KARADENİZ TEKNİK ÜNİVERSİTESİ
OF TEKNOLOJİ FAKÜLTESİ
YAZILIM MÜHENDİSLİĞİ BÖLÜMÜ

YAPAY ZEKA TEKNİKLERİ İLE EL ÇİZİMİ TANIMA UYGULAMASI

BİTİRME ÇALIŞMASI

Betül ÇALIK

2021-2022 GÜZ DÖNEMİ

KARADENİZ TEKNİK ÜNİVERSİTESİ
OF TEKNOLOJİ FAKÜLTESİ
YAZILIM MÜHENDİSLİĞİ BÖLÜMÜ

YAPAY ZEKA TEKNİKLERİ İLE EL ÇİZİMİ TANIMA UYGULAMASI

BİTİRME ÇALIŞMASI

Betül ÇALIK

Bu projenin teslim edilmesi ve sunulması tarafımca uygundur.

Danışman : Dr. Öğr. Üyesi Eyüp GEDİKLİ .....

2021-2022 GÜZ DÖNEMİ

IEEE Etik Kuralları
IEEE Code of Ethics

Mesleğime karşı şahsi sorumluluğumu kabul ederek, hizmet ettiğim toplumlara ve üyelerine en yüksek etik ve mesleki davranışta bulunmaya söz verdiğimi ve aşağıdaki etik kurallarını kabul ettiğimi ifade ederim:

1. Kamu güvenliği, sağlığı ve refahı ile uyumlu kararlar vermenin sorumluluğunu kabul etmek ve kamu veya çevreyi tehdit edebilecek faktörleri derhal açıklamak;
2. Mümkün olabilecek çıkar çatışması, ister gerçekten var olması isterse sadece algı olması, durumlarından kaçınmak. Çıkar çatışması olması durumunda, etkilenen taraflara durumu bildirmek;
3. Mevcut verilere dayalı tahminlerde ve fikir beyan etmelerde gerçekçi ve dürüst olmak;
4. Her türlü rüşveti reddetmek;
5. Mütenasip uygulamalarını ve muhtemel sonuçlarını gözeterek teknoloji anlayışını geliştirmek;
6. Teknik yeterliliklerimizi sürdürmek ve geliştirmek, yeterli eğitim veya tecrübe olması veya işin zorluk sınırları ifade edilmesi durumunda ancak başkaları için teknolojik sorumlulukları üstlenmek;
7. Teknik bir çalışma hakkında yansız bir eleştiri için uğraşmak, eleştiriyi kabul etmek ve eleştiriyi yapmak; hatları kabul etmek ve düzeltmek; diğer katkı sunanların emeklerini ifade etmek;
8. Bütün kişilere adilane davranmak; ırk, din, cinsiyet, yaş, milliyet, cinsi tercih, cinsiyet kimliği, veya cinsiyet ifadesi üzerinden ayırımcılık yapma durumuna girişmemek;
9. Yanlış veya kötü amaçlı eylemler sonucu kimsenin yaralanması, mülklerinin zarar görmesi, itibarlarının veya istihdamlarının zedelenmesi durumlarının oluşmasından kaçınmak;
10. Meslektaşlara ve yardımcı personele mesleki gelişimlerinde yardımcı olmak ve onları desteklemek.

IEEE Yönetim Kurulu tarafından Ağustos 1990'da onaylanmıştır.

ÖNSÖZ

Bu bitirme tezi çalışmasında, yapay zeka teknikleri ile mobil ortamda el çizimi tanıma uygulaması geliştirirken elde edindiğim bilgiler dikkatinize sunulmaktadır.

Bitirme tezi çalışmamın planlanması, araştırılması ve oluşumunda desteklerini esirgemeyen değerli danışmanım Dr. Öğr. Üyesi Eyüp GEDİKLİ'ye saygılarımı sunuyor ve teşekkürü borç biliyorum. Ayrıca bu çalışmanın geliştirildiği süreç içerisinde manevi desteğini her an hissettiğim aileme ve arkadaşlarıma da teşekkür ediyorum.

Betül ÇALIK
Trabzon, 2022

İÇİNDEKİLER

ÖZET

Derin öğrenme alanındaki gelişmeler, görsel sanatın mevcudiyeti ile birleştirildiğinde araştırmacılar için yeni bir fırsat yaratmıştır. Bu fırsat, görsel sanatları analiz etmek ve daha fazla anlamak için otomatik araçlarla birlikte imkan sunmasıdır. Bu faydaya ek olarak görsel sanatlar hakkında daha derin bir anlayış, sanatı daha geniş bir nüfus için erişilebilir kılma gibi özellikler verilebilmektedir.

Bu çalışmanın amacı yapay zeka teknikleri kullanılarak geliştirilmiş el çizimi tanıma uygulamasında kullanıcının çizdiği resimlerin sınıflandırılması ve görsel sanatların analizine destek sağlama amacı taşımaktadır.

Geliştirilen projede kullanıcının çizmiş olduğu resimler için görüntü işleme, elde edilen görüntünün sınıflandırılması için yapay zeka teknikleri ve uygulamanın iOS ortamında kullanılabilmesi için Swift programlama dili kullanılmıştır. Veri seti olarak açık kaynak olan Quick Draw [1] veri setinden yararlanılmıştır.

Proje geliştirilirken kullanıcının rahatlıkla kullanabileceği bir arayüze sahip olmasına dikkat edilmiştir.

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SEMBOLLER DİZİNİ

CNN: Evrişimli Sinir Ağları (Convolutional Neural Network)

SDLC: Yazılım Geliştirme Yaşam Döngüsü (Software Development Life Cycle)

UML: Birleşik Modelleme Dili (Unified Modelling Language)

1. GENEL BİLGİLER

1.1. Giriş

Resim, tarih öncesi çağlardan beri insanlar tarafından temel bir görsel iletişim aracı olarak kullanılmaktadır. Dokunmaya duyarlı olan telefon ve tablet gibi cihazların popülaritesi ve kullanımının yaygınlaşmasıyla birlikte günümüzde bu tarz cihazlar birçok temel ihtiyacı karşılamanın yanı sıra görsel iletişim aracı olan resim çiziminin de yapılabileceği ortamlar haline gelmişlerdir.

Sinirbilim çalışmalarına göre resim çizmenin, yazı yazmaya göre veriyi görselleştirmenin daha etkileyici bir yolu olduğu ileri sürülmektedir [2]. Resim çizmenin insanlık tarihindeki iletişim açısından önemi ve veriyi görselleştirmedeki etkileyiciliği göz önüne alındığında, resim çizmenin bu denli yaygın bir şekilde kullanılması kaçınılmazdır. Bu durum, otomatik el çizimi tanıma araçlarına olan ilginin artmasına neden olmuştur. Günümüzde kullanımı yaygınlaşmış olan tablet ve telefonlarda kullanılabilecek uygulamalar sayesinde otomatik el çizimi tanıma araçları bu platformlara entegre edilebilmektedir. Kullanıcının çizdiği resimlerin sınıflandırılabilmesi için eğitilmiş bir derin öğrenme ağından yararlanılabilir.

Resim sınıflandırma sisteminde derin öğrenme algoritmalarının kullanılmasının ana sebebi, derin öğrenme algoritmalarının birçok problem için yüksek performans ve yüksek doğruluk sunuyor olmasıdır. Derin öğrenmede kullanılan katmanlı yapıyla beraber tüm katmanların beraber öğrenebilmesi, üstündeki veya altındaki katmana bakarak kendini güncelleyebilmesi mümkündür. Bundan dolayı derin öğrenme modellerinin yüksek doğruluk oranı sunduğu savunulabilir.

1.2. Amaç

Bu projede amaç, mobil platformda kullanıcı deneyimine uygun olarak geliştirilmiş uygulamada kullanıcının el çizimi yapması, elde edilen çizimin derin öğrenme katmanlarını beslemesi, bu el çiziminin yapay zeka kullanılarak sınıflandırılması ve bu sınıflandırmanın kullanıcıya sunulmasıdır.

1.3. Yapay Zeka

Yapay zeka, 1950'li yıllarda yeni gelişmeye başlamış ve bilgisayar bilimcilerin bugün de cevabını aradığı "bilgisayarlar düşünebilir mi?" sorusuna cevap ararken ortaya çıkmıştır. Yapay zeka kısaca "normalde insanlar tarafından yerine getirilen düşünsel faaliyetleri otonom haline getirme" olarak tanımlanabilir [3].

Yapay zeka, doğayı taklit etme yöntemidir. Doğayı ne kadar iyi anlarsak, yapay zekayı o derece geliştirme fırsatımız doğar. Yapay zekanın ana hedefi insanların refahını arttırmaktır. Karmaşık problemlere bir çözüm önerisi sunarak insanların daha rahat bir yaşam sürmesine olanak sağlar.

Yapay zekanın temelinde birçok bilim bulunmaktadır. Bilgisayar bilimleri, felsefe, matematik, psikoloji, sinir bilimi, dil bilimi gibi birçok alan yapay zekanın oluşumuna katkı sağlar.

Yapay zekaya örnek olarak sürücüsüz araçlar, satranç bilgisayarı, web tabanlı uygulamalar, dijital kameralar, otomatik robotlar verilebilir. Bu yapay zeka içeren sistemler elle programlanmak yerine eğitilirler. Örneğin, tatil fotoğraflarınızı sınıflandıracak bir program yazmak isterseniz insanlar tarafından etiketlenmiş örnek fotoğrafları makine öğrenme sisteminizi eğitmek için kullanmanız halinde fotoğrafların hangi etikete ait olduğunu ilişkilendirebilecek istatistiksel kuralları kendi kendine öğrenebilir [3].

Yapay zeka, makine öğrenmesini, derin öğrenmeyi ve daha birçok alanı içine alan geniş bir alandır. Her ne kadar iyi tanımlanmış problemlerde yapay zeka bizlere iyi bir çözüm sunsa da, daha karmaşık problemler olan görüntü sınıflandırma, ses tanıma gibi problemlerde istenen sonuçları pek de sağlayamadığı görülmüştür. Bu yüzden ilerleyen yıllarda daha karmaşık problemleri çözmek için makine öğrenmesi yaklaşımı ortaya çıkmıştır.

Bu çalışmada görüntü sınıflandırma gibi karmaşık bir problem kullanıldığı için, makine öğrenimin gelişmiş bir alanı olan derin öğrenme metotları araştırılmış ve uygulanmıştır.

1.3.1. Yapay Zeka Teknikleri

Yapay zeka teknolojilerine ilişkin teknikler altı temel alana toplanabilir. Bunlar [4]:

- Uzman Sistemler
- Yapay sinir ağları
- Genetik algoritmalar
- Bulanık önermeler mantığı
- Zeki etmenler
- Örüntü tanıma

1.4. Makine Öğrenmesi

Makine öğrenmesi, yapay zekanın alt alanlarından birisidir. Makine öğrenmesi sistemleri eğitilerek çalışmaktadır ve elde ettikleri verilerden kullanışlı gösterimler aramaktadır. Bu fikirden yola çıkarak ses tanımadan otonom araç sürmeye kadar birçok farklı alana çözüm getirmektedir [3].

