Mühendislikte Bilgisayar Yazılımı Uygulamaları

Editör
Eyyüp GÜLBANDILAR

BİDGE

BİDGE Yayınları

Mühendislikte Bilgisayar Yazılımı Uygulamaları

Editör: Prof. Dr. Eyyüp GÜLBANDILAR

ISBN: 978-625-372-182-4

1. Baskı
Sayfa Düzeni: Gözde YÜCEL
Yayınlama Tarihi: 25.06.2024
BİDGE Yayınları

Sertifika No: 71374
Yayın hakları © BİDGE Yayınları
www.bidgeyayinlari.com.tr - bidgeyayinlari@gmail.com
Krc Bilişim Ticaret ve Organizasyon Ltd. Şti.
Güzeltepe Mahallesi Abidin Daver Sokak Sefer Apartmanı No: 7/9 Çankaya / Ankara

İÇİNDEKİLER

BÖLÜM I

İçme Suyu Şebekelerinin MQTT Protokolü ile Uzaktan Yönetimi

Meltem TEKELİ AKDAĞ[1]
Mehmet Fatih DEMİRAL[2]
Ali Hakan IŞIK[3]

GİRİŞ

İçme suyu şebekelerinin etkin bir şekilde yönetilmesi, su kaynaklarının sürdürülebilirliği ve su güvenliği açısından büyük önem taşımaktadır. Geleneksel yöntemlerle yapılan manuel takip ve kontrol süreçleri, zaman alıcı, maliyetli ve hata eğilimli olabilmektedir. Bu nedenle, gelişmiş teknolojilerin kullanılmasıyla içme suyu şebekelerinin uzaktan takip ve kontrolü, verimliliği

[1]Burdur Mehmet Akif Ersoy Üniversitesi, Fen Bilimleri Enstitüsü, Burdur, Türkiye, Orcid: 0000-0003-0946-4242
[2] Burdur Mehmet Akif Ersoy Üniversitesi, Mühendislik-Mimarlık Fakültesi, Endüstri Mühendisliği Bölümü, Burdur, Türkiye, Orcid: 0000-0003-0742-0633
[3] Burdur Mehmet Akif Ersoy Üniversitesi, Mühendislik-Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümü, Burdur, Türkiye, Orcid: 0000-0003-3561-9375

artırarak daha etkili bir şekilde gerçekleştirilebilir. Bu makalede, Nesnelerin İnterneti (Internet of Things- IoT. Makalenin kalanında IoT olarak anılacaktır.) sistemlerinde önemli bir protokol olan MQTT (Message Queuing Telemetry Transport) protokolünün içme suyu şebekelerinde kullanılabilme durumu incelenmiştir. MQTT; hafif, güvenilir, esnek ve düşük bant genişliği gerektiren IoT uygulamaları için ideal bir iletişim protokolüdür. İçme suyu şebekelerinde kullanıldığında, MQTT protokolü sayesinde sensörlerden alınan verilerin anlık olarak izlenmesi, analiz edilmesi ve gerektiğinde müdahalede bulunulması mümkün hale gelir. Makalenin ilerleyen bölümlerinde, MQTT protokolünün içme suyu şebekelerinde nasıl kullanıldığı, uygulama alanları ve getirdiği avantajlar detaylı bir şekilde açıklanmıştır. Ayrıca, örnek bir uygulama ile MQTT protokolünün etkinliği ve verimliliği de değerlendirilmiştir. Örnek uygulamada, MQTT ile içme suyu şebekesinden alınan veriler internet ortamına aktarılmış, bir arayüz ile zaman ve mekân kısıtlaması olmadan erişime açılmıştır. SMS ve mail bilgilendirme sistemleri ile ilgililere anlık bildirim göndererek içme suyundaki tüm değişikliklerin hızla takip edilmesi sağlanarak içme suyu şebekelerinin kalitesinin korunması ve olası halk sağlığı sorunlarının önlenmesi hedeflenmiştir. Sonuç olarak, içme suyu şebekelerinin MQTT protokolü ile uzaktan takip ve kontrolü, su kaynaklarının daha etkin yönetilmesini sağlayarak su güvenliği ve sürdürülebilirliğe katkıda bulunabilir.

Son yıllarda içme suyu şebekelerinin uzaktan yönetiminin önemini vurgulayan çok sayıda çalışma yapılmıştır. Karadirek vd. (2012) Antalya'da Hurma Depo'ya SCADA sistemi kurarak saha ve laboratuvar ölçümlerini karşılaştırmış ve verimlilik analizi

yapmıştır. Uzaktan izleme ile elde edilen verilerin ölçüm değerleri ile uyumlu olduğu ve SCADA sisteminin verimli bir yöntem olduğu sonucuna ulaşmışlardır. Uçaner ve Özdemir (2013) ise Amerika'da bir örnek şebekede ek klorlama istasyonları kurulumu için Genetik Algoritmalar (GA) ve bilgisayar programı kullanarak strateji geliştirmiştir. Farklı çözüm uzayları deneyerek kullanılan toplam klor miktarının düşürülebileceğini göstermişlerdir.

Özkaya vd. (2016) PIC mikrodenetleyici kullanarak GPRS haberleşme sağlayan bir sistem kurmuş ve her türlü hava şartında şebekelere uzaktan erişme imkanı sağlamışlardır. Daldal (2018) ise Ankara Kızılcahamam'da bilgisayarlı otomasyon sistemi kurarak elektrik tasarrufu, depo taşmaları ve su sarfiyatının önlenmesi gibi hedeflere ulaşmıştır. Mohanasundaram vd. (2018) Hindistan'da akıllı su şebekesi sistemi kurarak son kullanıcı şebekeleri üzerinden su faturalarını ve suyun kalitesini uzaktan yönetebilmişlerdir. Akdeniz ve Muhammetoğlu (2019) Antalya Yeşilbayır şebekesini EPANET modeli ile modellemiş ve SCADA sistemi ile uzaktan takibini sağlamıştır. Kurban vd. (2020) ise İç Anadolu Bölgesi'nde bir belediyede SCADA ile web tabanlı uzaktan izleme sistemi kurarak kuyulardaki pompaların hidrolik analizlerini yapabilmişlerdir. Son olarak Ancy Jenifer vd. (2022) bir su kulesinin MQTT protokolü kullanılarak uzaktan yönetimini sağlamışlardır.

Farklı araştırmacılar tarafından farklı yöntemler kullanılsa da, tüm çalışmalar içme suyu şebekelerinin uzaktan takip ve yönetiminin önemini vurgulamaktadır. Uzaktan takip sistemlerinin gelişmesi ile birlikte yöntemlerin çeşitlendiği ve daha verimli yöntemlere geçiş olduğu görülmektedir. Bu sistemlerin kullanımıyla su kaynaklarının korunması, şebeke arızalarının önceden tespit

edilmesi ve su kayıplarının azaltılması gibi birçok fayda sağlanabilir. Son yıllarda, internetin ve akıllı sensör teknolojisinin gelişimi, içme suyu şebekelerinin uzaktan takibi ve kontrolü için yeni fırsatlar sunmuştur. Bu bağlamda, MQTT protokolü düşük kaynak tüketimi, hafiflik ve kolay entegrasyon gibi avantajları sayesinde diğer IoT sistemlerinde olduğu gibi içme suyu şebekelerindeki veri iletişimi ve kontrol süreçlerinde de önemli bir rol oynamaktadır.

IoT, internet üzerinden diğer cihaz ve sistemlerle veri bağlantısı ve paylaşımı amacıyla sensörler, yazılımlar ve diğer cihazlar kullanılarak oluşturulan bir iletişim ağıdır. IoT'nin temel prensipleri şunlardır:

- Bağlantı: IoT, nesnelerin internete bağlanabilmesini sağlar. Bu sayede nesneler, diğer cihazlarla iletişim kurabilir ve veri alışverişi yapabilir.
- Sensörler: IoT, nesnelerin çevresindeki verileri toplamak için sensörler kullanır. Sensörler, çeşitli parametreleri ölçerek veri üretir ve bu verileri diğer cihazlarla paylaşır.
- Veri Paylaşımı: IoT, nesnelerin ürettiği verileri diğer cihazlarla paylaşmayı sağlar. Bu sayede farklı cihazlar arasında veri alışverişi gerçekleştirilebilir ve bu verilerin analizi yapılarak farklı uygulamalar geliştirilebilir.
- Uzaktan Kontrol: IoT, nesnelerin uzaktan kontrol edilmesini sağlar. Bu sayede nesneler, başka cihazlar tarafından kontrol edilebilir ve istenen işlemler gerçekleştirilebilir.

IoT sistemi tasarlanırken sistemin gereksinimlerine ve çevre koşullarına göre çeşitli protokollerden biri tercih edilebilir. En yaygın kullanılan protokoller şunlardır;

- MQTT (Message Queuing Telemetry Transport): MQTT, IoT cihazlarının hafif ve güvenilir bir şekilde iletişim kurmasını sağlayan bir mesajlaşma protokolüdür. Bu protokol, düşük bant genişliği ve düşük pil tüketimi gerektiren cihazlar için idealdir. Genellikle sensör verilerinin iletilmesi için tercih edilir.
- CoAP (Constrained Application Protocol): CoAP, IoT cihazlarının enerji verimliliği ve düşük bant genişliği kullanımı için tasarlanmış bir iletişim protokolüdür. HTTP'ye benzer bir yapıya sahiptir ancak daha az kaynak tüketir. Bu nedenle, IoT cihazlarının internet üzerinden iletişim kurarken kullandığı bir diğer önemli protokoldür.
- Bluetooth: Bluetooth, kısa mesafeli kablosuz iletişim için yaygın olarak kullanılan bir teknolojidir. IoT cihazları arasında veri transferi için sıkça tercih edilir. Özellikle akıllı ev cihazları ve giyilebilir teknolojiler arasında Bluetooth tabanlı iletişim oldukça yaygındır.
- Zigbee: Zigbee, düşük güç tüketimi, düşük veri hızı ve kısa mesafeli kablosuz iletişim için tasarlanmış bir standarttır. Bu protokol, IoT cihazları arasında düşük güç tüketimi gerektiren uygulamalarda kullanılır.

Örneğin, akıllı aydınlatma sistemleri ve ev otomasyonu için idealdir.

- HTTP (Hypertext Transfer Protocol): internet bağlantılı sistemlerde en yaygın kullanılan protokoldür. ıot cihazları arasında iletişim sağlamak için kullanılabilir, ancak yüksek güç tüketimi gerektirir.

Her protokolün kendine göre avantaj ve dezavantajları vardır. MQTT ve COAP az güç tüketirken HTTP yüksek güç tüketimi gerektirir ancak HTTPS ile desteklendiğinde daha güvenlidir. Zigbee ve Bluetooth kısa mesafede çok iyi sonuç çıkarabilir ancak Bluetooth yüksek enerji tüketir, Zigbee yüksek veri trafiği durumunda performans sorunları yaşayabilir. MQTT, Bluetooth ve HTTP daha yaygın kullanıldığı için geniş bir kullanıcı tabanına sahiptir. Bütün bunlar göz önünde bulundurularak; bu çalışmada uzun mesafeler arası iletişim ihtiyacı, düşük güç tüketimi gerektirmesi ve entegrasyonu kolay olmasından dolayı MQTT protokolü tercih edilmiştir.

MQTT, bir yayın-abone modeli kullanır. Bu modelde, bir MQTT sunucusuna (broker) bağlı cihazlar, belirli konular (topics) üzerinde yayın yapabilir (publish) veya bu konuları dinleyebilir. Bir cihazın yayın yaptığı konuyu dinlemek isteyen diğer cihazlar, bu konuya abone olarak (subscribe) ilgili veriyi alabilirler. Bu sayede, cihazlar arasında veri akışı sağlanır.

Veri akışının kesintisiz olması için dikkat edilmesi gereken MQTT'nin temel kavramları şunlardır: (Soni ve Makwana, 2017)

- Yayınla/Abone Ol: MQTT protokolünde, yayıncılar mesaj yayınlar ve kullanıcılar ilgilendikleri konulara abone olur. Bu, bir Yayınla/Abone Ol modeli olarak kabul edilir. Aboneler, ilgilendikleri konulara abone olarak, o konulara yayınlanan her mesajı alırlar. Öte yandan, istemciler konulara mesaj yayınlayabilir, böylece tüm aboneler o konuların mesajlarına erişebilir.
- Konular ve Abonelikler: MQTT'de, yayıncılar mesajları konulara yayınlar, bunlar mesajın konusu olarak kabul edilir. Aboneler, böylece belirli mesajları almak için konulara abone olurlar. Konu abonelikleri, alınan verileri belirli bir konuyla sınırlayan ifadelerle belirtilebilir.
- Servis Kalitesi Seviyeleri: Bu protokol, bir mesajın teslimat garantisi ile ilgili olarak bir mesajın iki tarafı arasında bir anlaşma olan Servis Kalitesi (QoS) seviyelerini tanımlar. Üç seviye QoS desteği sağlar, bunlar aşağıda açıklanmıştır.
    - QoS-0 (En fazla bir kez): Bu servis kalitesi seviyesinde, mesaj en fazla bir kez gönderilir ve mesajın teslimatını garanti etmez.
    - QoS-1 (En az bir kez): Bu servis kalitesi seviyesinde, veri en az bir kez gönderilir ve bir mesajın birden fazla kez teslim edilmesi mümkündür.
    - QoS-2 (Tam olarak bir kez): Bu servis kalitesi seviyesinde, mesaj tam olarak bir kez gönderilir, bunun için 4 yönlü el sıkışma kullanılır.
- Saklanan Mesajlar: MQTT'de, mesajlar brokerda tüm mevcut istemcilere dağıtıldıktan sonra saklanır. Aynı konu için yeni bir abonelik alındığında, o konunun saklanan mesajları yeni istemciye gönderilir.
- Temiz Oturumlar ve Güvenilir Bağlantılar: Bir abone brokerla bağlantı kurduğunda, temiz oturum bağlantısı kalıcı olarak kabul

edilir. Bu durumda, en yüksek QoS atamasına sahip ardışık mesajlar, bağlantı yeniden kurulduğunda teslim edilmek üzere saklanır. Bu bayrağın kullanımı isteğe bağlıdır.

