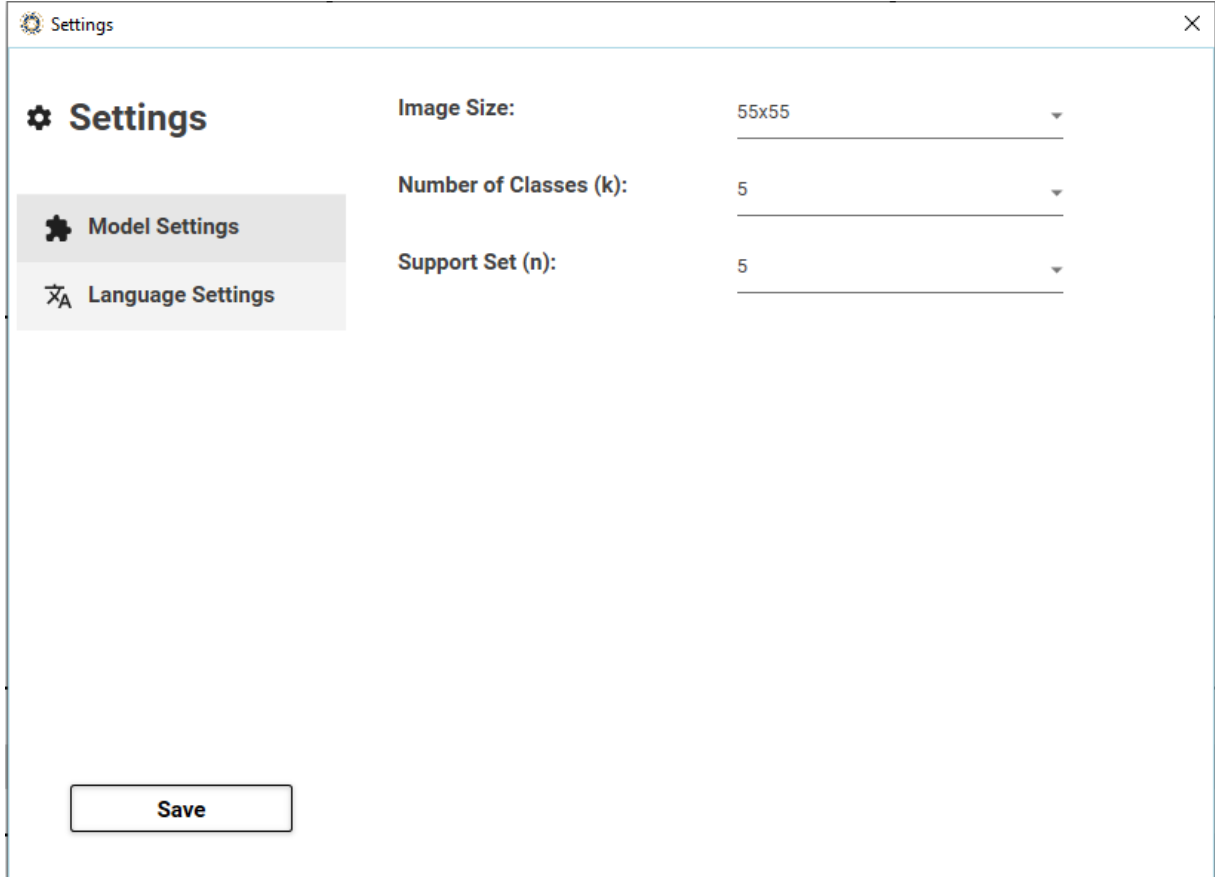
Uygulamayı kullanabilmek için giriş yapmak gerekmektedir. Ana ekranda giriş yap butonuna tıklandığında Şekil 6.1.1'deki giriş ekranı açılır. Kullanıcı adı ve şifre ile giriş yapıldıktan sonra uygulama kullanıma hazır hale gelir.



Şekil 6.1.1 – Giriş Ekranı

6.2 - Ayarlar Ekranı

Ayarlar ekranı üzerinden eğitilmiş modeller içerisinde istenilen seçilebilir ve seçilmiş model ile eğitim yapılabilir. Ayrıca 'Dil Ayarları' sekmesi üzerinden de uygulamanın dili değiştirilebilir. Uygulamada 'İngilizce' ve 'Türkçe' dil desteği bulunmaktadır.