Makine öğrenmesi sistemleri giriş verilerine göre alt başlıklara ayrılırlar. Bu başlıklar aşağıdaki gibidir:

- **Danışmanlı (Supervised) Öğrenme:** Veri, etkiye tepki prensibiyle çalışan sistemlerden alınır ve giriş/çıkış düzeyinde organize edilir. *Tahmin* ve *sınıflandırma* problemleri danışmanlı öğrenme algoritmasıyla modellenebilir.
- **Danışmansız (Unsupervised) Öğrenme:** Sınıf bilgisi olmayan/verilmeyen veri içerisindeki grupları kesmeyi hedefler. Daha önceki verilerden yararlanarak bir sonraki modeli yorumlar. Kümeleme teknikleri danışmansız öğrenmeye uygundur.
- **Takviyeli (Reinforcement) Öğrenme:** Bazen öğretici, sisteme beklenen sonucu söylemez fakat sistemin ürettiği çıktı için "doğru/yanlış" şeklinde fikir belirtir. Bu öğrenme şekline takviyeli öğrenme adı verilmektedir. Bu öğrenme şeklinde öğrenme süreci geri bildirimlerle gerçekleşmektedir. Boltzmann makinesi, LVQ ve genetik algoritmalar takviyeli öğrenmeye örnek sayılabilir.

Bu çalışma kapsamında kullanıcının çizdiği el çizimleri sınıflandırılacağı için sınıflandırma problemi kapsamında değerlendirilmiştir ve bunun için danışmanlı (supervised) öğrenme tekniği kullanılmıştır.

1.5. Yapay Sinir Ağları

Yapay sinir ağları, beynin bir işlevini yerine getirme yöntemini modellemek için tasarlanan bir sistemdir. Bir yapay sinir ağı, yapay sinir hücrelerinin birbirleriyle çeşitli şekillerde bağlanmasıyla oluşmaktadır. Beynin öğrenme sürecine uygun olarak yapay sinir ağı, bir öğrenme sürecinden sonra bilgi toplama, hücreler arasındaki bağlantı aralıkları ile bu bilgiyi saklama ve genelleme yeteneğine sahip paralel dağıtılmış bir işlemcidir.

Yapay sinir ağları bilinen hesaplama yöntemlerinden farklı bir hesaplama yöntemi önermektedir. Bulundukları ortama uyum sağlayan, adaptif, eksik bilgi ile çalışabilen, belirsizlikler altında karar verebilen, hatalara karşı toleranslı olan bu hesaplama yönteminin hayatın hemen hemen her alanında başarılı uygulamalarını görmek mümkündür. Oluşturulacak olan ağın yapısının belirlenmesinde, ağ parametrelerinin seçiminde, belirli bir standardın olmaması, problemlerin sadece nümerik bilgiler ile gösterilebilmesi, eğitimin nasıl bitirileceğinin bilinmemesi ve ağın davranışlarını açıklayamamasına rağmen bu ağlara olan ilgi her geçen gün artmaktadır.

Özellikle, sınıflandırma, örüntü tanıma, sinyal filtreleme, veri sıkıştırma ve optimizasyon çalışmalarında yapay sinir ağları en güçlü teknikler arasında sayılabilir.

Yapay sinir ağlarının özellikleri:

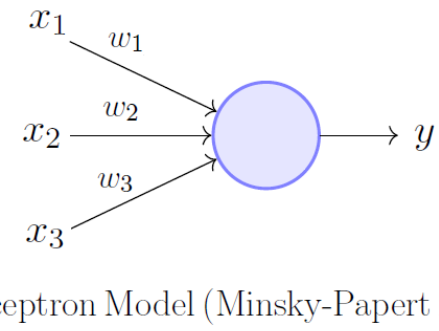
- Doğrusal değillerdir.
- Görülmemiş örnekler hakkında bilgi üretebilirler.
- Eksik bilgi ile çalışabilirler.
- Sınıflandırma yapabilirler
- Sadece nümerik (sayısal) bilgiler ile çalışabilirler.
- Analizi ve tasarımı kolaydır.

Yapay sinir ağlarının görülmemiş örnekler hakkında bilgi üretebilmesi ve sınıflandırma yapabilmesi gibi özelliklerinden dolayı geliştirilmiş olan projede kullanılmıştır.

1.5.1. Yapay Sinir Ağlarının Genel Yapısı

Yapay sinir ağları katmanlar şeklinde düzenlenmektedir. Tek katmanlı yapay sinir ağına *perceptron* ismi verilmektedir. En basit haliyle yapay sinir ağı, giriş katmanı ve çıkış katmanı olmak üzere iki katmandan oluşmaktadır. Problemin girdileri ağırlıklar ile çarpılıp toplandıktan sonra elde edilen değerin bir eşik değerinden büyük veya küçük olmasına göre girdinin sınıfı belirlenir. Sınıflar 1 veya –1 rakamları (bazen 1 ve 0 rakamları) ile gösterilir. Öğrenme sırasında hem ağın ağırlıkları hem de eşik değer ünitesinin ağırlık değeri değiştirilir. Eşik değer ünitesinin çıktısı sabit olup 1'dir [5].

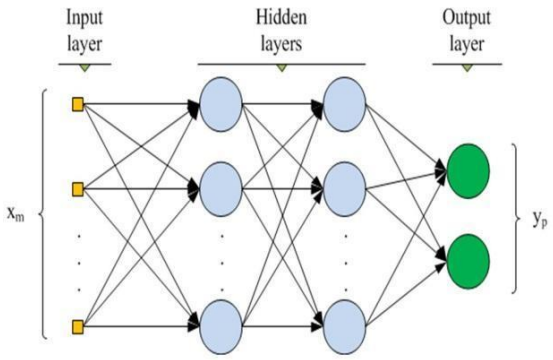
Aşağıdaki şekilde en basit yapay sinir ağı olan perceptron'un modeli verilmiştir.



Şekil 1. Perceptron Modeli [Gerry Saporito, What is a perceptron?, 2019]

Yapay sinir ağı, birbirine bağlı yapay nöronlardan oluşan bir ağdır. Bu birbirine bağlı düğümler, insan beynini taklit ederek bilgileri birbirine aktarır. Her bir düğüm girdi alır ve iletmeden önce bazı işlemler gerçekleştirir. Bu işlemler *aktivasyon fonksiyonu* olarak adlandırılan doğrusal olmayan matematiksel bir fonksiyon tarafından gerçekleştirilir [6]. Giriş ve çıkış katmanları arasında *gizli katman* adı verilen katmanlar bulunabilir. Bu katmanlar sinir ağının performansını arttırmak için kullanılmaktadır. Gizli katmanların ve içerisindeki düğümlerin artması genellikle doğruluk oranını arttırır fakat çok fazla hesaplama gücü gerektirir. Her problem için katman sayısını arttırmak, performansı arttırmayabilir.

Aşağıdaki şekilde birden çok katmana sahip yapay sinir ağının modeli verilmiştir.



Şekil 2. Çoklu Katmanlı Yapay Sinir Ağı [Md Tanjil Sarker, Basic Application and Study of Artificial Neural Networks, 2017]

Düğümler arasında belirli ağırlıklar vardır. Bu ağırlıklar modelin her öğrenme turunda güncellenir. Performans (doğruluk) oranı yüksekse ağırlıklar güncellenmez. Bu ağırlıklar her turda geri besleme ile doğruluğu arttırmak adına güncellenir. Ağdan çıkan sonucun doğruluğuna göre belirli fonksiyonlar sayesinde bu ağırlıklar yenilenir [6].

Yapay sinir ağlarının katmanları:

- **Giriş Katmanı (Input Layer):** Dış dünyadan gelen girdileri alarak ara katmana gönderir. Bu katmanda bilgi işleme bulunmamaktadır. Gelen bilgi olduğu gibi bir sonraki katmana gönderilir. Girdi katmanındaki her bir düğüm bir sonraki katmanda bulunan düğümlerin hepsine bağlanır. Genel olarak problemin bağımsız değişkeni sayısı, giriş katmanındaki düğümlerin sayısına eşittir.
- **Gizli Katman (Hidden Layer):** Gizli (ara) katmanlar, girdi katmanından gelen bilgileri işleyerek (ağırlık hesaplaması yaparak) bir sonraki katmana gönderir. Birden fazla ara katman bulunabilir.
- **Çıkış Katmanı (Output Layer):** Çıkış katmanı, varsa ara katmandan gelen bilgileri, yoksa giriş katmanından gelen bilgiyi işlemektedir. Girdi düğümlerine karşılık ağın ürettiği çıkışı dış dünyaya gönderir. Bütün giriş düğümlerin bir çıkışı bulunmaktadır.

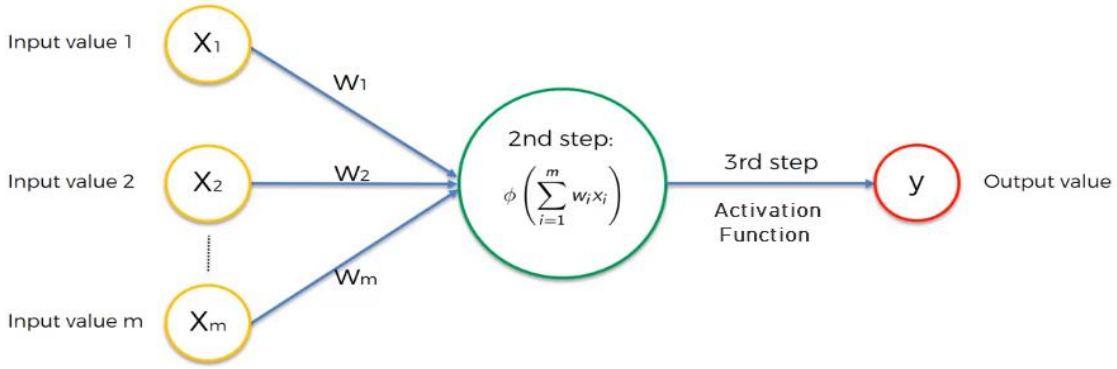
1.5.2. Yapay Sinir Ağının Tasarımı

Yapay sinir ağı tasarımının keskin hatları bulunmamaktadır. Problem tanımına, veri setlerine, istenilen çıktıya göre tasarımlar uygulanmaktadır. Her problem özeldir ve ona ait

bir yapay sinir ağı tasarımı bulunmaktadır.

Yapay sinir ağı tasarlanırken baz alınması gereken terimler aşağıda verilmiştir.

- **Aktivasyon Fonksiyonu:** Aktivasyon fonksiyonlarının amacı, yapay sinir ağlarının doğrusallığını ortadan kaldırmaktır. Bunun nedeni, gerçek hayatta birçok verinin doğrusallıktan uzak olmasıdır.
  Aynı zamanda, aktivasyon fonksiyonları her bir nöronun 0 - 1 veya (-1) - 1 arasında normalize olmasına yardımcı olur.