- Vasiyetler: Bir istemci, broker'a bir vasiyeti (mesajı) olduğunu bildirebilir, bu da beklenmedik bir şekilde bağlantının kesilmesi durumunda belirli bir konu veya konulara dağıtılması gerektiğini belirtir. Bu vasiyet, özellikle yöneticilerin bir sensörün sisteme bağlantısını kaybettiği anda haberdar edilmesi gereken güvenlik veya alarm sistemleri gibi sistemlerde çok yararlıdır.

MATERYAL VE YÖNTEM

Bu makaleye konu olan örnek çalışma; su kontrol cihazları ve dozaj pompaları aracılığı ile içme suyu şebekelerinin ve/veya yüzme havuzlarının internet ortamından yönetilmesini mümkün kılmaktadır.

Sistem Gereksinimleri

- Verileri okuyabilmek için sensörler (pH , ORP, sıcaklık, nem vb)
- Cihazdan alınan verileri aktarmak için haberleşme modülü (Wi- Fi bağlantısına sahip bir işlemci ile kontrol edilebilir. – Bu çalışmada ESP32-S3-WROOM-1 N16R2 tercih edilmiştir.)
- Verileri aktarabilmek için aktif internet bağlantısı
- MQTT broker kurulumu için sunucu (Bu çalışmada Raspberry Pi 4 8GB – Model 4B üzerine Mosquitto (Light, 2017) kurulumu tercih edilmiştir.)
- Veritabanı (Bu çalışmada MySQL tercih edilmiştir.)

- API hizmetlerini çalıştırabilmek için sunucu (Bu çalışmada fiziksel sunucu tercih edilmiştir.)
- Arayüz (web sitesi & mobil uygulama)

Sistem Mimarisi

Şekil 1'de sistem mimarisi verilmiş olan örnek çalışmanın çalışma prensibi şu şekildedir:

Su kontrol cihazlarına bağlı olarak çalışan pH, ORP, sıcaklık sensörlerinden gelen veriler cihaz tarafından işleniyor ve modbus aracılığı ile ESP modüllerine iletiliyor. ESP modülünde JSON verisine dönüştürülerek ve şifrelenerek MQTT protokolü ile MQTT broker'a iletiliyor ve bu veriler ilgili konuda yayınlanıyor. Her cihaz için standart olarak "cihaz seri numarası/kurulum anında cihaza verilen isim" konu olarak kullanılıyor. Geriye dönük rapor oluşturabilmek ve grafik görüntüleyebilmek için Node.js ile yazılmış olan ve fiziksel bir sunucuda kesintisiz çalışan bir uygulama aracılığı ile her konu için veritabanında ilgili tabloya kayıt yapılıyor. Amacına göre özelleştirilmiş Node.js uygulamaları, veritabanından ilgili veriyi çekiyor ve şifre çözme algoritmaları sonrası bu verileri kullanabiliyor.

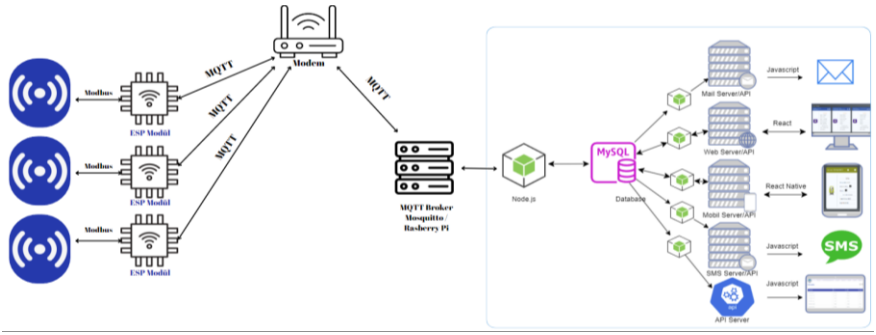
Cihazda alarm olması durumunda (değerlerin belirlenen sınırlar dışına çıkması, akış olmaması gibi durumlarda) Mail API aracılığı ile ilgili cihazın kullanıcısına mail göndererek; SMS API aracılığı ile de kullanıcının telefon numarasına SMS göndererek bildirim sağlanıyor. SMS gönderebilmek için OTP mesaj hizmeti sunan özel bir şirketin API hizmetinden faydalanılıyor.

API Server ile kullanıcılara, cihaz verilerini kendi sistemlerine entegre edebilme imkânı verilmektedir. Bir büyükşehir belediyesine ait olan yüzme havuzlarına ait pH, sıcaklık ve klor verileri anlık olarak ilgili belediyenin web sayfası üzerinden halka açık olarak sunulabilmektedir.

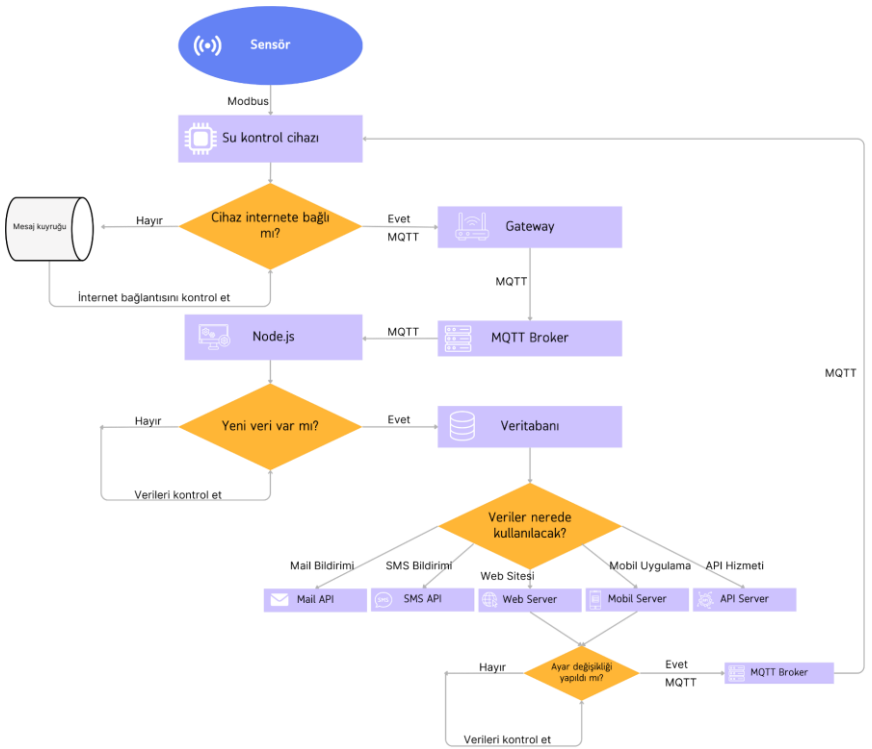
Web Server ile web sitesi üzerinden bir kullanıcı arayüzü hizmeti sunuluyor. Bu kullanıcı arayüzünde; admin paneli ile sahip olduğu tüm cihazları görüntüleyebilme, cihaz verilerini görüntüleyebilme, cihaz verilerinde değişiklik yapabilme, cihazlara ait raporlama ve grafikleri görüntüleyebilme imkânı bulunuyor.

Mobil Server aracılığı ile kullanıcılar web sitesindeki işlemleri bireysel cep telefonlarına Android ve IOS tabanlı uygulamaları yükleyerek yapabilmekteler.

Web server ve mobil server üzerindeki işlemler için MQTT protokolünün çift taraflı veri aktarımı avantajı kullanılıyor. Her istemci aynı zamanda hem abone hem de yayıncı olabilir. Cihaz ayarlarında yapılan değişiklikler şifrelenerek ilgili cihaza ait konunun ayar alt konusunda yayınlanıyor. Her ESP modülü kendine ait konunun ayar alt konusuna abone olduğu için ilgili mesajı alarak şifresini çözüp yine modbus ile ilgili cihaza iletiyor. İstenilen değişiklik cihaza uygulanmış oluyor. Şekil 2’de bu çalışmanın akış diyagramı görülebilir.



***Şekil 1**. Örnek Çalışmanın Sistem Mimarisi*



***Şekil 2.** Örnek Çalışmanın Akış Diyagramı*

Sistem Güvenliği

IoT sistemlerinde en önemli konu güvenliktir. MQTT ise yapısı gereği güvenlik için ekstra önlemler almayı gerektirmektedir. Güvenliği sağlayabilmek için bu sistemde kullanılan güvenlik önlemleri;

- Tüm ESP modüllerine gömülü, uzaktan güncellenebilir SSL sertifikası
- MQTT Broker üzerinde SSL sertifikası
- MQTT Broker'a kullanıcı adı ve şifre ile bağlanma
- API hizmetlerinin çalıştığı sunucuda SSL sertifikası
- Web sitesinde SSL sertifikası ile https koruması
- Şifreleme yöntemi ile veri aktarımı

BULGULAR VE TARTIŞMA

Bu çalışmada bir belediyeye ait içme suyu depoları, farklı bir belediyeye ait havuzlar ve farklı otellere bağlı yüzme havuzları sistemin etkinliğini görebilmek için demo cihazlar olarak sisteme bağlanmıştır. Her bir cihaz kendi kullanıcıları tarafından uzaktan yönetilmiştir.

Sistem kurulumunda bazı sorunlar ile karşılaşılmıştır.

Belediye gibi kurumsal tesislerde güvenlik sebebi ile verileri dışarı aktarmak mümkün olmayabiliyor. Bunun için ilgili belediyelerin Bilgi İşlem departmanları ile ortak çalışılarak uygun bir yöntemde karar kılınmıştır. Her cihazın mac adresi ve IP adresleri sisteme tanıtılarak sadece bu cihazlar için veri aktarımına izin verilmiş, böylece sorun çözüme kavuşmuştur.

Her cihaza ait verilerin geriye dönük raporlanabilmesi için her gelen veri, veritabanına kayıt edilmişti, bu da ortalama 13 saniyede 1 veriye denk gelmektedir. Ancak cihaz sayısı arttıkça bu yöntem veritabanında çok yüksek boyutlarda veri tutulmasına ve sistemi yormasına sebep oldu. Bu nedenle veri tutma sıklığı 10 dakikaya çıkarıldı. Arayüzde anlık olarak tüm değişiklikler takip edilebilecek ancak geriye dönük raporlar 10 dakikalık aralıklarla sunulacak şekilde düzenleme yapılmıştır.

Cihaz üzerinde yapılabilen her değişikliğin uzaktan yapılmasına imkan verilmişti ancak bu da yetkin olmayan kullanıcılar tarafından yapıldığında ve arayüz üzerinden bu işlemleri yapmak kolay olduğu için cihazların ayarlarının bilinçsiz değiştirilmesine zemin hazırladı. Bu değişiklikler cihaz performanslarını etkiledi. Böylece uzaktan yapılabilecek cihaz ayarlarına kısıtlamalar getirilmiştir.

Süreç boyunca karşılaşılan sorunlar çözüldükten sonra sistem verimli bir şekilde işlemeye başlamıştır. Bu sistem sayesinde bir içme suyu deposunun klor seviyesinin düştüğü hızlıca tespit edilip önlem alınmıştır. Böylece halk sağlığı açısından önemli bir adım atılmıştır. Aynı zamanda belediye havuzlarının pH, klor ve sıcaklık seviyeleri canlı olarak halka açılmış ve şeffaf belediyecilik alanında kazanım sağlanmıştır. Uzak mesafelerdeki cihazları yöneterek zaman ve maliyet açısından da tasarruf sağlanmıştır.

Bu çalışma, MQTT protokolü aracılığıyla verilerin internet ortamına aktarılması yoluyla içme suyu şebekelerinin tüm katmanlarının daha kolay, daha ulaşılabilir ve daha verimli bir şekilde yönetilebileceğini kanıtlamaktadır. Bu, literatürdeki diğer

içme suyu yönetimi çalışmaları ile karşılaştırıldığında önemli bir katkıda bulunmaktadır.

Önceki çalışmalar belirli bir alana odaklanmıştır. Örneğin, Jenifer vd. (2020) su kulelerindeki su seviyelerini izlemek için bir IoT sistemi geliştirmişlerdir. Mohanasundaram vd. (2018) ise su sayaçlarının uzaktan okunması için bir sistem önermişlerdir.

Bu çalışmanın önemli bir gücü, bir içme suyu şebekesinin her aşamasına uygulanabilecek şekilde geliştirilmiş olmasıdır. Bu sayede, su kaynaklarından şebekeye dağıtımına kadar tüm süreç tek bir sistem üzerinden takip ve kontrol edilebilir. Bu da daha geniş bir kapsam sunmakta ve içme suyu şebekelerinin daha etkin yönetilmesine katkıda bulunmaktadır. Ayrıca, MQTT protokolünün kullanımı, sistemin ölçeklenebilirliğini ve esnekliğini geliştirmektedir. Bu sayede, şebekenin büyüklüğüne veya karmaşıklığına bakılmaksızın sistem kolayca uyarlanabilir.

Özetle, bu çalışma içme suyu şebekelerinin yönetiminde yeni bir ufuk açmaktadır. MQTT protokolü aracılığıyla verilerin internet ortamına aktarılması, daha kolay, daha ulaşılabilir ve daha verimli bir yönetim anlayışının önünü açmaktadır. Bu çalışmanın bulguları, su kaynaklarının daha etkin kullanımı ve daha sürdürülebilir bir su yönetimi için önemli bir katkı sağlayacaktır.