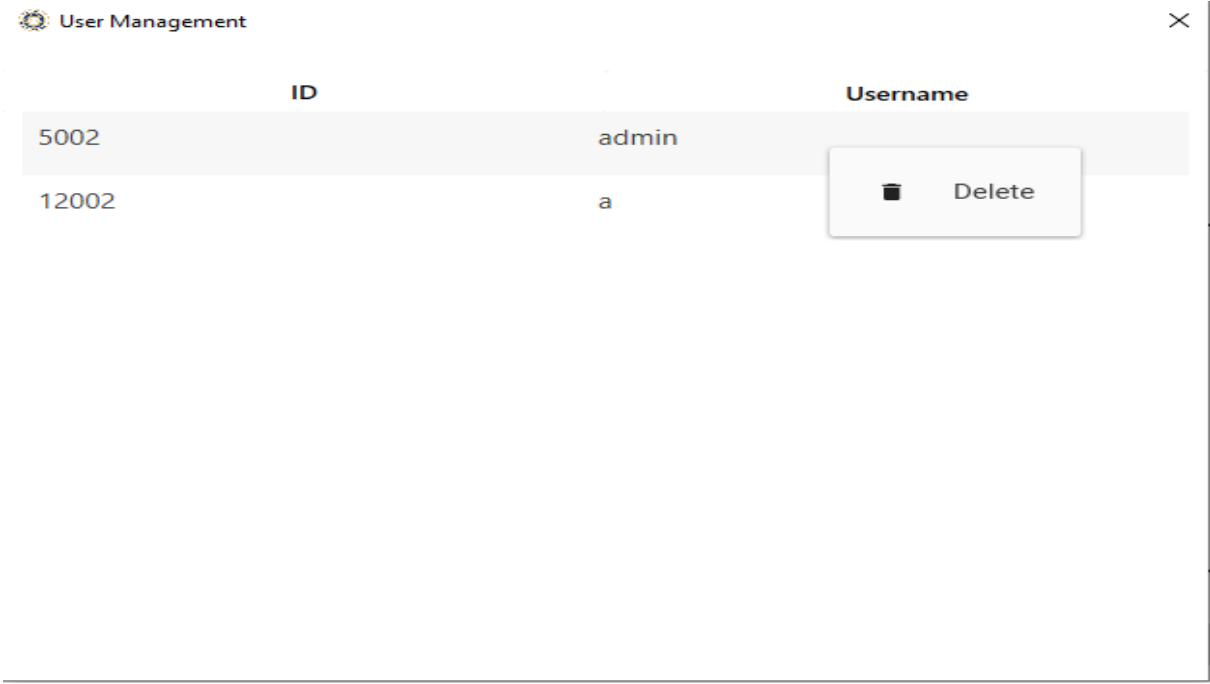


The screenshot shows a 'Settings' window with a title bar containing a gear icon and the word 'Settings'. Inside the window, there is a sidebar on the left with two tabs: 'Model Settings' (selected) and 'Language Settings'. The main area displays three settings: 'Image Size' set to '55x55', 'Number of Classes (k)' set to '5', and 'Support Set (n)' set to '5'. Each setting has a dropdown arrow. At the bottom left, there is a 'Save' button.

Şekil 6.2.1 – Ayarlar penceresi

6.3 - Kullanıcı Yönetimi

Uygulamaya kayıt olmuş kullanıcıların listesini görebilmek de mümkündür. Bunun için ana ekran üzerinden 'Kullanıcı Yönetimi' butonuna tıklanır. Açılan pencerede kullanıcıların ID'si ve kullanıcı adları gözükür. İstenilen kullanıcıya sağ tıklanarak kullanıcı silme işlemi de yapılabilir.



Şekil 6.3.1 – Kullanıcı yönetim ekranı

6.4 - Ana Ekran ve Sonuçlar

İmza sınıflandırması yapılabilmesi için destek seti verileri için 5 adet sınıftan sınıf başına 2 görüntü eklenmesi gerekir. Destek seti verileri eklendikten sonra sorgu seti verisi eklenir ve kontrol butonuna basılır. Uygulama arka planda bir Python kodu çalıştırır ve birkaç saniye içerisinde sonuçlar ekranda gösterilir. Sorgu seti verisinin her bir destek seti verisine olan öklid uzaklıkları hesaplanır. En düşük değer 'en az fark, en yüksek benzerlik' anlamına gelir ve tahmin edilen sınıf değeri olarak seçilir.



X

Destek Seti Kontrol Süresi = 3930,7075ms

050	054	062	050_forg	058

Sorgu Seti

050

Sonuçlar

Tahmin Edilen Sınıf	Tahmin Edilen Uzaklıklar
050	050_forg: 22.042343139648438 054: 63.84646224975586 050: 13.990625381469727 058: 63.35791778564453 062: 36.48693084716797

Şekil 6.4.1 – Ana ekran ve sonuçlar

7. PERFORMANS DEĞERLENDİRMELERİ

Model farklı görüntü formatlarında, farklı yol ve atış değerleriyle eğitilmiştir.