Şekil 3. Aktivasyon Fonksiyonu Modeli [Andrea Perlato, The Activation Function]

Aktivasyon fonksiyonu bulunmadan yapay sinir ağı, sınırlı öğrenme gücüne sahip doğrusal bir bağlantı görevi görür.

Doğrusal olmayan (non-linear) bazı aktivasyon fonksiyonları [6]:

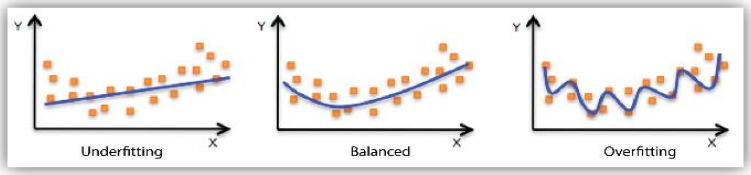
- **Sigmoid Fonksiyonu:** Çıkış verilerinin [0,1] arasında olmasını sağlar. Kullanımı performansı düşürebilir. Çok büyük ve çok küçük giriş verilerinde nerdeyse hiçbir işe yaramaz. Oldukça yavaştır.
- **TanH Fonksiyonu:** Sıfır merkezli bir aktivasyon fonksiyonudur. Son derece negatif ve pozitif değerlere sahip girdileri modellemeyi kolaylaştırır. Dezavantajı Sigmoid fonksiyonuna benzemesidir.
- **ReLu Fonksiyonu:** Özellikle Convolutional Neural Network'te (CNN) oldukça sık kullanılmaktadır. Çok hızlı çalışır. Eğer girdiler 0 veya negatif olursa fonksiyon sıfırlanır ve ağ hiçbir şey öğrenemez. Buna *dying ReLu* adı verilmektedir.
- **Softmax Fonksiyonu:** Çıkış verilerinin [0,1] arasında olmasını sağlar. Sigmoid'e oldukça benzemektedir. Sigmoid fonksiyonundan farkı birçok kategoriye sahip veriler için kullanılmasıdır.

- **Kayıp (Hata) Fonksiyonu:** Yanlış tahminlerden kaynaklanan maliyeti ölçen fonksiyondur. Asıl amaç, kayıp skorunu en aza indirgemektir. Kayıp skoru, gerçek değerden tahmin edilen değer çıkartılarak hesaplanır. Kayıp skoru, geri besleme (back propagation)'da kullanılır. Yani, geriye dönülerek ağırlıkların güncellenmesinde kullanılır.

  Bazı hata fonksiyonları:

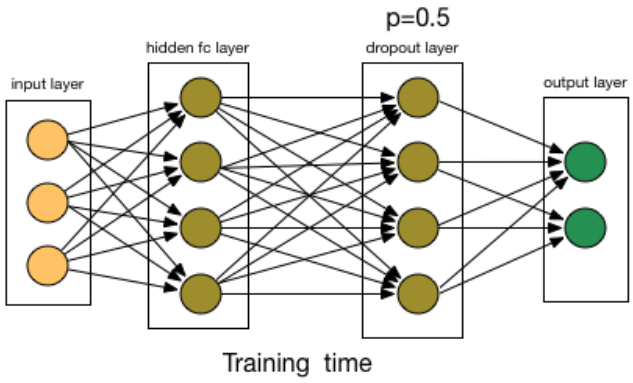
  - **Binary Cross Entropy:** Çıkış değeri 0-1 arasında olacağı tahmin edilen modeller için kullanılmaktadır.
  - **Mean Squared Error (MSE):** Regresyon problemlerinde gelecek değeri tahmin etmek için kullanılır.
  - **Categorical Cross Entropy:** İki veya daha fazla kategoriye sahip problemlerde kullanılır.
- **Optimizasyon:** Optimizasyon ile ağırlıklar seçilen algoritmaya göre güncellenmektedir. Bu yüzden optimizasyon algoritması bir ağdan başka bir ağa göre değişkenlik gösterebilir.

  Bazı optimizasyon algoritmaları: SGD, Adam, Adagrad, Adadelta, RMSProp şeklindedir.
- **Hata Çeşitleri:** Yapay sinir ağı modeli geliştirirken öncelikli amaç en iyi genelleme yapan modele erişmektir. Kullanılan modelin veri setinden öğrendiklerine göre iki çeşit hata çeşidi ile karşı karşıya kalırız. Bu hatalar aşağıda verilmiştir.
  - **Eksik Öğrenme (Underfitting):** Model öğrenmeyi tam gerçekleştiremediğinde karşılaşılan durumdur. Daha fazla veri kullanılarak veya daha karmaşık bir model geliştirilerek (ara katman eklemek gibi) bu durum çözülebilir.
  - **Aşırı Öğrenme (Overfitting):** Modelin aşırı öğrenmesi her ne kadar iyi gibi algılansa da aslında iyi değildir. Model, girilen verileri aşırı derecede ezberlediğinde overfitting durumu ortaya çıkar. Model öğrenmek yerine ezberlediği için eklenen yeni verilerde başarısızlığa uğrar. Veri setinin çeşitlendirilmesiyle veya bazı teknikler uygulanarak bu durum çözülebilir.



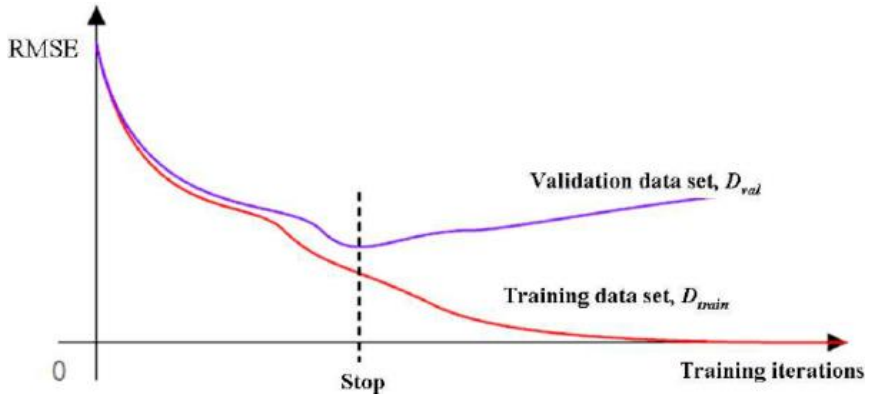
Şekil 4. Eksik ve Aşırı Öğrenme [AWS Developer Guide, Model Fit: Underfitting vs. Overfitting]

- **Başarım İyileştirme (Regularization):** Aşırı öğrenme (overfitting) ile karşılaşılan problemlerde, aşırı öğrenmenin üstesinden gelmek için öğrenilen tekniklere başarım iyileştirme denmektedir. Elde edilen modelin görünmeyen verilerinin genelleştirilmesini iyileştirmek için kullanılır. Karmaşık problemleri sınırlayan bir tekniktir. Aşırı öğrenmenin üstesinden gelebilmek için uygulanabilecek bazı teknikler aşağıda verilmiştir.
  - **Düğüm Seyreltme (Drop-Out):** Gizli veya giriş katmanından belirli kurallara göre (eşik değeri kullanarak veya rastgele) belirli düğümlerin kaldırılması tekniğidir. Örneğin aşağıdaki şekilde görüleceği üzere eşik değeri (treshold) 0.5 olarak belirlendiği bir ağ modelinde drop-out uygulandığında drop-out uygulanan gizli katmandaki node sayısı bir sonraki katmanda yarıya düşürülmüştür [7].



Şekil 5. Düğüm Seyreltme [Cristina Scheau, Regularization In Deep Learning, 2016]

  - **Ezberlemeden Önce Durdurma (Early Stopping):** Model, overfit ile karşılaşacağı an eğitimi durdurur.



Şekil 6. Ezberlemeden Önce Durdurma [CommonLounge, Regularization Methods in Deep Learning]

1.6. Derin Öğrenme

Derin öğrenme, birbirini takip eden katmanlarda veriler işlenirken giderek artan şekilde daha kullanışlı gösterimler elde edilebilen makine öğrenmesinin bir alt alanıdır. Derin derken kastedilen derin öğrenmenin bir takım derin bilgiler elde edebilmesi değil, birbirini takip eden gösterim katmanları ifade edilmektedir. Modeldeki katman sayısı modeldeki derinliği oluşturmaktadır. Buna *katmanlı gösterim öğrenimi* veya *hiyerarşik gösterim öğrenme* demek de uygun olabilir. Modern derin öğrenme modelleri, onlarca hatta yüzbinlerce birbirini takip eden katmanlar içermektedir. Oysa diğer makine öğrenme algoritmaları genelde bir veya iki katmandan oluşur ki buna da sığ öğrenme denilebilir [3].

Makine öğrenmesi ile çözümünde zorluklar yaşanan aşağıdaki konularda derin öğrenme önemli çözümler geliştirmiştir [3]:

- Görüntüleri yüksek başarı oranında sınıflandırma.
- Sesleri yüksek başarı oranında tanıma.
- Amazon Alexa gibi dijital yardımcılar hayat bulmuştur.

Bazı derin öğrenme algoritmaları aşağıdaki gibi verilebilir:

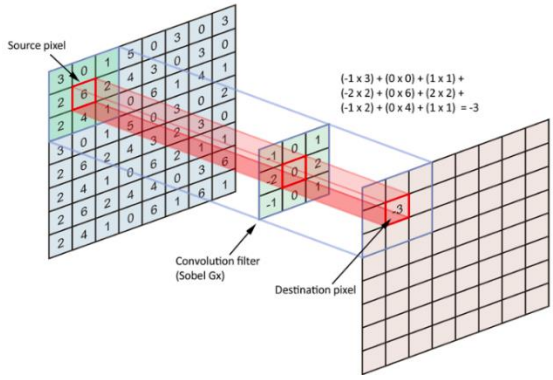
- Convolutional Neural Network (CNN)
- Recurrent Neural Network (RNN)
- Multilayer Perceptron (MLP)

Derin öğrenme algoritmaları neredeyse her tür veriyle çalışabilmektedir. Karmaşık problemleri çözebilmek için büyük miktarda bilgi ve işlem gücü gerektirir.