SONUÇLAR

Bu makalede, IoT sistemlerinde önemli bir protokol olan MQTT protokolü ile içme suyu şebekelerinin uzaktan takip ve kontrol edilebilirliği incelenmiştir. MQTT, hafif, güvenilir, esnek ve düşük bant genişliği gerektiren IoT uygulamaları için ideal bir iletişim protokolüdür. İçme suyu şebekelerinde kullanıldığında,

MQTT protokolü sayesinde sensörlerden alınan verilerin anlık olarak izlenmesi, analiz edilmesi ve gerektiğinde müdahalede bulunulması mümkün hale gelir.

Su kaynaklarının verimli kullanımı, su kalitesinin izlenmesi, su kayıplarının önlenmesi ve su arzının güvenliği gibi konular, akıllı şehirlerin gelişimine katkı sağlayabilecek en önemli uygulamalardan biri olan uzaktan yönetilebilen bir su şebekesi sistemi ile çözülebilir.

Gelecekte, su şebekeleri yapay zeka desteği ile otomatik olarak kendini ayarlayabilen, öngörücü bakım yapabilen ve olası arızalara karşı hızlı tepki verebilen dinamik bir yapıya kavuşturulabilir. Ayrıca, MQTT protokolünün avantajlarından yararlanarak, su şebekesi sistemi diğer akıllı şehir uygulamalarıyla entegre edilebilir. Böylece, şehir yönetiminde daha etkin kararlar alınması sağlanabilir.

KAYNAKLAR

Akdeniz, T., & Muhammetoğlu, H. (2019). Antalya İçme Suyu Şebekesinin Bir Bölümünün Online İzleme (SCADA) ve Coğrafi Bilgi Sistemleri (CBS) Araçları Kullanılarak Hidrolik Modellemesi. Su Kaynakları, 4(1), 12-22.

Ancy Jenifer, J., Leelipushpam Paulraj, G. J., Jebadurai, I. J., & Jebadurai, J. (2022). IoT-Based Smart Water Tank Supply Management System Using MQTT Protocol. In 2022 Seventh International Conference on Parallel, Distributed and Grid Computing (PDGC) (s. 647-652). Solan, Himachal Pradesh, Hindistan. https://doi.org/10.1109/PDGC56933.2022.10053134.

Daldal, N. (2018). İçme suyu şebeke otomasyonunun tasarımı ve gerçekleştirilmesi. Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi, 24(5), 831-836.

Karadirek, I. E., Kara, S., Yılmaz, G., Altındal, T., Muhammetoglu, H., Muhammetoglu, A., Kitis, M., Soyupak, S., Yigit, N., Harman, B., Palancı, İ., Akdeniz, T., & Cengiz, K. (2012). Antalya Konyaaltı Bölgesi İçme Suyu Kalitesinin İzlenmesi ve Yönetimi. Çevre Bilim ve Teknoloji Dergisi, 4, 35-40.

Kasaplı, K. (2014). İçmesuyu Şebekelerinde Maliyet Tahmini Amacıyla Yapay Sinir Ağları Kullanımı Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi, İstanbul

Kurban, R., Güner, M., & Kütük, A., (2020). İçme Suyu Temin Sistemlerinin SCADA Sistemleri ile Uzaktan Kontrolü ve Pompaların Hidrolik Analizi. 10. Pompa, Vana ve Kompresör Kongresi (pp.113-117). Ankara, Turkey

Mohanasundaram, V. S., Joyce, A., Naresh, K., Gokulkrishnan, G., Kale, A., Dwarakanath, T., & Haribabu, P. (2018). Smart water distribution network solution for smart cities: Indian scenario. 2018 Global Internet of Things Summit (GIoTS), 1-6.

Özkaya U., Ulukut Ö., Çömlekçi S., & Vardar G. "İçme suyu şebekesi kontrol otomasyonu". EMO Dergisi, http://www.emo.org.tr/ekler/adcff841230f72a_ek.pdf (15.10.2016).

Uçaner, M. E., & Özdemir, O. N. (2013). Genetik Algoritmalar İle İçme Suyu Şebekelerinde Ek Klorlama Optimizasyonu. Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 17(4).

Light, R. A. (2017). Mosquitto: server and client implementation of the MQTT protocol. The Journal of Open Source Software, 2(13). https://doi.org/10.21105/joss.00265

Soni, D., & Makwana, A. (2017). A survey on MQTT: A protocol of internet of things (IoT)2. *Communication Technology (ICCT), 2017 IEEE 17th International Conference on*, 1-6.

BÖLÜM II

Yazılım Projelerinde Çevik Yöntemler ile DevOps Süreçlerinin Entegrasyonu ve DevOps Araçlarının Kullanımı

Nurcihan DERE[1]
Kıvanç KIŞLAL[2]
Buket DOĞAN[3]

Giriş

Geleneksel yazılım geliştirme yaklaşımları genellikle aşamalı geliştirme süreçleri, bölünmüş ekipler ve bunun sonucunda ekipler arası kısıtlı iletişimi beraberinde getirir. Bununla birlikte uzun ve aşamalı geliştirme süreçlerinin sonunda ortaya çıkan ürünün müşteri ihtiyaçlarını karşılaması beklenir. Ancak bu süreçte değişen müşteri talepleri ve değişen pazar koşulları bunun gerçekleştirilmesini zorlaştırabilir. Bu yapıda bölünmüş ekipler haline çalışmak işbirliğini azaltabilir, süreçleri zorlaştırabilir.

[1] Marmara Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı,, nurcihankarayakali@marun.edu.tr, Orcıd: 0009-0009-6072-6990
[2] Mühendis, DIP Bilgisayar Yazılım Ticaret Anonim Şirketi, Orcid,:0009-0008-9015-5773
[3] Doç.Dr., Marmara Üniversitesi , Teknoloji Fakültesi Bilgisayar Mühendisliği, Orcid: 0000-0003-1062-2439

İletişim sorunları ekipler arasında bilgi eksikliğine neden olur, bu durum karşılaşılan sorunların etkili bir şekilde çözümlenmesini engeller. Bu nedenle geleneksel yazılım geliştirme yöntemleri günümüzün iş gereksinimlerini karşılamak için yeterli değildir. Günümüzde bu tür zorluklara daha hızlı ve esnek bir çözüm sağlamak için çevik (agile) yaklaşımlar daha yaygın olarak kullanılmaktadır. Çevik metodolojiler, sıkı iş birliği, hızlı döngüler ve esneklik gibi prensiplere odaklanarak yazılım geliştirme süreçlerini daha etkili ve başarılı bir şekilde yürütmeyi hedeflemektedir. Çevik yöntemler değişen müşteri ihtiyaçlarına daha hızlı cevap vermeyi hedefler ve ürünün gelişim sürecinde sürekli iyileştirilmesini sağlar. Projelerde çevik yaklaşımların uygulanması Yazılım Geliştirme Yaşam Döngüsünün (Software Development Life Cycle) esnekliğini, verimliliğini artırır ve değişen şartlarda daha hızlı aksiyon alabilmeyi sağlar (Dzamashvili Fogelström vd., 2010: 1).

DevOps, yazılım geliştirme (Development) ve işletme (Operations) süreçlerini entegre eden, iletişimi artıran, yazılım geliştirme ve dağıtım süreçlerinde otomasyonu vurgulayan yöntemler ve uygulama gruplarını ifade eder. Bu yaklaşım yazılım geliştirme süreçlerinde çevikliği sağlamak için geliştirme ve operasyon ekiplerini bir araya getirir, ekipler arasındaki işbirliğini artırır (Akbar vd., 2022: 2). DevOps'un temel amaçlarından biri de yazılım yaşam döngüsünü daha hızlı ve verimli hale getirmektir. Sürekli entegrasyon, sürekli teslimat, sürekli dağıtım gibi prensiplere odaklanır. Çevik birçok yöntemde olduğu gibi, DevOps ve sürekli teslimattaki fikirlerin çoğu aslında aynı temel kavramların farklı adlarıdır (Verona, 2016: 4). DevOps yaklaşımı da proje

geliştirme ve ürünü dağıtma aşamalarında sağladığı avantajlar sebebiyle tercih edilmektedir. DevOps, geliştirme ve operasyon ekiplerinin bir yazılım projesinin test, dağıtım ve yönetim aşamalarının hepsinin kapsandığı, SDLC (Software Development Life Cycle)'nin tamamında birlikte çalıştığı bir proje yönetim disiplinidir (Arifoğlu, 2020: 32). DevOps, geliştirme (development) ve işlemlerin (operations) birleşimi olarak da ifade edilebilir. DevOps, geliştirici ile operasyon arasında iletişim ve işbirliğini artırarak iki grup arasındaki boşluğu ekip çalışması ile doldurur (Giamattei vd., 2023: 3).

Sürekli geliştirme, test etme ve sürekli entegrasyon gibi süreçleri otomatikleştirir (Batra & Jatain, 2020). Geliştirme ve operasyon üyelerinin arasında gelişen iş birliği ve bütünlük DevOps ile çalışmanın başarısını ve verimliliğini artırır. Yazılım projelerinde test aşamasını otomatize etmek için DevOps araçları kullanmak, projelerin zaman ve maliyet açısından verimli bir şekilde yönetilmesine olanak tanır.

Sürekli entegrasyon (Continuous Integration- CI) ve sürekli dağıtım (Continuous Deployment -CD) yazılımın hızlı bir şekilde gerçekleştirilmesini ve teslim edilmesini sağlar; bu da projelerin verimliliğini artırır. Sürekli entegrasyon uygulamalarının ana faydaları, riski azaltmak ve yazılımı hatasız ve güvenilir kılmaktır. Sürekli teslimat ile sürümler daha hızlı teslim edilebilir ve ürün kalitesinde iyileştirme sağlar, müşteri memnuniyetini artırır (Arachchi ve Perera, 2018:1). Çevik metodoloji, pek çok projede ekiplerin verimliliğini artırmak amacıyla yaygın bir şekilde kullanılmaktadır. Yazılım projelerinde, Çevik yaklaşımlarla birlikte işlerde ve süreçlerde otomasyonu sağlamak için DevOps süreçlerini

kullanmak, zaman ve maliyet açısından önemli faydalar sağlamaktadır. Bu bütünleşik yaklaşım, geliştirme ve operasyon ekipleri arasındaki işbirliğini güçlendirirken, yazılım ürünlerinin hızlı ve güvenilir bir şekilde üretilmesine ve teslim edilmesine olanak tanır. DevOps'da geliştirme ve operasyon ekiplerinin birlikte çalışması ekip içindeki bilgi birikimini, ortak yetkinliği ve yeterliliği artırır (Faustino vd., 2022: 2).

DevOps ile İlgili Çalışmalar

DevOps yazılım geliştirme süreçlerinde taleplere uygun olarak yazılım geliştirme, test etme, kullanıcıdan geri dönüş alma ve iyileştirme gibi süreçleri uçtan uca sorunsuz bir şekilde ilerletmeye imkan sağlamaktadır. Literatürde yer alanla bu alanla ilgili önemli çalışmalar aşağıda yer almaktadır.

Geleneksel yazılım geliştirme yöntemleri genellikle yukarıdan aşağıya doğru planlama (şelale modeli) ve sıralı uygulama prensiplerine dayanır (Bick vd., 2017: 935). López-Fernández ve arkadaşlarının yaptığı çalışmaya göre, yeni yazılım projelerinde ve sık sık güncelleme yapılan projelerde geleneksel yöntemleri kullanmak yerine DevOps yaklaşımını benimsemek daha akıllıca olacaktır. Departmanlar Arası Dev & Ops işbirliği şirketlere ürünlerini daha kaliteli ve daha hızlı sunmaları için bir fırsattır (López-Fernández vd., 2021: 1).

Sevindik'in yaptığı çalışmada, Scrum, DevOps, Kanban bir arada kullanıldığı model önerisi ile kaynak kısıtları ve belirsizliğin çok olduğu projelerin eş zamanlı yürütmesi sağlanabilir. Aynı kaynakların birden fazla projelerde çalışması sırasında yaşanan çakışmaların önüne geçilebilir. Aynı zamanda, kaynakların görev

değişikliği yaparken harcadığı adaptasyon süreci düşürülebilir (Sevindik, 2015: 99).

Flefil ve arkadaşlarının Ürdün'de yaptığı bir çalışmada, araştırmaya katılan çalışanlar, DevOps'un yalnızca bir metodoloji değil, şirketlerin öğrenip benimsemesi ve kullanmaya başlaması gereken bir kültür olduğunu vurgulamaktadır. Ayrıca teknolojiye ayak uydurmanın ve yazılım sektöründe yeni otomasyon araçlarının kullanımının önemli olduğununa değinmektedirler (Flefil vd., 2022: 7).

Azad çalışmasında, DevOps'un kuruluştaki başarısı için geliştirme ve operasyon ekibi arasındaki işbirliğinin gerekli olduğu açıklanmaktadır. Geliştirme ve operasyon ekipleri arasında sorumlulukların paylaşılmasının önemi ve paylaşılan sorumlulukların, ekipler arasında etkili iletişimi de sağlayacağı vurgulanmaktadır (Azad, 2022: 84).

Projelerde ortak Agile ve DevOps anlayışında, ekipler arasında etkili ve kaliteli kod üretme sorumluluğu paylaşılır. Bu durum, her iki ekibin birlikte çalışması gerektiğini ve ortak hedeflere ulaşmak için işbirliğinin önemli olduğunu göstermektedir (Almeida vd. 2022: 11).