Eğitim süresi, eğitim sonucunda elde edilen doğruluk (accuracy) ve kayıp (loss) değerleri Tablo 7.1’de görülmektedir.

	105x105				55x55			
	20 yol 5 adım	20 yol 1 adım	5 yol 5 adım	5 yol 1 adım	20 yol 5 adım	20 yol 1 adım	5 yol 5 adım	5 yol 1 adım
Doğruluk	0.9983	0.9905	0.9868	0.9885	0.9853	0.9826	0.986	0.9849
Kayıp	0.0622	0.0314	0.0486	0.0364	0.0652	0.0572	0.0579	0.0491
Eğitim Süresi	36'13'	19'5"	9'24"	5'20"	12'41"	6'43"	4'28"	2'35"

Tablo 7.1 – Farklı parametrelerle eğitim ve sonuçlar

8. SONUÇ

El yazısı imza, bir kişinin kimliğini doğrulamak için her yerde kullanımı nedeniyle, önemli bir biyometrik veri türüdür. Bu özelliği sebebiyle dolandırıcılık ve sahteciliklere açıktır.

Bu duruma karşı olarak , son on yılda, araştırmacılar çevrimdışı imza doğrulaması için çok çeşitli yöntemler önerdiler. Orijinal imzaları ve yetenekli taklitçileri ayırt etmek zorlu bir görev olmaya devam ederken, son birkaç yılda, derin öğrenme alanındaki gelişmeler nedeniyle hata oranları önemli ölçüde düştü ve bu alandaki çalışmalara olan ilgi arttı.

Bu çalışmada ise çevrimdışı el imzası tanıma ve doğrulama sistemine birkaç adımlı öğrenme metodunu kullanarak farklı bir bakış açısı getirildi.

Bir sinir ağı tarafından öğrenilen bir temsil uzayında her sınıfı örnekleri aracılığıyla temsil edebileceği fikrinden yola çıkılarak, birkaç adımlı öğrenme için Prototipik Ağlar adlı basit bir yöntem kullanıldı. Bu ağlar, birkaç atış ayarında iyi performans gösterecek şekilde eğitildi. Yaklaşım, son zamanlardaki meta-öğrenme yaklaşımlarından çok daha basit ve daha verimlidir ve Eşleştirme Ağları için geliştirilmiş karmaşık uzantılar olmadan bile son teknoloji sonuçlar üretir.

Model eğitiminden önce veri ön işleme ve analiz adımlarını uygulandı. Model farklı parametrelerle eğitildi ve çıkan sonuçları karşılaştırdık. Eğitilen modelleri kullanıcıya sunabilmek için bir arayüz tasarımı yapıldı.

Bu tip doğrulama sistemlerine neden ihtiyaç duyulduğu ve bu sistemlerin nasıl tasarlanabileceğine dair bilgiler edindik.

Daha fazla veri, daha iyi bir veri işleme ve analiz adımları ile birlikte çok başarılı imza tanıma ve doğrulama sistemleri tasarlanabilir.

9. KAYNAKÇA

Jake Snell, Kevin Swersky, Richard Zemel Prototypical Networks for Few-shot Learning

Luiz G. Hafemann , Robert Sabourin and Luiz S. Oliveira 2017 Offline Handwritten Signature Verification - Literature Review

Sounak Dey, Anjan Dutta , J. Ignacio Toledo , Suman K.Ghosh , Josep Lladós , Umapada Pal 2017 SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification

Gabe Alvarez, Blue Sheffer, Morgan Bryant, Offline Signature Verification with Convolutional Neural Networks

Luiz G. Hafemann, Robert Sabourin, Member, IEEE, and Luiz S. Oliveira 2019 Meta-learning for fast classifier adaptation to new users of Signature Verification systems

Moises Diaz, Miguel A. Ferrer, Donato Impedovo, Muhammed Imran Malik, Guiseppe Pirlo, Rejean Plamondon 2019 A Perspective Analysis of Handwritten Signature Technology

Hsin-Hsiung Kao, Che-Yen Wen 2020 An Offline Signature Verification and Forgery Detection Method Based on a Single Known Sample and an Explainable Deep Learning Approach

Raia Hadsell, Sumit Chopra, Yann LeCun 2005 Dimensionality Reduction by Learning an Invariant Mapping

<https://neptune.ai/blog/understanding-few-shot-learning-in-computer-vision>

<https://www.analyticsvidhya.com/blog/2021/05/an-introduction-to-few-shot-learning/>