1.6.1. Evrişimli Sinir Ağları (CNN)

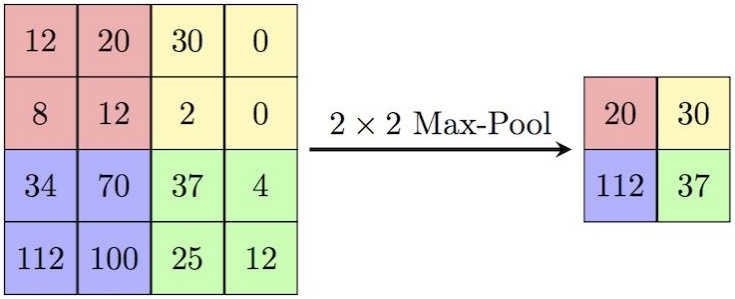
Evrişimli Sinir Ağları, görüntü veya video içeren girdiler için derin öğrenmeyi kullanan bir yapay zeka algoritmasıdır. Evrişimli katmanlar bölgesel örüntüler halinde öğrenimini gerçekleştirirler. Resimler için evrişimli katmanları kullanacak olursak, girdilerden küçük iki boyutlu pencereler (frame) olarak öğrenirler. Evrişimli bir sinir ağı *Conv2D* ve *MaxPooling2D* katmanlarının üst üste getirilmesiyle oluşturulmaktadır. Conv2D katmanı filtreleme için kullanılmaktadır. Evrişim katmanları öğrenmeyi bu şekilde gerçekleştirir.

Evrişimli sinir ağlarındaki Conv2D katmanında filtreleme aşağıdaki görseldeki gibi yapılmaktadır:



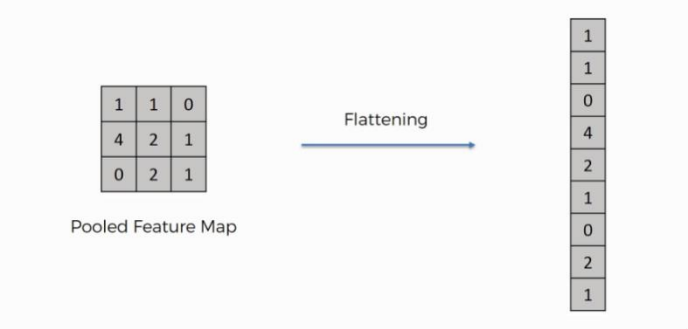
Şekil 7. CNN Filtreleme [Lakshmanan Meiyappan, Beginners Guide - CNN Image Classifier | Part 1]

MaxPooling2D katmanı ise boyutun indirgenmesi amacıyla kullanılmaktadır. Boyut indirgemenin amacı, daha az katsayıyla işlem yapmak ve evrişim katmanlarının her seferinde daha büyük pencerelere bakmasını sağlamaktır.



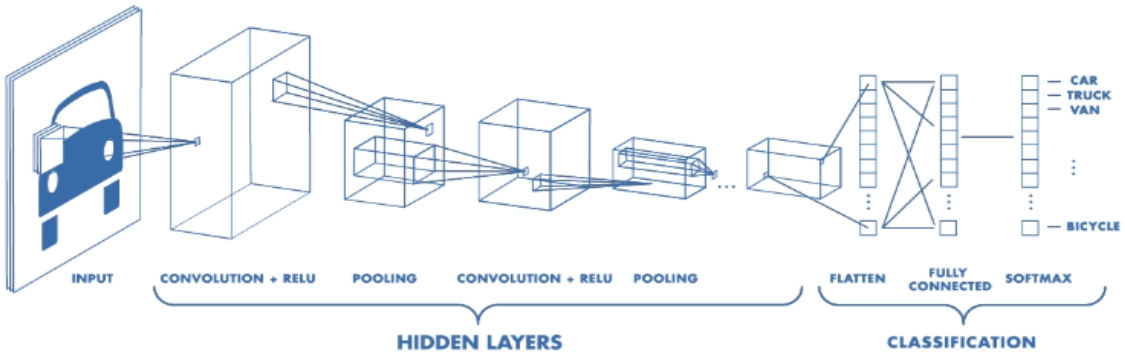
Şekil 8. CNN MaxPooling ile Boyut İndirgeme [Computer Science Wiki, Max-pooling / Pooling]

Evrişim işlemi ile görüntüden yalnızca öznitelikler çıkarılır, kendi başına tahmin edemezler. Bu sorunun üstesinden gelmek için Flatten katmanı eklenmelidir. Bu katman evrişimden çıktıyı almak ve doğru çıktıyı tahmin etmek için eğitilir. Evrişimin çıktısı 3 boyutlu bir matristir, bu nedenle her değeri tamamen bağlantılı bir katmandaki bir nörona bağlayarak düzleştirme işlemi uygulamamız gerekir.



Şekil 9. CNN Flattening İşlemi [SuperDataScience Team, Convolutional Neural Networks (CNN): Step 3 – Flattening, 2018]

Evrişimli Sinir Ağı (CNN)'in bütün katmanlarla birlikte mimarisel gösterimi aşağıdaki şekildeki gibi özetlenebilir:



Şekil 10. CNN Mimarisi [Mathworks, Introduction to Deep Learning: What Are Convolutional Neural Networks?, 2017]

2. YAPILAN ÇALIŞMALAR

Bu çalışmada el çizimi tanıma uygulaması geliştirilirken kullanılan teknik ve yöntemler ele alınmıştır. El çizimi tanıma uygulaması bir mobil uygulaması olarak geliştirilmiştir ve iOS platformuna uygun bir şekilde çalışmaktadır. Oluşturulan uygulamada kullanıcı resim çizme alanına resim çizmekte ve bu resmin hangi sınıfa ait olduğu belirlenmektedir.

2.1. Yazılım Yaşam Döngüsü (SDLC)

Projenin sağlıklı bir şekilde yürütülebilmesi ve başarıyla tamamlanabilmesi için uygun bir yazılım yaşam döngüsü izleyebilecek bir yöntem kullanılmalıdır. Belirli bir yöntem uygulanmayan projelerin üst yönetim tarafından takibi oldukça güçleşir [8].

Yazılımın ürününün hem üretim hem de müşterideki kullanım süreci boyunca geçirdiği tüm aşamalar yazılım geliştirme yaşam döngüsü (“software development life cycle”, “SDLC”) olarak adlandırılır. Yazılım geliştirme süreci, zamanlamaya dayalı ve içerik olarak bölünmüş aşamalardan oluşmaktadır. Bu sayede yazılım planlı bir şekilde geliştirilmektedir. Yazılım işlevleri ile ilgili gereksinimler sürekli olarak değiştiği ve genişlediği için, söz konusu aşamalar sürekli bir döngü biçiminde ele alınır. Döngü içerisinde herhangi bir aşamada geriye dönmek ve tekrar ilerlemek söz konusudur [9].

2.2. Yazılım Yaşam Döngüsü Aşamaları

Yazılım yaşam döngüsünün aşamaları aşağıdaki gibi verilebilir:

1. Planlama Aşaması
2. Analiz Aşaması
3. Tasarım Aşaması
4. Gerçekleştirim Aşaması
5. Test Aşaması
6. Uygulama Aşaması
7. Bakım Aşaması

Şekil 11. Yazılım Yaşam Döngüsü Aşamaları [Big Water Consulting, Software Development Life Cycle (SDLC)]

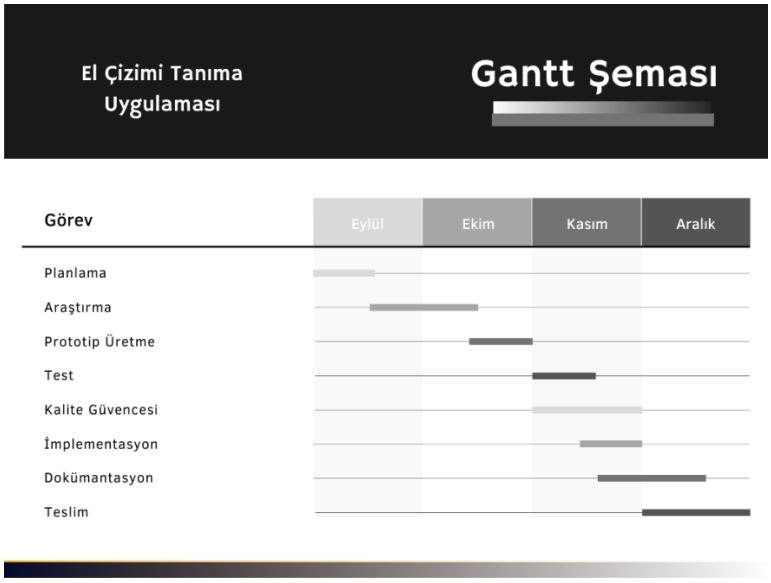
2.2.1. Planlama Aşaması

Planlama aşaması, projenin genel hatlarının ortaya çıkarıldığı aşamadır. Bu aşamada fizibilite (yapılabilirlik) çalışması yapılır. Projeye dair planlamalar bu aşamada yapılır. Bu çalışmada planlama aşamasında Gantt çizelgesinden yararlanılmıştır.

Gantt Çizelgesi Henry Gantt tarafından 1910'lu yıllarda ortaya çıkarılmış ve günümüze kadar üretim ve proje planlamalarında yaygın bir şekilde kullanılan bir görsel iş planlama aracıdır [10]. İş yönetiminde planlılığı sağladığı için oldukça kullanışlıdır. Bir Gantt Çizelgesi tasarlanırken iki husus dikkate alınır:

1. **Görev Listesi:** Her bir satır görev paketleri içerir.
2. **Zaman Çizelgesi:** Her bir görev için verilen zaman bu kısımda yer alır.

Planlama aşamasında Gantt Çizelgesi'nin kullanılma sebebi proje takibini kolaylıkla yapabilmektir. Bu çalışmaya dair oluşturulan Gantt çizelgesi aşağıda verilmiştir.



Tablo 1. Gantt Şeması

2.2.2. Analiz Aşaması

Analiz aşamasında proje detayları belirlenir. Gereksinimler ve kullanım senaryosu diyagramları bu aşamada oluşturulur. Gereksinim analizi, fonksiyonel gereksinim analizi ve fonksiyonel olmayan gereksinim analizi olarak iki alt başlıkta incelenebilir.

2.2.2.1. Fonksiyonel Gereksinim Analizi

Fonksiyonel gereksinim, sistemin yapması gereken özellikleri tanımlayan bir analizdir. Fonksiyonel gereksinim, belirli koşullar karşılandığında sistemde tanımlanan

belirli işlevleri kapsar. Bu uygulamaya dair fonksiyonel gereksinimler, gerekli incelemeler doğrultusunda aşağıda tanımlanmıştır.

- Uygulama, kullanıcının çizmesi için bir kelime önerebilir.
- Uygulama, kullanıcının çizmesi gereken süreyi belirleyebilir.
- Uygulama, kullanıcı verilen kelimeyi çizdiği takdirde bir bilgilendirme ekranı gösterebilir.
- Uygulama, kullanıcı verilen kelimeyi gereken süre içerisinde çizemediği takdirde bir bilgilendirme ekranı gösterebilir.
- Uygulama, her iki bilgilendirme ekranından sonra yeni kelimeye geçebilir.
- Kullanıcı, uygulama üzerinde resim çizebilir.
- Kullanıcı, uygulama üzerinde çizdiği resmi silebilir.
- Kullanıcı, verilen kelimeyi "ileri" butonuna basarak değiştirebilir.
- Kullanıcı, verilen kelime için kalan süreyi uygulama üzerinde görebilir.