DevOps başarısı, diğer kalite özelliklerine göre değiştirebilirlik, test edilebilirlik ve desteklenebilirliğe öncelik veren gevşek bağlantılı mimarilerle en iyi şekilde ilişkilendirilir. Ayrıca operasyon uzmanları gelişmiş ve karmaşık operasyon görevlerini yerine getirmek için yazılım geliştirme ekiplerinin bir parçası olmalıdır. Test aşamalarını araçlar ile otomatik olarak

gerçekleştirmek de yazılım projelerinde ürünün sürekli değişimi ve sürekli dağıtımında faydalı olacaktır (Shahin vd., 2023: 26).

Agrawal ve Rawat çalışmalarında DevOps için öncelikle çeviklik, kalite, yenilik ve arızaların önlenmesi olmak üzere dört ana değer üzerinde odaklandığını ifade etmişlerdir. DevOps, üründeki kaliteyi artırarak son kullanıcı memnuniyetini artıracaktır (Agrawal ve Rawat, 2019: 985-985).

Çevik/DevOps ortamlarında mobil uygulamalar için test otomasyon araçlarının değerlendirildiği çalışmaya göre hem sürekli entegrasyon, hem sürekli dağıtım hem de sürekli teslimat için en çok kullanıldığı bildirilen araçlardan biri Jenkins'tir. Sıklıkla kullanılan android mobil uygulama ön yüz test araçları Appium, Espresso ve Robotium'dur (Baktha, 2020: 7-8).

Vihara Jayakody DevOps yaklaşımının yazılım geliştirme süreçlerinde uygulanması için gereken başarı faktörlerini analiz ettiği çalışmasında, geliştirme ve operasyon ekipleri arasındaki işbirliğinin önemini vurgulamıştır. Buna göre, bilgi paylaşımı, iletişim, süreç paylaşımı, projelerde ortak sorumluluk bilinci gibi etmenler DevOps süreçlerinin başarıyla uygulanması için dikkat edilmesi gereken temel faktörlerdendir (Jayakody ve Wijayanayake, 2023: 63-64).

Yöntem

Çevik Süreçlerin DevOps Araçları ile Gerçekleştirilmesi

Çevik yöntemlerdeki sürekli entegrasyon ve sürekli dağıtım projelerde üretkenliği artırır ve hızlı teslimatı sağlar (Arachchi ve Perera, 2018: 1). Çevik yöntemler, en popüler geliştirme yöntemlerinden biridir, ancak Çevik yöntemler ağırlıklı olarak

yazılım geliştirme kısmına odaklanmaktadır. Dağıtım ve operasyon alanına katılımı yoktur. Bu durumda, birçok şirket projelerin geliştirme süreçlerini iyileştirmek için çevik yöntemler ile DevOps'u entegre etmeyi amaçlamaktadır (Wang ve Liu, 2018: 5-6). DevOps araçları kullanılarak geliştirme aşamasında kodların entegrasyonu yapılabilir ve kodlar düzenli olarak sürüm versiyonları şeklinde test aşamasına ilerletilebilir.

Sürekli entegrasyonun (Continuous Integration – CI) ile yazılım projelerinde geliştirilen kodlar sürekli entegrasyon ile düzenli aralıklarla birleştirilir. Sürekli entegrasyonun amacı, bir değişiklik üretime geçmeden önce geri bildirim sağlayarak sorunları mümkün olan en kısa sürede tespit etmektir. Bu, kusurları önler aynı zamanda geliştirici üretkenliğini artırır, sürüm sıklığını hızlandırır ve değişikliklerin iletişimini geliştirir (Boone vd., 2022: 1-2). Sürekli entegrasyon ile kontrol edilen geliştirmeler sık sık yeni uygulama sürümleri çıkılarak dağıtılır. Bu Sürekli Dağıtım (Continuous Delivery – CD) olarak ifade edilir. Sürekli entegrasyon ve sürekli teslimat şirketlerin ürünlerin, daha hızlı güncellemelerini ve daha hızlı dağıtmalarını sağlar. Sürekli testler ile daha kaliteli ürün elde edilebilir. Ayrıca sürekli entegrasyon ve sürekli teslimat ile yazılım geliştirme süreçlerinde müşterilerden daha fazla geribildirim alınabilir. Geliştirme ve operasyon ekipleri arasındaki bağlantı güçlendirilir, manuel görevler ortadan kaldırılabilir (Shahin vd., 2017: 2).

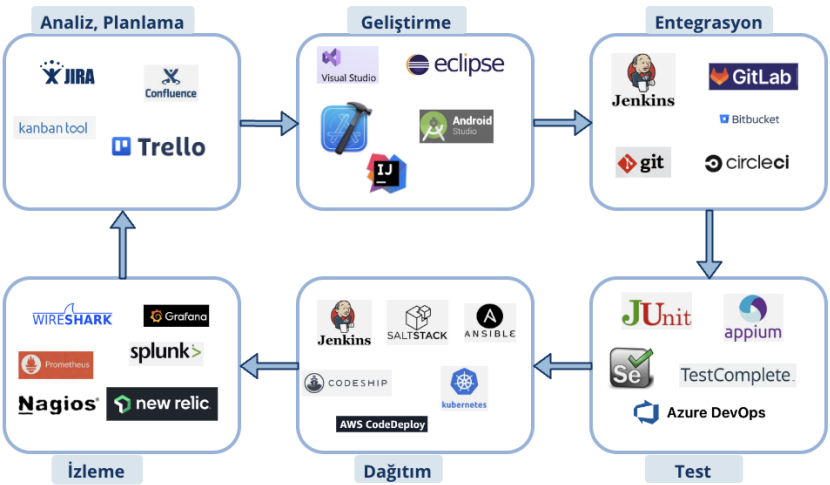
Çevik yöntemlerdeki Sürekli entegrasyon (Continuous Integration- CI) ve sürekli dağıtım (Continuous Deployment -CD) yaklaşımları DevOps araçları ile daha başarılı uygulanabilir. Geliştirilen ürünün testi ve dağıtımı daha hızlı yapılabilir. Bu

nedenle yazılım projelerinde kullanılan çevik yöntemler ile DevOps bir arada projelerde verimliliğin artması beklenir. DevOps, bir kuruluşun uygulamaları ve hizmetleri yüksek hızda sunma yeteneğini artırmak için felsefeleri, uygulamaları ve araçları birleştirir (Dörnenburg, 2018: 72-74).

DevOps'un hıza (üretim ortamına çıkış süresine) odaklanarak yüksek kalite ve düşük maliyet sağlaması, yazılım geliştirme yaşam döngüsü boyunca testlerin bir otomasyon ile gerçekleştirilmesine önem verilmesi gerektiğini gösterir. Ayrıca testlerin gerçekleştirilme süresi ve hataların giderilmesi teslimat süresini önemli ölçüde etkiler (Pando ve Dávila, 2022: 658-659). Geliştirmelerin testi ile hatalar tespit edilebilecek, düzenlenecek ve daha hızlı kaliteli ürün elde edilecektir. DevOps ile otomatik yazılım testi becerileri küçük şirketlerin dünya pazarında kaliteli ürün alanında rekabet etmesine yardımcı olacaktır (Azad, 2022: 89).

Yazılım testi, yazılım geliştirme yaşam döngüsünün başlangıcında belirlenen gereksinimlerin karşılanma derecesini kontrol eden aktiviteler bütünü olarak tanımlanmaktadır (Hancı & Gökbay, 2017). DevOps süreçleri, testlerin otomatize edilmesine ve geliştirme senaryoları için uygun test ortamının oluşturulmasına odaklanır (Agrawal ve Rawat, 2019: 983-984). Testlerin bir test otomasyon aracı kullanılarak gerçekleştirilmesi ürünün hızlı ve sistematik bir şekilde kontrol edilmesini sağlar. Test otomasyon araçları ile detaylı test senaryoları yazılabilir, manuel testlerde karşılaşılabilen eksik ya da unutulmuş test senaryoları engellenmiş olur. Geliştirilen üründe kontroller otomatik olarak ilerletilir ve bu test süreçleri ve sonuçları detaylı olarak raporlanır.

Aşağıda Şekil 1'de yazılım proje geliştirme aşamaları ve bu aşamalarda kullanılabilecek örnek DevOps araçları gösterilmektedir. Yazılım yaşam döngüsündeki aşamalar, DevOps araçları ve özellikleri aşağıdaki gibi açıklanabilir.



Şekil 1. *Yazılım Geliştirme Aşamaları ve DevOps Araçları*

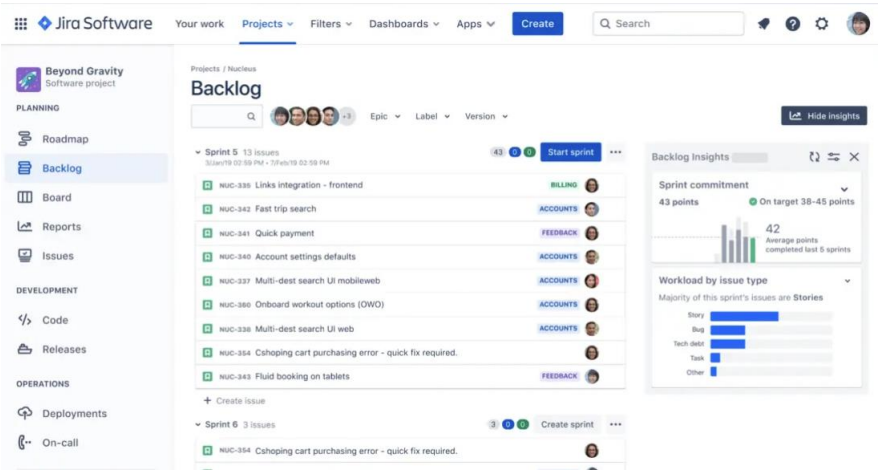
Analiz ve Planlama

Yazılım geliştirme süreçlerinde analiz ve planlama aşaması, projenin gereksinimlerini belirleme, kapsamını tanımlama, kaynak ve zaman çizelgesini oluşturma gibi çeşitli adımları içerir. Yazılım projesinin başlangıcında, proje hedefleri belirlenir, kaynaklar tahsis edilir ve bir proje planı oluşturulur. Analiz ve planlama aşamalarındaki doğru ve kapsamlı çalışma, projenin başarılı bir şekilde ilerlemesi için temel bir adımdır. Bu süreçte, DevOps araçları kullanılarak süreçler etkili ve hızlı bir şekilde yönetilebilir. Bu da projenin başarılı bir şekilde ilerlemesine katkı sağlar. Aşağıda örnek araçlardan bir kısmı anlatılmaktadır.

JIRA, Atlassian şirketi tarafından geliştirilen ve yazılım geliştirme süreçlerini, proje yönetimini ve iş takibini kolaylaştırmak amacıyla kullanılan yönetim aracıdır. Proje ekipleri, görevleri ve işleri organize etmek, atamak ve takip etmek için JIRA'yı kullanabilirler. Yazılım geliştirme süreçlerinde, hata takibi, projelere yeni özellik ekleme, geliştirme görevleri için kullanılabilir. Yazılım geliştiriciler, bu platform üzerinde çalışma durumlarını takip edebilir ve ekip içinde işbirliği sağlanabilir. Şekil 2'de örnek JIRA arayüzü gösterilmektedir. JIRA, birçok diğer yazılım geliştirme aracı ve servisi ile entegre edilebilir. Bu entegrasyonlar, farklı araçlar arasında veri paylaşımını ve iş akışını kolaylaştırır (Mittal ve Mehta, 2020: 1-4).

Kanban panosu, Kanban yönetim metodolojisinin temel unsurlarından biridir. Projelerde akışları görselleştirmek, büyük iş süreçlerini daha küçük ve kolay parçalara ayırmak ve yönetmek için kullanılabilir. Bu panolar, iş akışını ve görevlerin durumlarını açık bir şekilde göstererek ekiplerin işlerini daha etkili bir şekilde organize etmelerine yardımcı olur. Yazılım geliştirme süreçlerinde kullanılan Kanban panosu genellikle Backlog, Analiz, Geliştirme, Test ve Dağıtım gibi aşamalara bölünmüştür. Devam eden işler "Yapılıyor", tamamlanan işler "Yapıldı" gibi alt sütunlarda yer alır. Panolar, ekiplerin devam eden çalışmayı görsel olarak gözlemlemesine ve kendi görevlerini organize etmesini kolaylaştırır (Damij ve Damij, 2021: 3). DevOps mimarisinde Kanban panosu ile ekip içi ve ekipler arası iletişim daha etkili hale gelir. Confluence Atlassian tarafından geliştirilen bir belge yönetim platformudur, projelerde döküman yönetimi için kullanılır. JIRA ile entegre edilebilir. Proje belgelerini oluşturmak, düzenlemek ve paylaşmak

için Confluence kullanılabilir. Yazılım geliştirme standartları, kodlama yönergeleri ve diğer yazılım geliştirme süreçleri için belgeler oluşturulabilir. Tartışma sayfaları, yorumlar ve izleme özellikleri sayesinde takım üyeleri arasında etkileşimi kolaylaştırır (Reha ve Fai, 2021: 87-88).



Şekil 2. *JIRA Kullanıcı Arayüzü (Jira Software,2024)*

Geliştirme

Yazılım geliştirme süreçlerinde geliştiriciler proje gereksinimlerine uygun olarak kodlarını yazarlar ve kodlar bir version kontrol sistemi kullanılarak yönetilir. DevOps'da yer alan sürekli geliştirme ile geliştiricilerin yaptığı değişiklikler, ekipteki tüm geliştiricilerle paylaşılır ve işbirliğini teşvik eder. Sürekli geliştirme prensipleri, ekiplerin daha hızlı ve güvenilir bir şekilde yazılım teslim etmelerine imkan tanır. Yazılım geliştirme süreçlerinde DevOps ile entegre edilebilen geliştirme araçlarından bazıları aşağıdaki gibidir.

Visual Studio, Microsoft'un geliştiricilere yönelik olarak tasarlamış olduğu bir entegre geliştirme ortamıdır (Integrated Development Environment- IDE). Yazılım projeleri oluşturmak, düzenlemek, derlemek, hata ayıklamak ve dağıtmak gibi birçok geliştirme görevini destekleyen kapsamlı bir araçtır. Eclipse, Java ve diğer birçok programlama dili için geliştirilmiş açık kaynaklı bir entegre geliştirme ortamıdır. Geniş bir eklenti ve uzantı ekosistemine sahiptir. Geliştiricilerin ihtiyaçlarına uygun özellikleri eklemelerine ve IDE'yi özelleştirmelerine imkân tanır. Geliştiricilere kod yazarken otomatik tamamlama, kod düzenleme, hata kontrolü ve hızlı gezinme gibi özellikler sunar.

Xcode, Apple tarafından Mac bilgisayarlar, iPhone'lar ve diğer Apple ekosistemindeki cihazlar için uygulama geliştirmeyi sağlamak için geliştirilmiş bir yazılım aracıdır. Geliştiricilere uygulama projelerini oluşturmak, düzenlemek ve yönetmek için çeşitli özellikler sunar (Apple Inc, 2024, paragraf 1). Android Studio, Android uygulamaları geliştirmek için tasarlanmış ve geliştirilmiş bir entegre geliştirme ortamıdır. Google tarafından geliştirilen Android Studio, Android uygulama ve oyun geliştiricilerine modern bir geliştirme deneyimi sunmayı amaçlar. XML tabanlı bir kullanıcı arayüzü tasarım aracına sahiptir. Bu araç, görsel olarak kullanıcı arayüzleri oluşturmayı ve düzenlemeyi sağlar (Android Studio, 2024, paragraf 1-8).

Entegrasyon

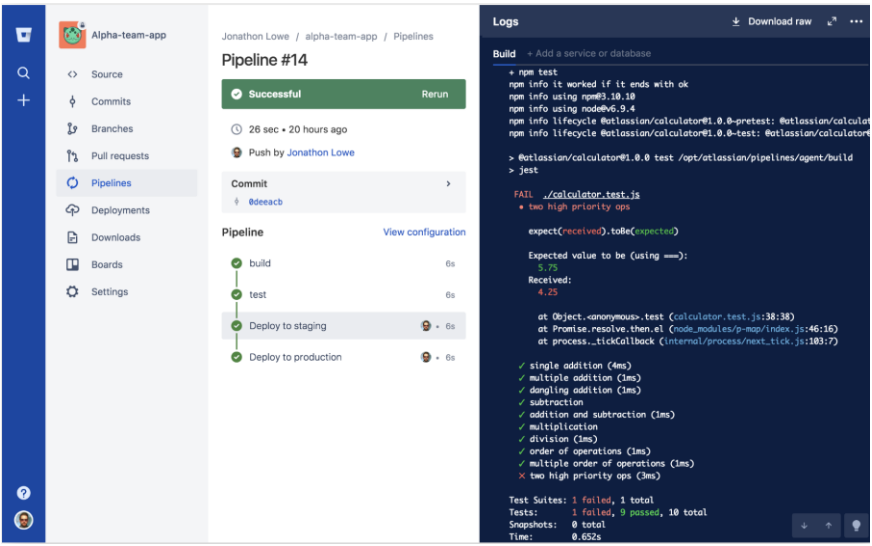
DevOps'da yer alan sürekli entegrasyon ile geliştiricilerin yaptığı değişiklikler, entegrasyon araçları kullanılarak belli aralıklarda otomatik olarak birleştirilir ve test edilir. Bu adım, yazılımın sürekli olarak kontrol edilmesini ve uyumluluk

sorunlarının erken aşamalarda tespit edilmesini sağlar. DevOps ile hataların erken aşamalarda yakalanması, yazılım projelerinin daha güvenilir, hızlı ve etkili bir şekilde ilerlemesine katkı sağlar. Sürekli entegrayon aşamasında kullanılabilecek örnek araçlar aşağıda yer almaktadır.

Yazılım geliştirme süreçlerinde kod versiyon kontrolü için Git tabanlı bir sürüm kontrol sistemi olan Bitbucket kullanılabilir. Kodlar ve yapılan değişikliklerin geçmişi ekip üyeleri arasında paylaşılabilir, izlenebilir (Benedetti vd., 2023: 5-6). Şekil 3'de gösterildiği gibi otomatize edilerek geliştirilen kodlar sürümler halinde dağıtılabilir.

Jenkins, açık kaynaklı bir sürekli entegrasyon ve sürekli dağıtım aracıdır. Jenkins, yazılım geliştirme süreçlerini otomatikleştirmek ve iyileştirmek amacıyla kullanılır. Yazılım geliştirme süreçlerindeki kod değişikliklerini düzenli olarak birleştirip test etmeyi sağlar.

CircleCI, yazılım geliştirme süreçlerini sürekli entegrasyon ve sürekli dağıtım prensiplerine dayalı olarak ilerletmek için kullanılan bulut tabanlı bir hizmettir. CircleCI, yazılım projelerinin hızlı bir şekilde derlenmesini, test edilmesini, ve dağıtılmasını sağlayarak geliştirme süreçlerini optimize eder, versiyon kontrol sistemleri, bulut platformlar ve diğer araçlarla entegrayonu sağlar.



Şekil 3. *Bitbucket ile Sürüm Oluşturma (Atlassian, 2024)*

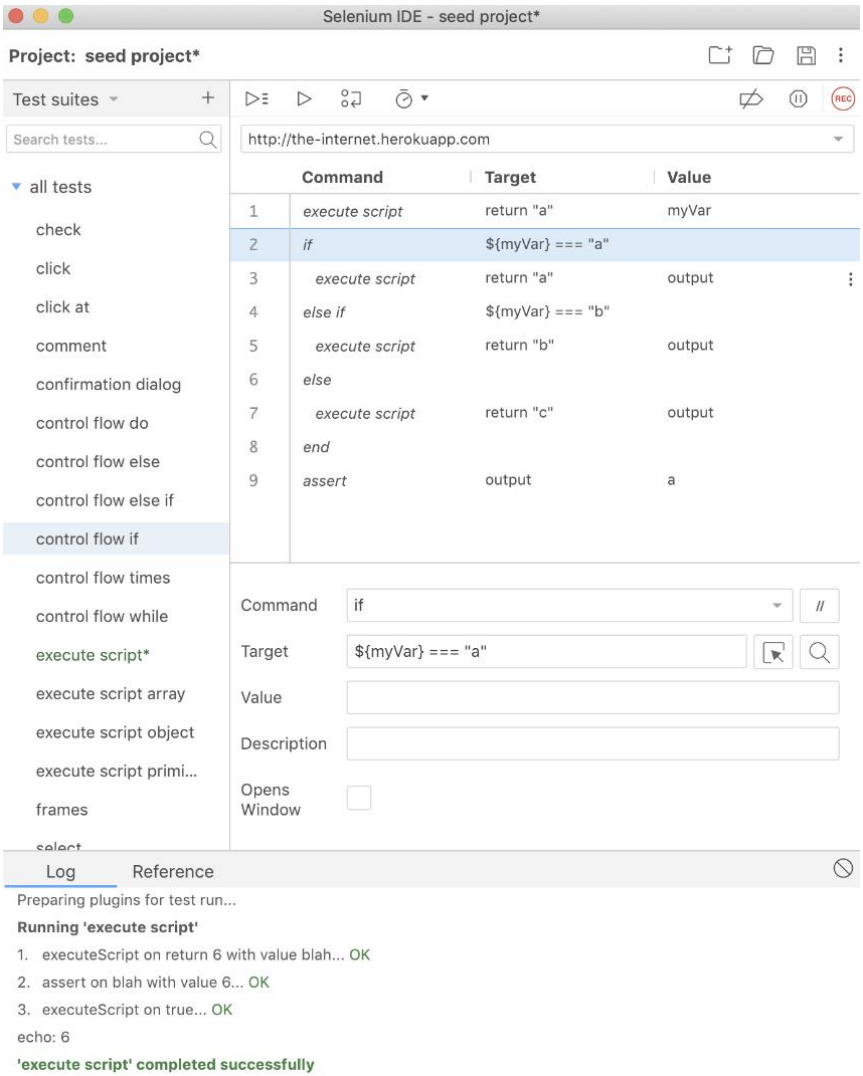
Test

Yazılım geliştirme süreçlerinde test aşamasında, yazılım farklı durum ve kullanım senaryolarında test edilir. DevOps süreçlerinde sürekli test aşamasında ise, bu testler otomatikleştirilir ve yazılım sürekli olarak kontrol edilir, yeni kod eklendikçe veya değişiklik yapıldıkça otomatik test komutları çalıştırılarak hızlı geri bildirim sağlanır. Bu süreç, yazılımın güvenilirliğini artırmak, hataları erken aşamalarda tespit etmek ve hızlı bir şekilde kullanıma sunmak için önemlidir. DevOps'ta sürekli test, sürekli entegrasyon ve sürekli teslimat süreçlerinin ayrılmaz bir parçasıdır. DevOps süreçlerinde kullanılabilecek örnek test araçlarının bazıları Şekil 1'de gösterilmiştir, aşağıda bir kısmı açıklanmıştır.

Çevik yazılım geliştirme süreçlerinde planlama, geliştirme, test ve ürünün dağıtımı aşamaları DevOps araçları ile gerçekleştirilebilir. Örneğin web uygulamaları için Azure DevOps

ile uygulamalar test edilebilir ve test sonuçları rapor halinde elde edilebilir. Azure DevOps, uçtan uca test yönetimi yapmak için bir dizi test özelliği sağlar. Test planları, test takımları ve test senaryoları, diğer destekleyici özelliklerle birlikte test yönetiminin temel öğeleridir. Ayrıca entegre platform, test senaryoları ile gereksinimler, uygulamalar, hatalar ve diğer işlemler arasında tam izlenebilir bir uçtan uca bir entegrasyon sağlar (KK , 2020: 159-186).

Web uygulamaları için bir diğer araç olan Selenium IDE açık kaynak kodlu bir test otomasyon aracıdır. Birçok platformla entegrasyonu kolaydır. Web uygulamalarında test için tarayıcı tabanlı etkileşimleri kaydedip tekrar aynı sıra ile gerçekleştirme imkanı sunar. Selenium ile gerçekleştirlen paralel testler test uzmanları için kolaylık sağlar ve hızlı geribildirim almayı mümkün kılar (Izzat ve Saleem, 2023: 1-3). Selenium IDE kullanarak gerçekleştirilen bir test senaryosunda web uygulamasında yapılan tüm veri girişi ve aksiyonlar zaman etiketi ile kaydedilebilir. Test aşamalarındaki sonuçlar başarılı ya da başarısız gibi statüler ile test tablolarına eklenebilir (Ezhuthachan ve Tailor, 2022: 264-270). Şekil 4'de örnek Selenium arayüzü yer almaktadır.



Şekil 4. *Selenium IDE ile Test için Kontrol Aşamaları ("Control Flow,")*

JUnit, Java programlama dilinde yazılmış test otomasyon kütüphanesidir. JUnit, yazılım geliştiricilerin birim testlerini oluşturmasını, çalıştırmasını ve değerlendirmesini kolaylaştırır. Paralel test yürütme özelliği ile eşzamanlı testler gerçekleştirilebilir

(Mythily vd., 2023: 1549). Appium ise web sürücüsü kullanarak yerel uygulamaları, mobil web uygulamalarını ve hibrit uygulamaları test etmek için kullanılan açık kaynaklı test otomasyon aracıdır. Otomasyon sayesinde gerçekleştirilen testler projelerde harcanan insan kaynağı, zaman ve maliyet açısından tasarruf etmeyi sağlar (Izzat ve Saleem, 2023: 1-3).

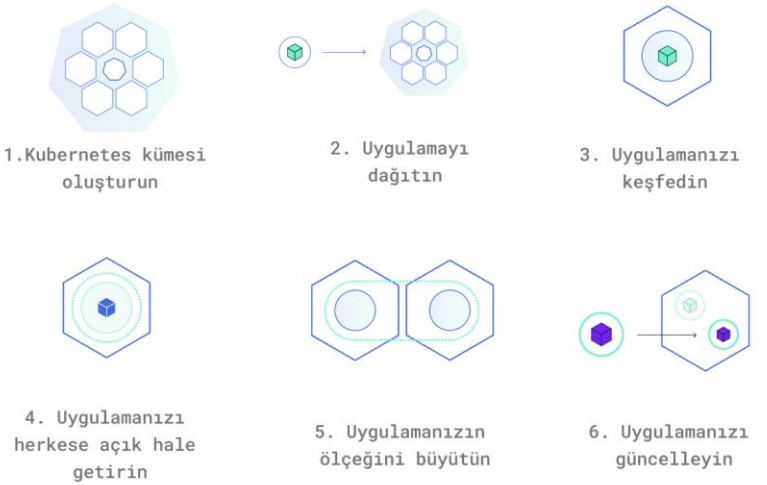
Dağıtım

Sürekli dağıtım, yazılımın geliştirme ve test aşamalarını geçtikten sonra, otomatik ve sürekli bir şekilde üretim ortamına dağıtılmasını ifade eder. Sürekli dağıtım, sürekli entegrasyon ve sürekli teslimat süreçleri ile birlikte düşünüldüğünde yazılımın sürekli olarak kontrol edilmesi, test edilmesi ve kullanıma sunulması için bir bütün olarak ele alınır. Bu, hataların erken aşamalarda tespit edilmesini ve güncellemelerin daha hızlı bir şekilde yayımlanabilmesini sağlar. DevOps süreçlerinde dağıtım aşamasında kullanılabilecek örnek araçlar aşağıdaki gibidir.