2.2.2.2. Fonksiyonel Olmayan Gereksinim Analizi

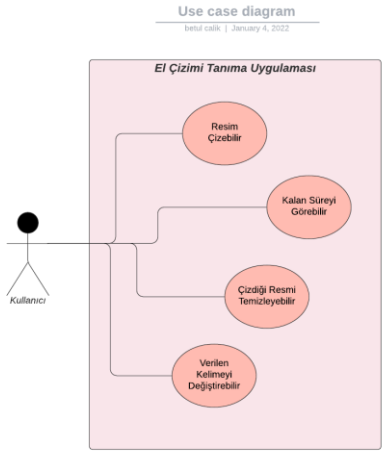
Fonksiyonel olmayan gereksinim analizi, belirli davranışlar yerine sistemin çalışmasını belirleyen kriterleri tanımlamaktadır. Fonksiyonel olmayan gereksinimler kalite özelliklerini belirtmektedir. Uygulamaların kararlı çalışması için gerekli bir dizi standardı temsil etmektedir.

- **Güvenilirlik (Security):** Geliştirilen yazılım hiçbir kullanıcı verisi barındırmadığından güvenilirdir.
- **Kullanılabilirlik (Usability):** Geliştirilen yazılımın kullanıcı dostu bir arayüze sahip olduğu için kullanımı kolaydır.
- **Uyumluluk (Compatibility):** Geliştirilen yazılım yalnızca iOS ve iPadOS işletim sistemlerine sahip cihazlarda kullanılabilmektedir.
- **İşlevsellik (Functionality):** Geliştirilen yazılım kullanıcıların çizimlerini tahmin edebilen, makine öğrenimi ile geliştirilmiş bir oyundur.

2.2.2.3. Kullanım Senaryosu (Use-Case) Diyagramları

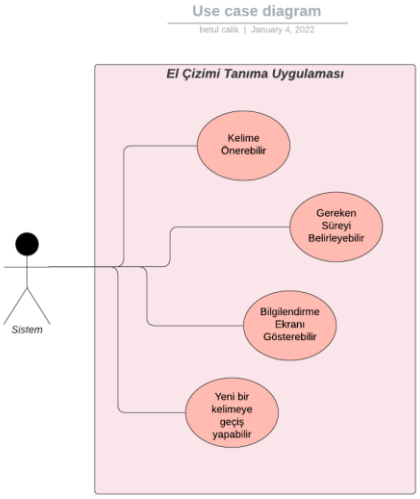
Kullanım senaryosu diyagramları, sistemdeki fonksiyonelliği görselleştirmek amacıyla kullanılır. Kullanım senaryosu diyagramı, bir kullanıcı ve bir sistem arasındaki etkileşimi anlatan senaryo topluluğudur.

Geliştirilen uygulamaya dair kullanım senaryosu diyagramları aşağıdaki gibidir.
Kullanıcı için kullanım senaryosu diyagramı:



Tablo 2. Kullanım Senaryosu Diyagramı - Kullanıcı

Sistem (uygulama) için kullanım senaryosu diyagramı:



Tablo 3. Kullanım Senaryosu Diyagramı - Sistem

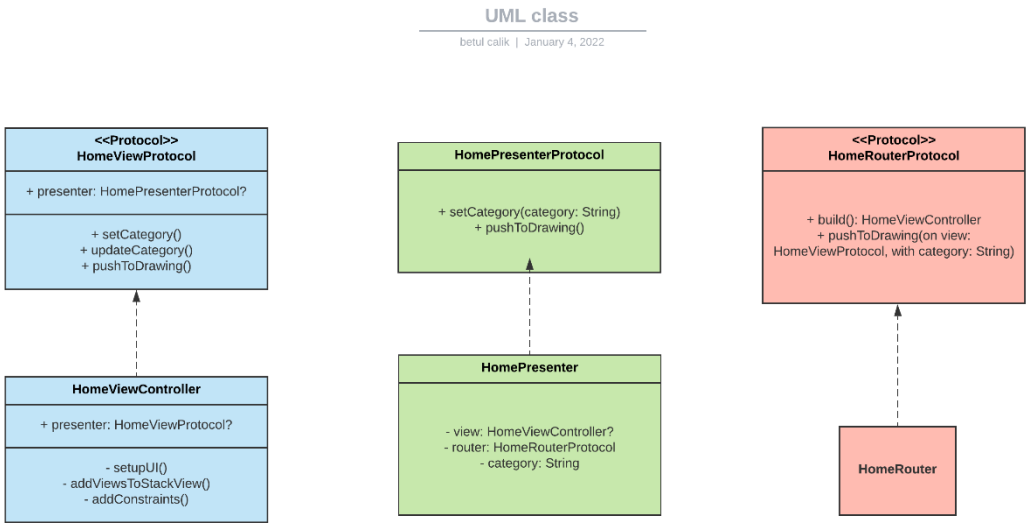
2.2.3. Tasarım Aşaması

Tasarım aşaması, yazılım sisteminin temel yapısını oluşturur. Bu aşamada, yapılacak işlemler detaylanır. Yazılımın yetenekleri, arayüzleri belirlenir. Kullanım senaryosu harici UML diyagramları bu aşamada oluşturulur. Sistemi detaylandırdığı için önemli kısımlara odaklanmayı sağlar. Bu aşama, kodlamaya geçmeden önceki son aşamadır. Bu aşamada "problem nasıl çözülür?" sorusuna cevap aranacaktır.

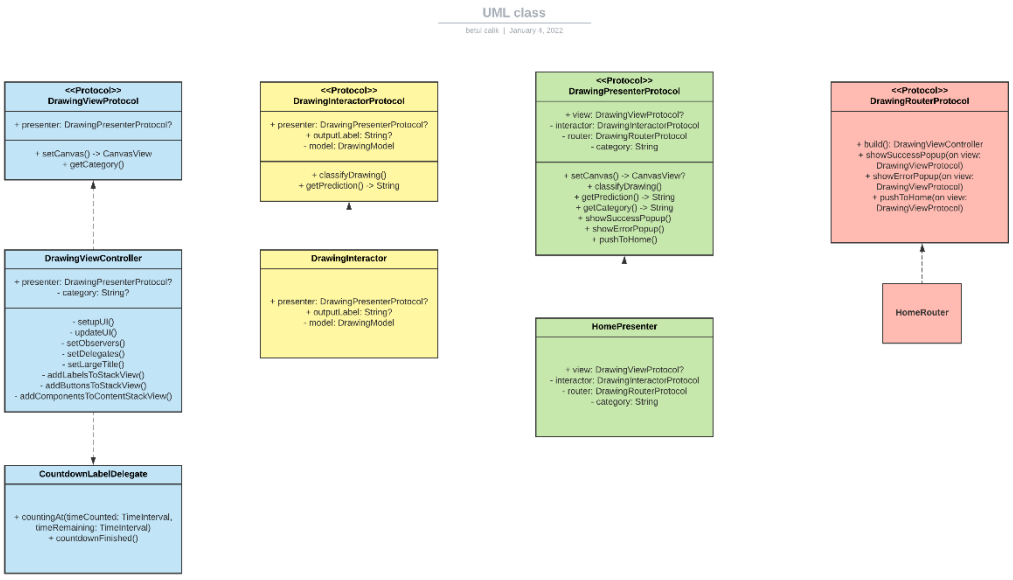
2.2.3.1. Sınıf Diyagramları

Sınıf diyagramları, sistemdeki sınıfları görselleştirmek amacıyla kullanılır. Sistemin sınıflarını, niteliklerini ve nesneler arasındaki ilişkileri göstermektedir. Bir tür statik yapı diyagramıdır.

Geliştirilen projede temiz mimari kullanılmasına özen gösterilmiş ve tüm proje VIPER tasarım desenine göre oluşturulmuştur. Projede home ve drawing olmak üzere iki modül bulunmaktadır. Aşağıda home modülü ve drawing modülü için iki ayrı sınıf diyagramı bulunmaktadır.

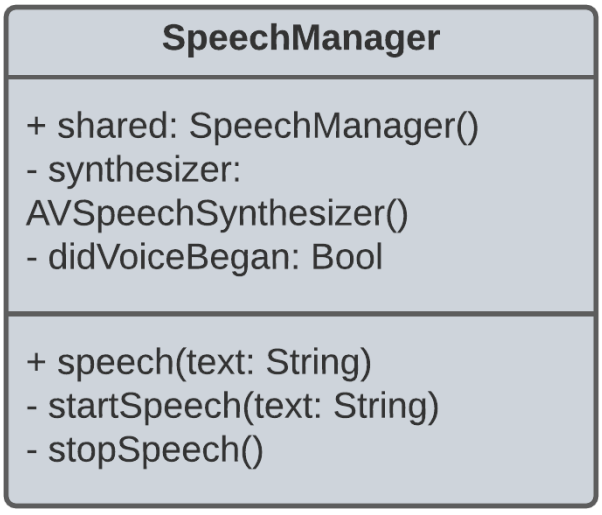


Tablo 4. Sınıf Diyagramı - Home Modülü



Tablo 5. Sınıf Diyagramı - Drawing Modülü

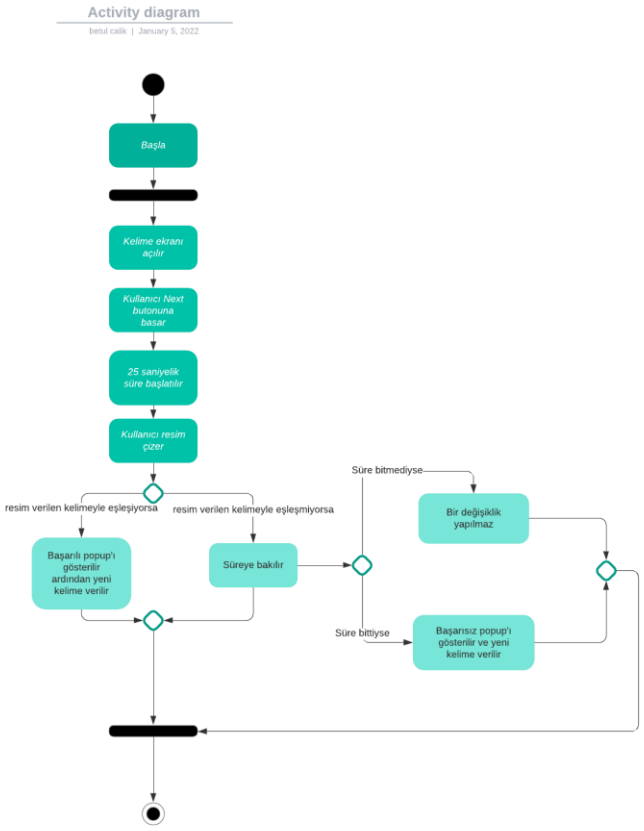
Geliştirilen projede ayrıca makine öğrenmesinin kullanıcı çiziminden tahmin ettiği kelimeyi seslendirmesi özelliği de mevcuttur. Seslendirme için managers modülünde SpeechManager class'ı oluşturulmuştur. SpeechManager'a ait sınıf diyagramı ise aşağıdaki gibidir.



Tablo 6. Sınıf Diyagramı - Speech Manager

2.2.3.2. Aktivite Diyagramı

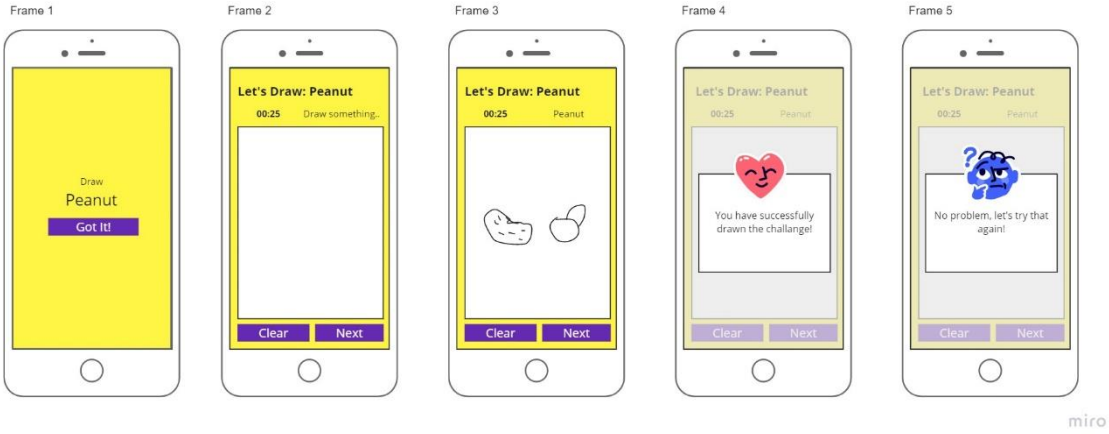
Aktivite diyagramı, sistemdeki eylemlerin akışını gösteren temel UML diyagramlarından birisidir. Diyagram, yukarıdan başlayarak aşağıya doğru dallanır. Bu uygulamaya dair aktivite diyagramı aşağıda verilmiştir.



Tablo 7. Aktivite Diyagramı

2.2.3.3. Arayüz Tasarımı

Arayüz, kullanıcı ve uygulama arasında etkileşim sağlayan bir bileşendir ve arayüzün kullanılabilir bir şekilde tasarlanması kullanıcı açısından önemli bir yere sahiptir. Geliştirilen mobil arayüzünde kullanıcı deneyimi göz önünde bulundurularak yalın ve anlaşılır bir tasarıma dikkat edilmiştir. Geliştirilen arayüz tasarımı aşağıdaki diyagramda gösterilmiştir.



Tablo 8. Uygulama Arayüzü Tasarımı

Yukarıdaki arayüz tasarımı incelendiğinde ilk ekran, uygulama açıldıktan sonra kullanıcıya gösterilecek ilk ekran olma özelliği taşımaktadır. Bu ekranda kullanıcıya çizmesi gereken kelime verilmiştir. Kullanıcı "Got it! (Anladım!)" butonuna bastıktan sonra karşılaşacağı ekran Frame2'de gösterilmiştir. Bu ekranın sol üst kısmında kullanıcının resmi çizmesi gereken süre, sağ üst kısmında ise kullanıcının çizmesiyle başlayacak olan tahmin yer almaktadır.

Frame 3'te bir resim çizilerek sağ üst kısımdaki tahminin değiştiği gözlemlenebilmektedir. Kullanıcı "clear" butonuna basarak resmi temizleyebilir, "next" butonuna basarak bir sonraki kelimeye yani Frame 1'e geri dönebilmektedir.

Frame 4, kullanıcı verilen resmi çizdiğinde gösterilecek olan ekrandır ve kalan süreye bakmaksızın doğru kelime çizildiğinde anlık olarak gösterilmektedir. Frame 5 ise yalnızca kullanıcı süresi dolduğunda ve kullanıcı resmi doğru çizemediğinde gösterilen ekrandır.

Frame 4 ve Frame 5 ekranları yalnızca 5 saniye gözükmektedir ve her iki ekrandan sonra kullanıcı Frame 1'e yönlendirilerek uygulamanın akışı bu şekilde sağlanmaktadır.

2.2.4. Gerçekleştirim Aşaması

Gerçekleştirim aşaması yazılımın oluşturulmaya başlandığı aşamadır. Bu aşamada projenin geliştirilmesi başlar ve tasarım aşamasında oluşturulan tasarımlara göre proje kodlanmaya başlanır. Bu aşama, yazılım yaşam döngüsünün en uzun süren aşamasıdır ve bu aşamada çeşitli programlama dilleri ve derleyiciler gibi çeşitli programlama araçları kullanılır.

2.2.4.1. Veri Seti

Verilerin oluşturduğu koleksiyona veri seti adı verilmektedir. Ham veriler işlenerek yararlı bilgilere dönüştürülmektedir. Bu projede, yapay zekayı eğitebilmek için Google tarafından geliştirilmiş açık kaynaklı bir veri seti olan **Quick Draw** [1] veri setinden yararlanılmıştır. Bu veri seti kullanılarak yapay zeka eğitilmiş ve eğitilen yapay zeka proje içerisinde kullanıcıların resimlerini tahmin etmede kullanılmıştır.

Quick Draw veri seti, Quick Draw oyununu oynayan kullanıcıların katkılarıyla 345 kategoride 50 milyon çizimden oluşan bir koleksiyondur.

Şekil 12. Quick Draw Veri Seti Örneği [Google Creative Lab, Quick Draw Dataset]

Quick Draw veri seti Google tarafından birkaç farklı dosya tipine dönüştürülmüştür böylece indirilmesi ve kullanımı kolaylaştırılmıştır. Bu formatlar kısaca aşağıda verilmiştir:

1. Basitleştirilmiş Çizim Dosyaları (.ndjson)
2. Binary Dosyaları (.bin)
3. Numpy Bitmap Dosyaları (.npy)

Geliştirilmiş olan projede Numpy Bitmap Dosyaları kullanılmıştır. Bu formatta tüm basitleştirilmiş çizim dosyaları 28x28 grayscale (gri tonlamalı) bitmap'e dönüştürülmüştür. Numpy Bitmap dosyalarını indirmek ve yapay zekayı eğitmek için Python programlama dili kullanılmış, eğitim için ise Turi Create'ten yararlanılmıştır. Turi Create, Apple tarafından geliştirilen açık kaynaklı bir Python kütüphanesidir ve makine öğrenmesi modeli oluşturmak için kullanılmaktadır.

Quick Draw veri seti oldukça fazla sınıfa ve veriye sahip olduğu için projede yalnızca belirli kategoriler kullanılmıştır. Bu kategoriler toplamda 23 tanedir ve sırasıyla aşağıdaki tabloda kullanılan tüm kategoriler verilmiştir:

KATEGORİLER
Apple (Elma)
Banana (Muz)
Bread (Ekmek)
Broccoli (Brokoli)
Cake (Kek)
Carrot (Havuç)
Coffee Cup (Kahve Bardağı)
Cookie (Kurabiye)
Donut
Grapes (Üzüm)
Hot Dog
Ice Cream (Dondurma)
Lollipop (Lolipop)
Mushroom (Mantar)
Peanut (Fıstık)
Pear (Armut)
Pineapple (Ananas)
Pizza
Potato (Patates)
Sandwich (Sandviç)
Steak (Biftek)
Strawberry (Çilek)
Watermelon (Karpuz)

Tablo 9. Makine Öğrenmesinde Kullanılan Kategoriler

2.2.4.2. Sunucu Taraflı Teknolojiler

Bir önceki bölümde belirtildiği gibi, yapay zekanın eğitiminde Quick Draw veriseti kullanılmıştır. Geliştirilmiş olan proje bir iOS uygulaması olduğu için, Apple tarafından geliştirilmiş olan Turi Create kütüphanesinden yararlanılarak yapay zeka eğitilmiştir.

- **Python:** Python, çok çeşitli uygulamalar için kullanılabilen popüler bir programlama dilidir.
- **Turi Create:** Turi Create, Apple tarafından geliştirilen açık kaynaklı bir kütüphanedir. Danışmanlı ve danışmansız makine öğrenimi modellerini oluşturmak için kullanılmaktadır. Turi Create; sınıflandırma, öneri sistemleri, nesne tespiti gibi modelleri oluşturmada kullanılabilmektedir. Geliştirilmiş olan projede resim sınıflandırılması kullanıldığı için bu kütüphaneden yararlanılmıştır.
  Turi Create ile oluşturulan model, CoreML'e dönüştürülerek iOS, macOS, tvOS ve watchOS uygulamalarında kullanılabilmektedir.
- **Core ML:** Core ML, Apple tarafından geliştirilen bir framework'tür. Geliştiricilerin makine öğrenimi modellerini uygulamalarında kullanabilmesini sağlar. Böylece uygulama üzerinde sınıflandırma, öneri sistemleri, nesne tespiti gibi özellikler kullanılabilir.

Geliştirilen projede Python kullanılarak veriler ön işlemden geçirilmiş, Turi Create ile model eğitilmiş ve son olarak Core ML ile birlikte uygulama içerisinde kullanmaya hazır hale getirilmiştir.

2.2.4.3. Mobil Taraflı Teknolojiler

Mobil tarafında, uygulamanın geliştirilmesi için Xcode ortamı kullanılmıştır. Xcode, Apple'ın macOS için geliştirdiği bir derleyici (IDE)'dir. Bu proje, iOS ve ipadOS'te kullanılabilecek şekilde tasarlanmıştır. Xcode'da birçok programlama diliyle uygulamalar geliştirilebilmektedir. Bu projede Swift programlama dili kullanılmıştır.