Kubernetes, kapsayıcı uygulamaları hızlı bir şekilde dağıtmaya, ölçeklendirmeye ve yönetmeye olanak tanıyan, yaygın kullanılan açık kaynak kodlu bir DevOps aracıdır (Ogala, 2022: 5). Şekil 5'te Kubernetes'in temel bileşenleri gösterilmektedir. Konteyner teknolojileri kullanılarak oluşturulan uygulamaların bu bileşenler aracılığıyla dağıtımı, güncellemesi ölçeklendirmesi ve yönetimi gibi işlemleri gerçekleştirilebilir. Kubernates, uygulama ve altyapının sağlıklı bir şekilde çalışması için izleyen ve değerlendiren bir yapıya sahiptir. Uygulamalarda hatalı olan nesneleri algılar ve otomatik olarak yeniden başlatarak sistemdeki hatalara karşı güvenilir bir çalışma ortamı sağlar (Mondal vd., 2022: 2941-2942).

Jenkins ve Codeship, sürekli entegrasyon ve dağıtım için kullanılan otomasyon araçlarıdır (Sen vd., 2022: 146). Jenkins, kodlarda yapılan değişiklikleri versiyon kontrol sistemlerinde takip edilerek otomatik olarak bir dizi işlemi başlatabilir, yazılımı derleme, test etme ve dağıtım aşamalarını gerçekleştirilebilir. Bu araçlar ile DevOps süreçlerinde otomasyon ve sürekli dağıtım gerçekleştirilebilir. Azure DevOps ise Microsoft'un bulut tabanlı hizmet ve araç setidir. Yazılım geliştirme süreçlerini yönetmek ve takip etmek için otomasyonlar sunar. Azure DevOps, tüm yazılım yaşam döngüsü boyunca geliştirme, test ve dağıtım süreçlerini destekleyen entegre bir platformdur (Azure DevOps, 2024).

Ansible yapılandırma yönetimi, yazılım dağıtma, bulut sağlama gibi görevleri otomatik olarak gerçekleştirmeyi sağlayan açık kaynak kodlu bir otomasyon aracıdır. İşlemleri gerçekleştirmek için olan süreç detayları prosedür şeklinde yapılır. Sürekli dağıtım ve teslimatlar gerçekleştirilebilir (Santos vd., 2023: 89396). DevOps süreçlerinin hedefi olan sürekli iyileştirme ve otomasyon ilkesi doğrultusunda, sürekli entegrasyon ve sürekli teslimat prensipleri ile yazılımın hızlı ve güvenilir bir şekilde gerçekleştirilmesi ve üretim ortamına taşınması sağlanabilir.



Şekil 5. *Kubernetes'in Temel Bileşenleri (Kubernetes, 2024)*

SaltStack, açık kaynak kodlu bir otomasyon ve yönetim aracıdır. Özellikle büyük ölçekli sistemlerde, bulut altyapılarında ve veri merkezi gibi ortamlarda kullanılmak üzere tasarlanmıştır. Ölçeklenebilirlik konusunda ön planda olan bu platform, geniş çaplı altyapıları ve karmaşık sistemleri başarıyla yönetme yeteneği ile öne çıkar. DevOps yaklaşımında sürekli entegrasyon ve sürekli teslimat süreçlerinin bir parçası olarak kullanılabilir. Yine bu süreçlerde tekrarlanabilir görevleri otomasyon ile gerçekleştirmek için kullanılabilir.

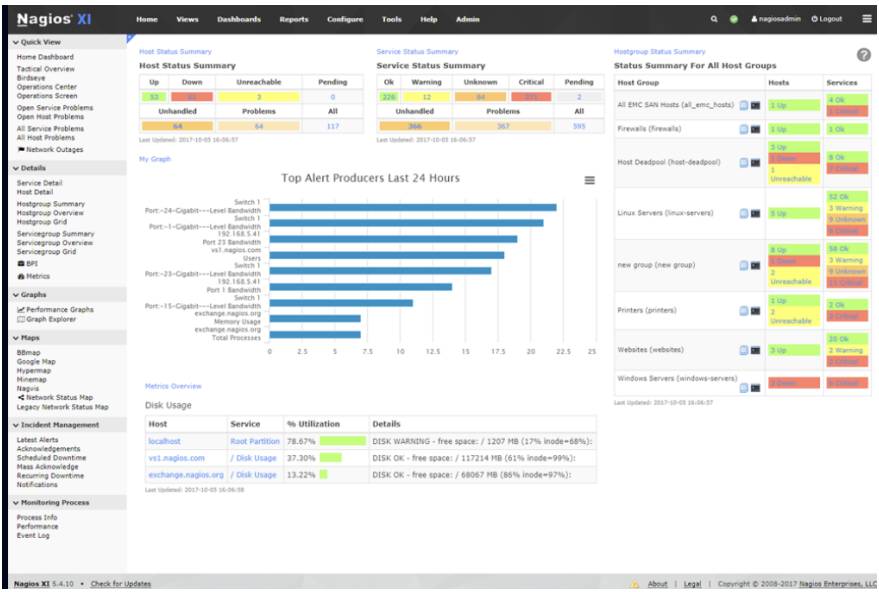
Sürdürülebilirlik ve İzleme

Wireshark, ağ analizi ve paket yakalama amacıyla kullanılan açık kaynaklı bir yazılımdır. Ağ trafiğini izleme, analiz etme ve bu trafiği paket düzeyinde inceleme özellikleri sunar. Wireshark, çeşitli

ağ protokollerini destekler ve ağ üzerindeki iletişimi detaylı bir şekilde gözlemleme imkanı tanır (Wireshark 2024).

Grafana, veri görselleştirme ve izleme amacıyla kullanılan açık kaynaklı bir platformdur. Farklı veri kaynaklarından gelen verileri toplar, bu verileri grafikler, tablolar ve panolar üzerinde görselleştirir. Kullanıcılar, bu görselleştirmeleri kullanarak sistem performansını, uygulama metriklerini, ağ verilerini izleyebilir. Grafana, kullanıcılara belirli verilere erişim ve değişiklik yapma yetkileri vererek rol yönetimi sağlar (Grafana, 2024).

Nagios, ağı ve altyapıyı, ağa bağlı cihazları izlemeyi sağlayan bir uygulamadır. Şekil 6'te örnek Nagios arayüzü gösterilmektedir. Sunucular, anahtar noktalar ve ağ cihazları gibi çeşitli öğeleri izleyerek, performans sorunlarını tespit eder ve bildirir. Anlık bildirimler ve ayrıntılı raporlar sağlayarak, sistem yöneticilerine ağlarının durumu hakkında gerçek zamanlı bilgi sunar. Prometheus da bir izleme aracıdır. Özellikle kapsayıcı ortamları için tasarlanmıştır. Ölçümleri, işlem kayıtlarını ve hata ayıklama verilerini toplayarak, uygulamaların performansını analiz etmeyi ve sorunları tespit etmeyi sağlar. Splunk, web siteleri ya da uygulamalar tarafından oluşturulan verileri aramak, analiz etmek ve görselleştirmek için kullanılan bir yazılım uygulamasıdır. Karmaşık sorgular, görselleştirmeler ve raporlar oluşturarak işletmelerin verilerini etkili bir şekilde anlamalarını ve kullanmalarını sağlar (Kothawade). DevOps süreçlerinde bu izleme araçları kullanılabilir. Bu, performansın, güvenliğin, hataların ve genel sistem durumunun sürekli olarak izlenmesini ve değerlendirilmesini sağlar.



Şekil 6. *Nagios Arayüzü (Nagios, 2024)*

Sonuç

Yazılım geliştirme süreçlerinde çevik yöntemler, sürekli entegrasyon ve sürekli teslimat yaklaşımları ile değişen isteklere ve ihtiyaçlara hızlı yanıt vermek, kaliteli ürün çıkarmak hedeflenmektedir. DevOps yaşam döngüsünde yer alan sürekli dağıtım, sürekli test, sürekli geribildirim gibi süreçler çevik yöntemlerdeki amaçlar ile örtüşmektedir. Bu süreçler, yazılımın her aşamasında geri bildirim almayı, hataları erken tespit etmeyi ve geliştirme sürecini sürekli iyileştirmeyi hedeflemektedir. Bu amaçlar doğrultusunda günümüzde yazılım projeleri için planlama, geliştirme, test, ürünü dağıtma ve izleme için pek çok DevOps aracı bulunmaktadır. Bu araçlar, sık ve düzenli test sürümleri oluşturmayı, bu sürümler üzerinden geri bildirim almayı ve yüksek kalitede ürün elde etmeyi sağlamaktadır. Yazılım testleri için kullanılan DevOps

araçları, projelerde çevik yöntemler ile entegre edildiğinde, test aşamaları daha hızlı gerçekleşmekte ve test sonuçları geliştirme ekiplerine daha erken iletilebilmektedir. Bu durum geliştirme sürecindeki herhangi bir hatayı daha erken ve hızlı tespit etmeyi mümkün kılmaktadır. Yazılım projelerinde testlerin gerçekleştirilme hızı projelerin teslimat sürecini de hızlandırmaktadır. Çevik yaklaşımdaki değişen müşteri isteklerine daha hızlı cevap verme ve uyum sağlama hedefi, DevOps araçları ile sürekli ve düzenli sürümler teslim ederek daha kolay gerçekleştirilebilmektedir. DevOps yaklaşımı ile projeler, kullanıcı geri bildirimlerine uygun bir şekilde geliştirilebilir, bu da müşteri odaklı iyileştirmeleri mümkün kılar. DevOps araçları, ekipler arasında iş birliğini artırır ve sürekli iyileştirme kültürünü destekler.

Sonuç olarak, çevik ve DevOps yaklaşımları bir araya geldiğinde, yazılım geliştirme süreçlerini daha esnek, daha hızlı ve verimli hale getirebilmek mümkündür. Bu, ekiplerin değişen taleplere hızlı yanıt vermesini sağlayacak, hataların erken tespit edilmesini ve projelerin başarıyla tamamlanmasını destekleyecektir. Müşteri memnuniyetini artırmak ve rekabet avantajı elde etmek için çevik süreçler ile DevOps prensiplerine odaklanan projeler, geleceğin yazılım geliştirme süreçlerine önderlik edecektir.

Kaynakça

Agrawal, P., & Rawat, N. (2019). *DevOps, a new approach to cloud development & testing*. 2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT).

Akbar, M. A., Rafi, S., Alsanad, A. A., Qadri, S. F., Alsanad, A., & Alothaim, A. (2022). *Toward successful DevOps: a decision-making framework.* IEEE access, 10, 51343-51362.

Almeida, F., Simões, J., & Lopes, S. (2022). *Exploring the benefits of combining devops and agile.* Future Internet, 14(2), 63.

Android Studio. Erişim tarihi: 08.02.2024, https://developer.android.com/studio

Arachchi, S., & Perera, I. (2018). *Continuous integration and continuous delivery pipeline automation for agile software project management*. Paper presented at the 2018 Moratuwa Engineering Research Conference (MERCon).

Arifoğlu, Ö. F. (2020). *Büyük Ölçekli Uygulamaların Devops Süreçlerine Uyarlanması ve Uygulanması*. (Yayımlanmamış Yüksek Lisans Tezi), Sakarya Üniversitesi, Sakarya

Azad, N. (2022). *Understanding DevOps critical success factors and organizational practices.* Paper presented at the Proceedings of the 5th International Workshop on Software-intensive Business: Towards Sustainable Software Business.

Azure DevOps. Erişim tarihi: 08.02.2024, https://azure.microsoft.com/en-us/products/devops

Baktha, K. (2020). *Evaluating the Performance and Capabilities of Popular Android Mobile Application Testing Automation Frameworks in Agile/DevOps Environment.*

Batra, P., & Jatain, A. (2020). *Measurement Based Performance Evaluation of DevOps*. Paper presented at the 2020 International Conference on Computational Performance Evaluation (ComPE).

Benedetti, G., Verderame, L., & Merlo, A. (2023). *A Preliminary Study of Privilege Life Cycle in Software Management Platform Automation Workflows.* Available at SSRN 4385101.

Bick, S., Spohrer, K., Hoda, R., Scheerer, A., & Heinzl, A. (2017). *Coordination challenges in large-scale software development: a case study of planning misalignment in hybrid settings*. IEEE Transactions on Software Engineering, 44(10), 932-950.

Boone, C., Brandt, C., & Zaidman, A. (2022*). Fixing continuous integration tests from within the IDE with contextual information. Paper presented at the Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension*.

Build powerful, automated workflows, Erişim tarihi: 08.02.2024, https://www.atlassian.com/software/bitbucket/features/pipelines

Control Flow. Erişim tarihi: 08.02.2024, https://www.selenium.dev/selenium-ide/docs/en/introduction/control-flow

Damij, N., & Damij, T. (2021). *An approach to optimizing Kanban board workflow and shortening the project management plan.* IEEE Transactions on Engineering Management.

Dörnenburg, E. (2018). *The path to devops*. IEEE software, 35(5), 71-75.

Dzamashvili Fogelström, N., Gorschek, T., Svahnberg, M., & Olsson, P. (2010). *The impact of agile principles on market-driven software product development. Journal of software maintenance and evolution*, Research and practice, 22(1), 53-80.

Ezhuthachan, S. D., & Tailor, J. K. (2022). *Test Case to Test Automation with Selenium IDE: a Case Study. Compendium of Management Case Studies*, 263, 2022.

Faustino, J., Adriano, D., Amaro, R., Pereira, R., & da Silva, M. M. (2022). *DevOps benefits: A systematic literature review. Software: Practice and Experience*, 52(9), 1905-1926.

Flefil, A., Alawneh, L., & Albalas, F. (2022). *DevOps Culture in Software Development Companies in Jordan*. Paper presented at the 2022 13th International Conference on Information and Communication Systems (ICICS).

Giamattei, L., Guerriero, A., Pietrantuono, R., Russo, S., Malavolta, I., Islam, T., Armbruster, M. (2023). *Monitoring tools for DevOps and microservices: A systematic grey literature review. Journal of Systems and Software*, 111906.

Grafana, Erişim tarihi: 08.02.2024, https://grafana.com/

Hancı, A. K., & Gökbay. (2017), I. *Yazılım Testi Tasarım Tekniklerinin Matematiksel Model Yaklaşımı ile Analizi*.

(Yayımlanmamış Yüksek Lisans Tezi), İstanbul Üniversitesi Fen bilimleri Enstitüsü Enformatik Ana Bilim Dalı. İstanbul

Izzat, S., & Saleem, N. N. (2023). *Software testing techniques and tools: A review. Journal of Education and Science,* 32(2), 30-44.

Jabbari, R., bin Ali, N., Petersen, K., & Tanveer, B. (2018). *Towards a benefits dependency network for DevOps based on a systematic literature review*. Journal of Software: Evolution and Process, 30(11), e1957.

Jayakody, V., & Wijayanayake, J. (2023). *Critical success factors for DevOps adoption in information systems development*. International Journal of Information Systems and Project Management, 11(3), 60-82.

Jira Software is the #1 software development tool used by agile teams. Erişim tarihi: 08.02.2024, https://www.atlassian.com/software/jira?&aceid=&adposition=&adgroup=144803328227&campaign=18452096610&creative=663328874107&device=c&keyword=jira&matchtype=e&network=g&placement=&ds_kids=p73363643282&ds_e=GOOGLE&ds_eid=700000001558501&ds_e1=GOOGLE&gad_source=1&gclid=Cj0KCQiAnrOtBhDIARIsAFsSe50hmKtZ8kAMUbU8HVfCBJxTrR98CDqR8B_QYFzITcFQork60a2pHQwaAvF0EALw_wcB&gclsrc=aw.ds

KK, A., & KK, A. (2020). *Test Management Using Azure DevOps. Azure DevOps for Web Developers: Streamlined Application Development Using Azure DevOps Features*, 159-186.

Kothawade, M. R.(2020) *DevOps and Selection of Appropriate Tool: A Review*.

Learn Kubernetes Basics, Erişim tarihi: 08.02.2024, https://kubernetes.io/docs/tutorials/kubernetes-basics/

López-Fernández, D., Díaz, J., García, J., Pérez, J., & González-Prieto, Á. (2021). *Devops team structures: Characterization and implications.* IEEE Transactions on Software Engineering, 48(10), 3716-3736.

Mittal, P., & Mehta, P. (2020). *Optimization of software development process by plugin integration with jira–a project management tool in devops.* Paper presented at the Proceedings of the International Conference on Innovative Computing & Communications (ICICC).

Mondal, S. K., Pan, R., Kabir, H. D., Tian, T., & Dai, H.-N. (2022). *Kubernetes in IT administration and serverless computing: An empirical study and research challenges*. The Journal of Supercomputing, 1-51.

Mythily, M., Kanakala, V. R., & Nambiar, R. (2023). *An Extensive Review of Spring Boot Testing Based on Business Requirements of the Software*. Paper presented at the 2023 4th International Conference on Smart Electronics and Communication (ICOSEC).

Ogala, J. O. (2022). *A Complete Guide to DevOps Best Practices*. International Journal of Computer Science and Information Security (IJCSIS), 20(2).

Pando, B., & Dávila, A. (2022). *Software Testing in the DevOps Context: A Systematic Mapping Study*. Programming and Computer Software, 48(8), 658-684.

Reha, M., & Fai, V. (2021). *Project Management Tools in the Classroom: Using the Atlassian Tool Suite in the Classroom*. Journal of Instructional Research, 10, 85-92.

Santos, Á., Bernardino, J., & Correia, N. (2023). *Automated application deployment on multi-access edge computing: A survey.* IEEE access.

Sen, A., Falter, S., & Mayer, N. (2022). *Using DevOps paradigm to deploy web applications*. Issues in Information Systems, 23(4).

Sevindik, G., (2015), *Yazılım geliştirme projelerine kritik zincir tabanlı çoklu proje planlama yaklaşımı*, (Yayımlanmamış Yüksek Lisans Tezi), İstanbul Teknik Üniversitesi, İstanbul

Shahin, M., Babar, M. A., & Zhu, L. (2017). *Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices.* IEEE access, 5, 3909-3943.

Shahin, M., Rezaei Nasab, A., & Ali Babar, M. (2023). *A qualitative study of architectural design issues in DevOps*. Journal of Software: Evolution and Process, 35(5), e2379.

Verona, J. (2016). Practical DevOps, Packt Publishing Ltd.

Wang, C., & Liu, C. (2018). *Adopting DevOps in Agile*, Challenges and Solutions. In.

Web Application Monitoring, Erişim tarihi: 08.02.2024, https://www.nagios.com/solutions/web-application-monitoring/

The world's most popular network protocol analyzer. Erişim tarihi: 08.02.2024, https://www.wireshark.org/

Xcode, Erişim tarihi: 08.02.2024, https://developer.apple.com/xcode/

BÖLÜM III

Büyük Dil Modeli Bazlı Otonom Ajanları Kullanan Uygulamalar Üzerine Bir Araştırma

Oğuz ALTUN[1]
Hamza Furkan ATMACA[2]

GİRİŞ

Büyük dil modelleri (BDM), doğal insan dillerini anlayabilme ve yazabilme konusunda eğitilmiş derin öğrenme modelleridir (Yilun Du, 2023). Bu modellerin ulaşılabilirlikleri ChatGPT, Sider gibi ara yüzü basit sohbet botlarının kullanıma sunulmasıyla birlikte artmıştır. Bu tarz sohbet botlarının doğal dilleri hızlı bir şekilde anlayabilme ve cevap verebilme yetenekleri, basit ara yüz ve kullanım kolaylığı ile birleşince kısa sürede geniş toplumlarca keşfedilmelerini sağlamıştır.

[1] Oğuz ALTUN, Dr. Öğr. Üyesi, Yıldız Teknik Üniversitesi, Bilgisayar Mühendisliği
[2] Hamza Furkan ATMACA, Yüksek Lisans Öğrencisi, Yıldız Teknik Üniversitesi, Bilgisayar Mühendisliği

ChatGPT'nin üreticisi OpenAI tarafından kullanıma sunulan OpenAI API, düşük kullanım ücretleri ile yazılım geliştiricilerin yapay zeka ile entegre yazılımlar geliştirebilmesini kolaylaştırmıştır. Bunun sonucunda BDM bazlı iş fikirleri artmıştır. Yenilikçi yazılım senaryoları API'lar aracılığıyla BDM kullanma eğilimi göstermişlerdir.

Büyük dil modelleri kullanılarak birçok sektörde işler kolaylaştırılabilmektedir. BDM'nin başarılı sonuçlar verdiği sektörlerden biri de mühendislik alanındaki yazılım geliştirme sektörüdür. Kendileri birer yazılım olan BDM bazlı uygulamaların harici yazılımlar geliştirebilme yetenekleri incelemeye değerdir. Yazılıma yazılım yaptırma fikri konuyu ayrıca ilgi çekici kılmaktadır. BDM bazlı sohbet botları, iş tanımı belli olmasına rağmen kodlanması zaman alan yazılım kesitlerini saniyeler içerisinde yazarak yazılım geliştirme işlemlerini hızlandırabilme özellikleriyle yazılım sektöründe ön plana çıkmaktadırlar.

BDM bazlı otonom ajanları kullanarak yazılım geliştirmeyi hedefleyen ChatDEV (Chen Qian, 2023), MetaGPT (Sirui Hong, 2023), AutoGEN (Qingyun Wu, 2023), AutoGPT (Hui Yang, 2023) gibi uygulamalar mevcuttur.

Otonom ajanlar yazılım geliştirme dışında da kullanılabilirler (Guohao Li, 2023). Örneğin otonom ajanlar ile oyunlar üzerinde de deneyler yapılmıştır (Guanzhi Wang, 2023).

Bu çalışmada BDM bazlı otonom ajanları kullanarak tarif edilen yazılımı derlenebilir bir biçimde kaynak kodlarıyla birlikte yazıp kullanıcıya vermeyi hedefleyen ChatDEV ve MetaGPT isimli iki uygulama üzerinde durduk. Bu uygulamalar, kendi içlerinde

sanal bir yazılım ekibi kurup gerçek bir yazılım ekibini simüle etmeyi, böylelikle kod kesitlerinden öte bütün bir yazılımı çıktı olarak vermeyi hedeflemektedirler. Uygulamalarda yazılım geliştirici, iş analisti, test mühendisi gibi rolleri üstlenen otonom ajanlar bulunmaktadır. Her bir ajan hem OpenAI API ile hem de kendi aralarında iletişim halindedir (Weize Chen, 2023).

Çalışmamızda, fikir olarak son derece ilgi çekici olan bu uygulamalar üzerinde yaptığımız deneylerle yöntemlerin verdikleri sonuçları yorumladık ve karşılaştırdık. Girdi olarak belirlediğimiz birbirinden farklı, basitten karmaşığa doğru giden 10 yazılım senaryosu kullandık. Her bir uygulamada bu 10 yazılım senaryosunu ayrı ayrı çalıştırdık. Toplamda 20 deney yaptık ve uygulama başına 10 olmak üzere 2 uygulama için toplamda 20 çıktı elde ettik. Çıktılar elde edilirken anlamlandırılabilecek ve aralarında karşılaştırma yapılabilecek verileri kayıt altına aldık.

İLGİLİ LİTERATÜR

Büyük dil modellerini (Muhammad Usman Hadi, 2023) (Yupeng Chang, 2023)daha etkin kullanmak amacıyla çeşitli uygulamalar ve yöntemler geliştirilmiştir. Bu uygulamalar, metin tabanlı problem çözme, doğal dil anlama, sanal asistanlar gibi alanlarda geniş bir yer tutmaktadırlar. Büyük dil modeli tabanlı uygulamaların sürekli olarak gelişmesi ve çeşitlenmesi, bu modellerin pratikteki etkinliğini artırmaktadır. Büyük dil modeli bazlı uygulamaların bu denli yoğun bir dağılımı olması ve hızla gelişmesi konuyu dikkat çekici kılmaktadır.

Büyük dil modellerini otonom ajanlar ile kullanmak olumlu sonuçlar vermektedir. Otonom ajanların iş birliği halinde BDM kullanması verimliliği artırmaktadır (Lei Wang, 2024).

Bu makalede, büyük dil modellerinin yazılım geliştirme sektöründeki varlığını güncel literatür üzerinden inceledik ve değerlendirdik (Xinyi Hou, 2024). Araştırma kapsamında ChatDEV ve MetaGPT uygulamalarını kullandık.

ChatDEV

ChatDEV (Chen Qian, 2023), yapay zeka yardımıyla yazılım geliştirmeyi amaçlayan büyük dil modeli bazlı çok ajanlı bir Python uygulamasıdır. Temelde gerçek bir yazılım ekibini simüle ederek kaliteli yazılımları düşük maliyetler ile kısa sürede üretmeyi amaçlar. Geleneksel bir yazılım geliştirme modeli olan şelale modelini baz alır. Şelale modelini tasarım, uygulama, test ve belgelendirme evrelerine göre ele alır. Bu evrelerde görev alan kişileri birer otonom ajan olarak simüle eder. CEO (İcra Kurulu Başkanı), CTO (Baş Teknoloji Yöneticisi), CPO (Baş Ürün Yöneticisi), Programmer (Programcı), Designer (Tasarımcı), Reviewer (İncelemeci) ve Tester (Testçi) olmak üzere yedi tanedir.

Ajanlar şelale modelinin evrelerine dağıtılmıştır. Tasarım evresinde CEO, CTO ve CPO; uygulama evresinde CTO, Programcı, Tasarımcı; test evresinde Programcı, İncelemeci, Testçi; belgelendirme evresinde ise CTO, Programcı, CEO ve CPO görev alırlar.

ChatDEV'i kullanmak isteyen kullanıcı, kendi OpenAI API anahtarını ChatDEV'e tanıttıktan sonra ihtiyaç duyduğu yazılımı cümleler ile tarif eder ve istediği yazılım ismini belirler. Bu verileri

konsol üzerinden ChatDEV'e parametre olarak verir. Böylelikle yazılım geliştirme süreci başlamış olur. Parametrelerle tarif edilen yazılım, şelale modeline dahil edilerek ajanlar tarafından OpenAI API ile iletişim halinde oluşturulmaya başlanır. Her bir ajan sırası geldiğinde kendisine atanan işi yapmaktadır. Programcının yazdığı kodları testçiye gönderip aldığı geri dönüşe göre sorunları çözmesi, incelemecinin tespit ettiği bulguları programcıya göndermesi gibi ajanlar arası iş birliği süreç boyunca devam eder.

MetaGPT

MetaGPT (Sirui Hong, 2023), yapay zeka yardımıyla yazılım geliştirmeyi amaçlayan çok ajanlı bir büyük dil modeli uygulamasıdır. ChatDEV ile benzerlikler içermektedir ancak ChatDEV'in yedi ajanına karşın MetaGPT, beş farklı rolde ajan ile çalışır. Bu ajanlar şunlardır:

Ürün Yöneticisi: İşletme odaklı analiz yapar ve bilgiler elde eder.

Mimar: Gereksinimleri sistem tasarım bileşenlerine çevirir.

Proje Yöneticisi: Görev dağıtımı yapar.

Mühendis: Belirtilen sınıfları ve fonksiyonları uygular.

Kalite Güvence Mühendisi: Kod kalitesini sağlamak için test senaryoları oluşturur.