- **Xcode:** Xcode, Apple tarafından geliştirilen bir derleyicidir. Xcode kullanarak geliştiriciler macOS, iOS, ipadOS, watchOS ve tvOS için uygulamalar

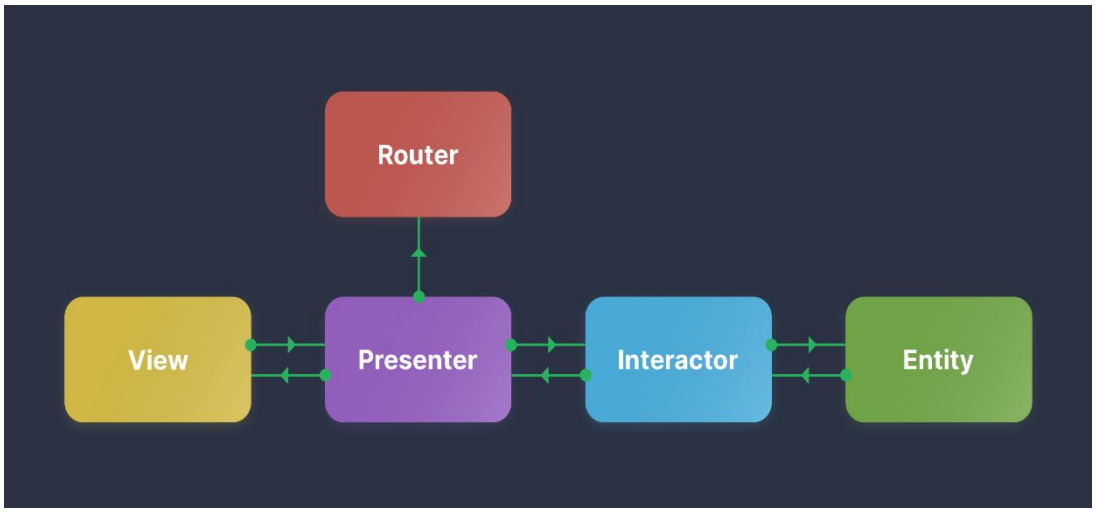
geliştirebilmektedir. Bu proje, uygulama iOS ve ipadOS'de kullanılabilecek şekilde tasarlanmıştır.

- **Swift:** Swift programlama dili, Apple tarafından geliştirilen açık kaynaklı bir programlama dilidir. Swift genel olarak iOS ve macOS geliştirilmek için kullanılır. iOS geliştirmek için kullanılabilecek en uygun programlama dillerinden birisidir.
- **Auto Layout:** Auto Layout, geliştirilen tasarımın birçok cihazda aynı görünüme sahip olmasını sağlamak için kullanılmaktadır. Birçok iPhone ve iPad cihazının bulunması ve bu cihazlarda tasarımın bozulmaması adına geliştirilen projede Auto Layout kullanılmıştır.
- **Cocoa Pods:** Cocoa Podlar, Xcode projeleri için kütüphane bağımlılıklarını yöneten bir araçtır. Projenin bağımlılıkları Podfile adı verilen tek bir metin dosyasında belirtilmektedir. Uygulama içerisinde birçok kütüphane kullanabilmemizi sağlamaktadır.
- **Countdown Label:** Geliştirilen uygulama içerisinde Coundown Label [11] adı verilen bir kütüphane kullanılmıştır. Uygulama içerisinde bulunan sayaç, Countdown Label kütüphanesi kullanılarak geliştirilmiştir. Dönüşüm animasyonu ve bazı yararlı işlevlere sahip basit bir geri sayım kütüphanesidir.
- **VIPER Tasarım Deseni:** VIPER, çoğunlukla iOS uygulamalarının geliştirilmesinde kullanılan bir mimarisel tasarım desenidir. Bu tasarım deseni singleton temellidir, yani uygulamaya eklenen her bir yeni özellik bir modül olarak oluşturulmaktadır. Yine her bir modülde VIPER'a göre farklı görevlere sahip beş sınıf oluşturulmaktadır. Her bir sınıf tek bir amaca yönelik olmalıdır. VIPER, iOS uygulamalarına yönelik Clean Architecture (Temiz Mimari) uygulaması olarak tanımlanabilmektedir.

  VIPER kelimesinin harfleri oluşturulacak beş sınıfın tanımlamasına denk gelmektedir. Bu sınıflar aşağıdaki gibidir:

  1. **View:** Kullanıcının gördüğü ve etkileşime geçtiği kısıma View adı verilir. View'dan alınan bilgiler Presenter'a aktarılır.
  2. **Interactor:** Temel mantık (logic) işlemler bu sınıf içerisinde yapılmaktadır. Presenter ile iletişim kurar ve Presenter'dan aldığı bilgileri mantıksal olarak işler. Kullanıcıya gösterilmesi gereken sonuuçlar Presenter'a tekrar yönlendirilir.

3. **Presenter**: Bütün bileşenleri birbirine bağlayan sınıftır. View'dan kullanıcı girişlerini alır ve Interactor'e yönlendirir. Interactor'de yapılan mantıksal işlemleri ise alır ve yine View'a yönlendirir. Ekranlar arası geçiş gibi işlemler ise View'dan alınan giriş ile beraber Presenter'a gelir, Presenter'dan Router'a yönlendirilerek geçiş sağlanır.
4. **Entity:** Bu sınıfta modüle ait temel model nesneleri bulunmaktadır. Entity yalnızca Interactor tarafından kullanılır.
5. **Router:** Router, uygulama içindeki geçişlerden sorumlu sınıftır. Modüller arası geçiş bu sınıftan yapılır.



Şekil 13. VIPER Tasarım Deseni Şeması [PixelMatters, [Tutorial] Part 1 — VIPER design pattern: what, when, why and how]

Geliştirilen uygulama, VIPER mimarisel tasarım deseni kullanılarak geliştirilmiş bir iOS uygulamasıdır. Uygulama içerisinde "**Home**" ve "**Drawing**" adında iki ayrı modül bulunmaktadır.

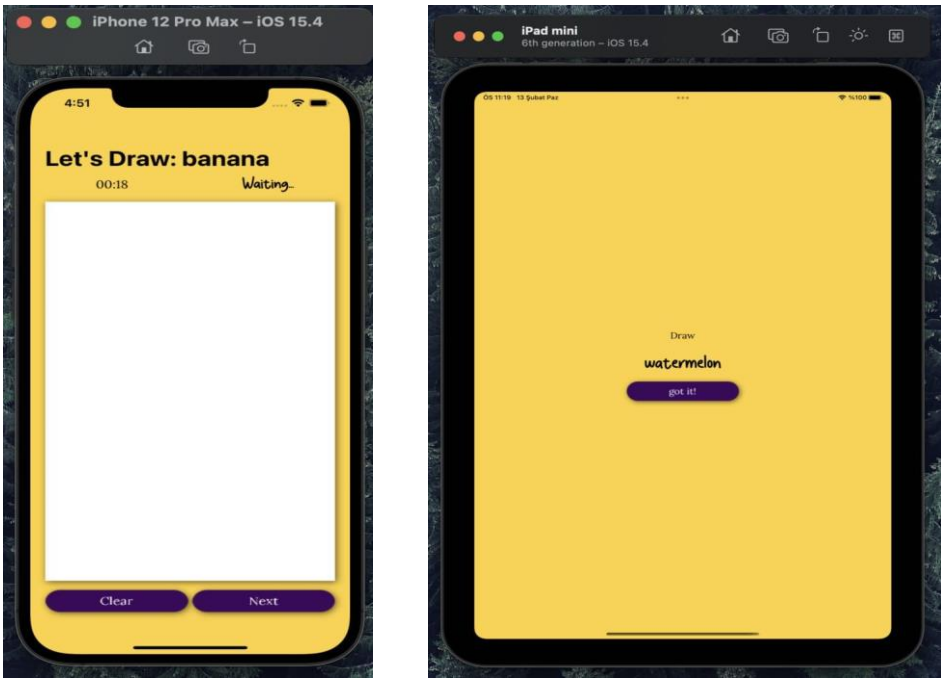
2.2.4.4. Uygulama Arayüzü Tasarımı

Tasarım aşamasında tasarlanan arayüz dikkate alınarak uygulama arayüzü aşağıdaki şekillerdeki gibi gerçekleştirilmiştir.

Uygulama aşağıdaki ekran ile açılmaktadır.

Şekil 14. Uygulama Açılış Ekranı (Splash Screen)

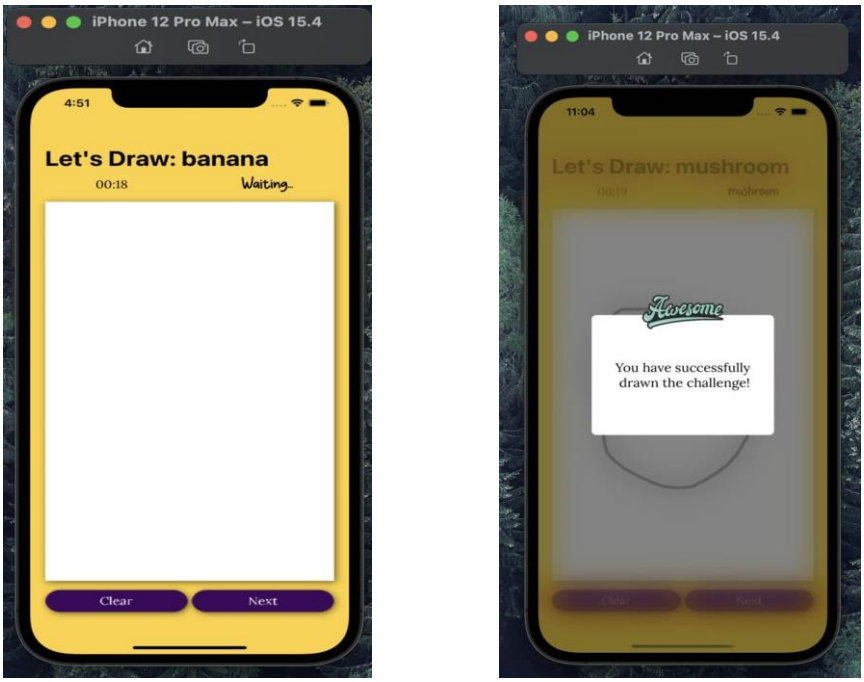
Açılış ekranından sonra kullanıcının çizmesi gereken kelime, yapay zekanın eğitildiği kelimeler arasından birisi seçilerek ekranda gösterilmektedir. Telefon olarak iPhone 12 Pro Max, tablet olarak iPad mini aşağıda gösterilmiştir.



Şekil 15. Uygulama Kelime Ekranı

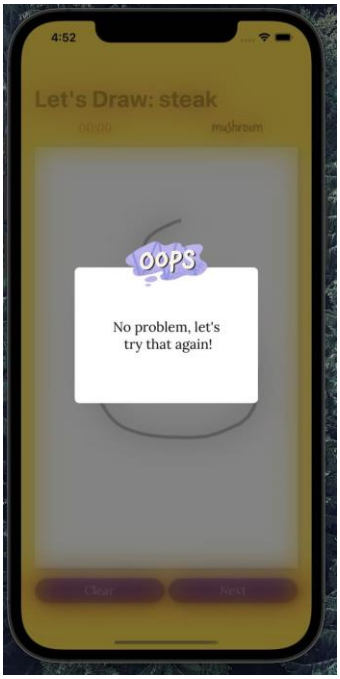
Kullanıcı “Got It (Anladım)” butonuna bastıktan sonra, kullanıcının resim çizebileceği ekran gösterilmektedir. Kullanıcı resim çizme başlamadan önce “Waiting (Bekleniyor)” gösterilmekte, kullanıcı çizime başladıktan sonra ise bu çizim yapay zeka eğitimine gönderilip kullanıcının kelimeyi doğru çizip çizmediği tahmin edilmektedir.

Anlık olarak tespit edebilmek için, iOS’un dokunma başlangıcı ve dokunma bitişi fonksiyonlarından yararlanılmıştır. Kullanıcı ekrandan her elini kaldırdığında resim tahmin edilmektedir. Aşağıdaki görsellerde, iPhone simülatöründe doğru tahmin edilmiş bir örnek sunulmuştur.



Şekil 16. Uygulama Çizim Ekranı

Aşağıdaki görselde ise süre dolduktan sonra kullanıcı çizimi doğru çizemediyse gösterilen ekran gösterilmektedir.



Şekil 17. Uygulama Yanlış Çizim Ekranı

2.2.5. Test Aşaması

Yazılım test aşaması, gerçek yazılım ürününün beklenen gereksinimleri karşılayıp karşılamadığını kontrol etmek ve yazılım ürününün kusursuz olduğundan emin olmak için gerçekleştirilen aşamadır. Bu aşamada, ilgilenilen bir veya daha fazla özelliği değerlendirmek için manuel veya otomatik araçlar kullanılarak yazılım bileşenleri test edilir.

Yazılım testinin amacı; hataları veya eksik gereksinimleri belirlemektir.

2.2.5.1. Genel Test Planı

Genel test planı, ürün testinin detaylarını içermektedir. Genel test planları test ile ilgili yapılacak işlerden oluşmaktadır.

2.2.5.2. Test Sonuç Raporları

Uygulama içerisinde kullanıcı resmini çizmeyi yarım bırakıp ileri butonuna basarak yeni bir kelimeye geçebilir. Kullanıcının süresi dolduğunda ise kullanıcıya başarısız ekranı gösterilmektedir. Bu durumda, kullanıcı resim çizmeyi yarım bırakıp ileri butonuna bastığında, başarısız ekranının gösterilip gösterilmediği test edilebilir. Test yapıldıktan sonra sonuçlar aşağıdaki tabloda gösterilmiştir.

TEST DURUMU FORMU		
GENEL BAKIŞ		
Test Durumu Numarası	**Test Durum Adı**	
1	Yeni kelimeye geçişin doğruluğunun kontrol edilmesi	
TEST EYLEMLERİ		
ÖN KOŞULLAR		
Resim çizme ekranında, ileri butonuna basıldığında kullanıcının önceden kalan çizimine ait başarısız popup'ı gösterilmemesi gerekmektedir.		
TEST ADIMLARI		
1- Çizim ekranına girilir. 2- Çizim ekranında resim çizmeye başlanır. 3- Süre dolmadan ileri butonuna basılır. 4- Kelime ekranında beklenir. 5- Kelime ekranında geçmiş kelimeye ait başarısız popup'ının gösterilmediğinden emin olunur.		
TEST SONUÇLARI		
BEKLENEN SONUÇ	**GERÇEKLEŞEN SONUÇ**	**SONUÇ**
Geçmiş kelimeye ait popup gösterilmemesi.	Geçmiş kelimeye ait popup gösterilmedi.	**Geçti**

Tablo 10. Test Durumu Formu - 1

Uygulama içerisinde kelime ekranına rastgele kelime gelip gelmediği kontrol edilmelidir. Test yapıldıktan sonra sonuçlar aşağıdaki tabloda gösterilmiştir.

TEST DURUMU FORMU		
GENEL BAKIŞ		
Test Durumu Numarası	**Test Durum Adı**	
2	Kelime ekranında kelimelerin rastgele gelip gelmediğinin kontrol edilmesi.	
TEST EYLEMLERİ		
ÖN KOŞULLAR		
Kullanıcı her kelimesinden sonra rastgele yeni bir kelimeye geçmelidir.		
TEST ADIMLARI		
1- Kelime ekranında ilk kelimeye bakılır. 2- Çizim ekranına geçilir. 3- Çizim ekranında ileri butonuna basılır. 4- Önceki kelimeden farklı bir kelime gelip gelmediği kontrol edilir.		
TEST SONUÇLARI		
BEKLENEN SONUÇ	**GERÇEKLEŞEN SONUÇ**	**SONUÇ**
Her bir kelime sonrası rastgele kelimeye geçilmesi.	Her seferinde rastgele bir kelime gösterildi.	**Geçti**

Tablo 11. Test Durumu Formu – 2

Uygulama içerisinde kullanıcının çizmesi gereken kelime hem kelime ekranında hem de çizim ekranında bulunmaktadır. Her iki ekranda da aynı kelimenin gösterilip gösterilmediği test edilmelidir. Test yapıldıktan sonra sonuçlar aşağıdaki tabloda gösterilmiştir.

TEST DURUMU FORMU	
GENEL BAKIŞ	
Test Durumu Numarası	**Test Durum Adı**
3	Hem kelime hem de çizim ekranında gösterilen kelimenin aynı olup olmadığının kontrolü
TEST EYLEMLERİ	
ÖN KOŞULLAR	

<table>
<tr><td colspan="3">Hem kelime hem çizim ekranında gösterilen kelimenin aynı olması gerekmektedir.</td></tr>
<tr><td colspan="3">TEST ADIMLARI</td></tr>
<tr><td colspan="3">1- Kelime ekranında gösterilen kelimeye bakılır.
2- Çizim ekranına geçilir.
3- Çizim ekranında gösterilen kelimeye bakılır.
4- Kelime ekranı ve çizim ekranında gösterilen kelimeler karşılaştırılır.</td></tr>
<tr><td colspan="3">TEST SONUÇLARI</td></tr>
<tr><td>BEKLENEN SONUÇ</td><td>GERÇEKLEŞEN SONUÇ</td><td>SONUÇ</td></tr>
<tr><td>Her iki ekranda da aynı kelimenin gösterilmesi.</td><td>Her iki ekranda da aynı kelime gösterilmektedir.</td><td>Geçti</td></tr>
</table>

Tablo 12. Test Durumu Formu – 3

3. SONUÇ

Bu projede, yapay zeka teknikleri kullanılarak eğitilmiş el çizimi tanıma uygulaması geliştirilmiştir. Birçok uygulama yazı yazma ve resim çizmeye olanak tanımakta fakat çok az uygulama yapay zeka ile çizimlerin ve yazıların tanımlanması özelliğini kullanmaktadır. Mobil ve tablet cihazlarında el çizimi ile not alma, resim çizme gibi aktivitelerin oldukça yaygınlaştığı bu zamanlarda geliştirilen bu uygulama sayesinde kullanıcıların çizdiği resimlerin yapay zeka ile algılanıp bu bilgilerin kullanılabileceği sonucuna ulaşılmıştır.

4. KAYNAKLAR

1. Google Creative Lab, Quick Draw Dataset Dokümantasyonu, Github.
2. Yi Li, Yi-Zhe Song, Shaogang Gong, Sketch Recognition by Ensemble Matching of Structured Features, 2013.
3. François Chollet, Python ile Derin Öğrenme, Buzdağı Yayıncılık, Ankara, 2019.
4. İzmir Katip Çelebi Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, Yapay Zeka ve Sağlık Uygulamaları, 2020.
5. Prof. Dr. Ercan Öztemel, Yapay Sinir Ağları, Papatya Yayıncılık, Ankara, 2012.
6. Dinesh Kumawat, 7 Types of Activation Functions in Neural Network, 2019.
7. Necmettin Çarkacı, Derin Öğrenme Uygulamalarında Başarım İyileştirme Yöntemleri (Regularization), 2017.
8. Dr. Erhan Sarıdoğan, Yazılım Mühendisliği Temelleri, Papatya Yayıncılık, Ankara, 2016.
9. Deniz Kılınç, Yazılım Yaşam Döngüsü Temel Aşamaları (Software Development Life Cycle Core Processes), 2016.
10. Cubicl, Gantt Çizelgesi Nedir?
11. Suzuki Keishi, Countdown Label, Cocoapod, Eylül 2018.

EKLER

EK-1 Standartlar ve Kısıtlar Formu

EK-1 Standartlar ve Kısıtlar Formu

Projenin hazırlanmasında uyulan standart ve kısıtlarla ilgili olarak, aşağıdaki soruları cevaplayınız.

1. Projenizin tasarım boyutu nedir? (Yeni bir proje midir? Var olan bir projenin tekrarı mıdır? Bir projenin parçası mıdır? Sizin tasarımınız proje toplamının yüzde olarak ne kadarını oluşturmaktadır?)

Quick Draw oyunundan esinlenerek geliştirdiğim bir mobil projesidir. Tasarımlar tamamen bana ait olmak ile Quick Draw projesinin mobile uyarlanmış halidir.

2. Projenizde bir mühendislik problemini kendiniz formüle edip, çözdünüz mü? Açıklayınız.

Geliştirilen proje kapsamlı bir yapay zeka ve mobil geliştirme bilgisi gerektirmektedir.

3. Önceki derslerde edindiğiniz hangi bilgi ve becerileri kullandınız?

Programlamaya Giriş ve Algoritma, Yazılım Gereksinimi Mühendisliği, Nesne Yönelimli Programlama, Yazılım Tasarımı ve Mimarisi, Yapay Zeka.

4. Kullandığınız veya dikkate aldığınız mühendislik standartları nelerdir? (Proje konunuzla ilgili olarak kullandığınız ve kullanılması gereken standartları burada kod ve isimleri ile sıralayınız).

ISO/IEC 25051 - Yazılım Sitemleri Kalite Gereksinimleri Ve Değerlendirme ISO/IEC 15504 - Geliştirilen Yazılımların Süreçlerinin Kalitesini İyileştirmek ISO/IEC15504 - Yazılım Yaşam Döngüsü Süreçlerinin Kullanılması Ve İyileştirilmesi

5. Kullandığınız veya dikkate aldığınız gerçekçi kısıtlar nelerdir? Lütfen boşlukları uygun yanıtlarla doldurunuz.

a) Ekonomi

Geliştirilen proje bir mobil projesi olup herhangi ekonomik kısıta sahip değildir.

b) Çevre sorunları:

Geliştirilen projenin çevreye herhangi bir zararı yoktur.

c) Sürdürülebilirlik:

Geliştirilen proje farklı kelimeler ve farklı konseptler uygulanarak oldukça sürdürülebilirdir.

d) Üretilebilirlik:

Geliştirilen projenin çalıştırılmasında herhangi bir problem olmadığı gözlemlendiği için üretilebilirdir.

e) Etik:

Geliştirilen proje etik olarak bir problem oluşturmamaktadır.

f) Sağlık:

Geliştirilen projenin sağlık açısından herhangi bir sakıncası bulunmamaktadır.

g) Güvenlik:

Geliştirilen projede kullanıcı bilgisi bulunmadığı ve küçük bir oyun gibi tasarlandığı için güvenlik açısından hiçbir kısıtı bulunmamaktadır.

h) Sosyal ve politik sorunlar:

Geliştirilen proje sosyal veya politik açıdan bir problem teşkil etmemektedir.