Bu ajanlar, karmaşık görevleri daha küçük ve spesifik görevlere bölmek için kullanılır. Her bir ajanın belirli bir uzmanlık alanı ve görevi vardır. Ajanlar; planlama, gereksinim analizi, mimari tasarımı, sistem tasarımı, kodlama, test, kabul kontrolü süreçlerinde görev alırlar ve MetaGPT'nin iş akışını ve görev dağıtımını

belirlemek için kullanılırlar. MetaGPT'nin görece daha az ajan içermesi ve daha küçük bir yazılım ekibini simüle etmesi, sonuçlar üzerinde olumlu veya olumsuz nasıl farklar yarattığını görmek açısından ilgi çekicidir.

YÖNTEM

Deneyimiz için çeşitli yazılım senaryoları hazırladık. Bütün senaryolar bir kullanıcı ara yüzü içerir ve hiçbiri konsol uygulaması değildir. Kullanıcı ara yüzü olan senaryolar seçmemizdeki amaç tasarımcı ajanların da sürece dahil olmasıdır. Böylelikle bütün ajanların sürece dahil olmasını ve sonuçların yanıltıcılığını minimize etmeyi amaçladık.

Senaryolar, ekrana "Merhaba Dünya!" yazılı bir pencere açan basit bir yazılımdan başlar ve yılan oyunu yapımına kadar kademe kademe zorlaşarak ilerler. Bu kademeli yaklaşım, olası başarısız sonuçlarda hangi kademeden itibaren başarısız olunabildiğini belirleyebilmek için önemlidir. Her bir senaryoyu uygulamalara tarif edecek Türkçe cümleler hazırladık. Bu cümleler deneylerimizde girdi olarak kullanılacaktır. Cümleler, uygulamanın tamamını anlatır ancak bunu yaparken minimum düzeyde bilgi vermeyi amaçlar. Cümleler; kullanılacak teknolojiler, programlama dilleri, algoritmalar, kodlama standartları, kullanılacak veri tipleri gibi teknik düzeyde bilgi içermezler. Bu kararların ajanlar tarafından alınıp uygulanması beklenir. Böylelikle büyük dil modeli bazlı uygulamaların gerçek performansları elde edilebilir.

Merhaba Dünya

Basit düzeyde mesaj vermeyi amaçlayan bir uygulama.

Girdi: “Basit bir pencerede "Merhaba Dünya" yazısı gösterilir.”

Toplama İşlemi

Girilen iki sayısı toplayıp sonucu kullanıcıya göstermeyi amaçlayan bir uygulama.

Girdi: “İki giriş kutusu ve bir buton içeren bir pencere. Kullanıcı sayıları girer, ardından butona tıklar ve toplama işlemi gerçekleştirilip sonuç başka bir kutuda gösterilir.”

Çarpım Tablosu

Girilen bir sayının çarpım tablosundaki karşılığını göstermeyi amaçlayan bir uygulama.

Girdi: “Bir giriş kutusu ve bir buton içeren bir pencere. Kullanıcı bir sayı girer, ardından butona tıklar ve çarpım tablosu başka bir pencerede gösterilir.”

Asal Sayı Kontrolü

Girilen bir sayının asal sayı olup olmadığını kontrol etmeyi amaçlayan bir uygulama.

Girdi: “Bir giriş kutusu ve bir buton içeren bir pencere. Kullanıcı bir sayı girer, ardından butona tıklar ve sonuç (asal veya asal değil) başka bir pencerede gösterilir.”

Faktöriyel Hesaplama

Girilen bir sayının faktöriyel hesabını yapmayı amaçlayan bir uygulama.

Girdi: “Bir giriş kutusu ve bir buton içeren bir pencere. Kullanıcı bir sayı girer, ardından butona tıklar ve faktöriyel sonucu başka bir pencerede gösterilir.”

Karakter Sayma

Girilen bir metnin karakter sayısını bulmayı amaçlayan bir uygulama.

Girdi: “Bir metin kutusu ve bir buton içeren bir pencere. Kullanıcı metni girer, ardından butona tıklar ve karakter sayısı başka bir pencerede gösterilir.”

Basit Hesap Makinesi

Girilen iki sayı arasında seçilen dört işlemden birini yapıp sonucu vermeyi amaçlayan bir uygulama.

Girdi: “İki giriş kutusu, bir işlem seçme alanı ve bir hesapla butonu içeren bir pencere. Kullanıcı sayıları ve dört işlemden birini seçer, ardından hesapla butonuna tıklar ve sonuç başka bir kutuda gösterilir.”

Metin Ters Çevirme

Girilen bir metnin ters çevrilmesini amaçlayan bir uygulama.

Girdi: “Bir metin kutusu ve bir buton içeren bir pencere. Kullanıcı metni girer, ardından butona tıklar ve ters çevrilmiş metin başka bir pencerede gösterilir.”

Bilgi Tahmini Oyunu

Kullanıcının aklında tuttuğu bir sayıyı bilgisayarın tahmin etmesini amaçlayan bir uygulama.

Girdi:

“Ana Pencere:

Başlangıçta, kullanıcıya bir "Başarılar!” mesajı ve talimatlar içeren bir ana pencere gösterilir.

Kullanıcı "Devam" butonuna tıkladığında, yeni bir pencere açılır.

Oyun Penceresi:

Bu pencere, kullanıcının aklında tuttuğu bir sayıyı tahmin etmeye çalışan bilgisayarın sorularını gösterir.

Kullanıcıya sorular sorulur ve bu sorulara yanıt vermesi istenir. Örneğin:

"Sayınız 50'den büyük mü?" (E/H)

"Sayınız 75'ten büyük mü?" (E/H)

Kullanıcı, "E" veya "H" tuşlarına basarak sorulara cevap verir.

Kullanıcı "Devam" butonuna tıkladığında, yeni bir soru gösterilir.

Sonuç Penceresi:

Oyun bittiğinde, bir sonuç penceresi açılır.

Pencere, bilgisayarın tahmin ettiği sayıyı gösterir.

Ayrıca, kaç tahminde bulunduğu ve doğru tahminde bulunup bulunmadığı gibi istatistikleri içerir.

Kullanıcıya "Yeniden Oyna" veya "Çıkış" gibi seçenekler sunularak oyunu tekrar başlatması veya çıkması istenir.”

Yılan Oyunu

Ok tuşları ile yönlendirilebilen grafiksel bir yılan oyunu yapmayı amaçlayan uygulama.

Girdi: “Bir oyun penceresi içinde yılanın, yemin ve oyun alanının bulunduğu bir grafiksel yılan oyunu. Kullanıcı yılanı kontrol etmek için klavye ok tuşlarını kullanır. Amaç yılanın yemleri yemesi ve yedikçe büyümesidir. Yılan bir kenardan çıkarsa karşı kenardan pencereye geri girer. Yılan kendi kuyruğunu yerse oyun sonlanır. Puan başlangıçta 0’dır. Yılan her yem yediğinde puan 1 artar. Oyunun sonunda puan başka bir pencerede gösterilir.”

DENEY SÜRECİ

ChatDEV ve MetaGPT’yi klonlayıp kendimize ait OpenAI API anahtarını uygulamalara tanımladık. ChatDEV tarafında projenin kaynak kodları içine, MetaGPT tarafında ortam değişkeni olarak bu anahtarı tanımladık. Belirlenen 10 yazılım senaryosunu ChatDEV ve MetaGPT uygulamalarına girdi olarak verdik. Sonuçları değerlendirmek ve yorumlamak üzere kaydettik.

ChatDEV Deney Süreci

Klonladığımız ChatDEV uygulamasının bulunduğu klasörde aşağıdaki standart terminal komutunu çalıştırarak uygulamalarımızın yapay zeka tarafından geliştirilmesini sağladık.

python3 run.py --task "Uygulamayı tarif eden metin" --name "UygulamaAdi"

ChatDEV, komutu gereği geliştirilecek uygulamanın ismini belirlememize olanak sağlıyordu. Oluşan uygulama chatdev/warehouse klasörü altında

UygulamaAdi_DefaultOrganization_YYYYMMDDhhmmss standart klasör ismiyle oluşuyordu. Yeterince açık olan bu klasör ismi standardına müdahalede bulunmadık.

ChatDEV, harcanan zaman ve maliyet gibi değerleri kendisi çıktı olarak veriyordu. Bu tarz yorumlanabilir anlamlı verileri kaydetmek için ek bir müdahalede bulunmadık.

MetaGPT Deney Süreci

Klonlanan MetaGPT uygulamasının bulunduğu klasörde aşağıdaki standart terminal komutunu çalıştırarak uygulamalarımızın yapay zeka tarafından geliştirilmesini sağladık.

metagpt "Uygulamayı tarif eden metin"

MetaGPT, komutu gereği geliştirilecek uygulamanın ismini belirlememize olanak sağlamıyordu. Oluşan uygulama metagpt/workspace klasörü altında YYYYMMDDhhmmss standart klasör ismiyle oluşuyordu. Uygulama ismini içermeyen bu klasör ismi standardına müdahalede bulunduk ve her bir uygulama için UygulamaAdi_YYYYMMDDhhmmss standart ismiyle klasörleri yeniden isimlendirdik.

MetaGPT, harcanan zaman değerini kendisi çıktı olarak vermiyordu. Bu anlamlı veriyi kaydedebilmek için komutun çalışmaya başladığı ilk zaman değeri ile komutun sonlandığı son zaman değeri arasındaki farkı hesaplayıp her bir uygulama altında oluşturduğumuz bir metin dosyasının içine kaydettik.

DENEYSEL SONUÇLAR

Girdilerimizde programlama dili belirtmedik. Programlama dili seçimini sistemlere bıraktık. Üretilen bütün yazılımlarda

sistemler tarafından Python programlama dili seçildi. Üretilen toplam 20 adet yazılımı ayrı ayrı çalıştırdık ve test ettik. Ekran görüntülerini kaydettik.

ChatDEV Deneysel Sonuçlar

ChatDEV tarafından üretilen 10 adet yazılımı ayrı ayrı derledik, çalıştırdık ve test ettik. 9 tanesi sorunsuz çalışırken, 1 tanesinde derleme hatası görüldü. Çalışan 9 uygulamanın 8'inde başlıklar ve metinler anlamlıydı. 2 uygulamanın ara yüz dili Türkçe, 7 uygulamanın ara yüz dili İngilizceydi.

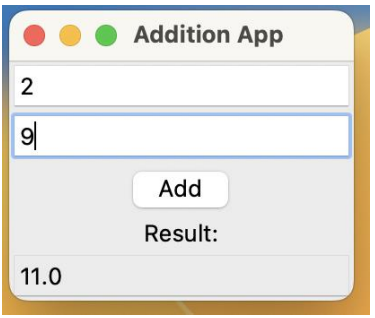
ChatDEV Merhaba Dünya

ChatDEV, tarif edilen uygulamayı başarıyla yazdı. Derleme hatası alınmadı. Uygulamayı çalıştırdığımızda Türkçe başlık ve ara yüzle "Merhaba Dünya" yazılı bir pencere açıldığını gördük. Metinler ve başlıklar anlamlıydı.

Görsel 1. ChatDEV tarafından üretilen Merhaba Dünya uygulamasının ekran görüntüsü.

ChatDEV Toplama İşlemi

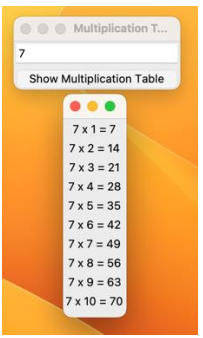
ChatDEV, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme hatası alınmadı. Uygulama, girdi olarak girilen sayılar üzerinde toplama işlemini yapıp sonucu çıktı olarak verdi. Uygulama dili İngilizceydi. Metinler ve başlıklar anlamlıydı.



Görsel 2. ChatDEV tarafından üretilen Toplama İşlemi uygulamasının ekran görüntüsü.

ChatDEV Çarpım Tablosu

ChatDEV, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme hatası alınmadı. Uygulama, girdi olarak girilen sayının çarpım tablosunu çıktı olarak verdi. Uygulama dili İngilizceydi. Metinler ve başlıklar anlamlıydı.

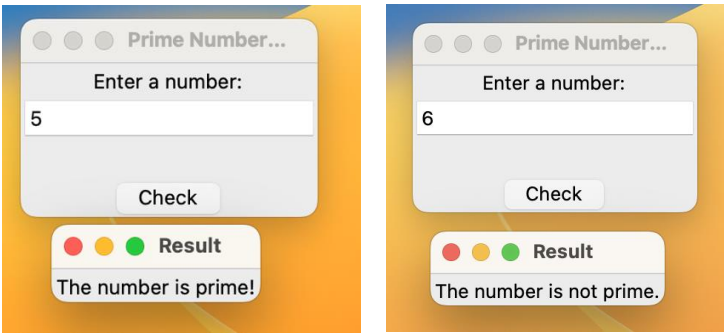


Görsel 3. ChatDEV tarafından üretilen Çarpım Tablosu uygulamasının ekran görüntüsü.

ChatDEV Asal Sayı Kontrolü

ChatDEV, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme

hatası alınmadı. Uygulama, girdi olarak girilen sayının asal sayı olup olmadığını çıktı olarak verdi. Uygulama dili İngilizceydi. Metinler ve başlıklar anlamlıydı.



Görsel 4. ChatDEV tarafından üretilen Asal Sayı Kontrolü uygulamasının ekran görüntüleri.

ChatDEV Faktöriyel Hesaplama

ChatDEV, bu uygulamada müdahalesiz çalıştırılabilir uygulama üretme konusunda başarısız oldu. Sorun, uygulamaya Flask'in dahil edilmiş olması ancak yayın moduna göre düzenleme yapılmamış olmasıydı.

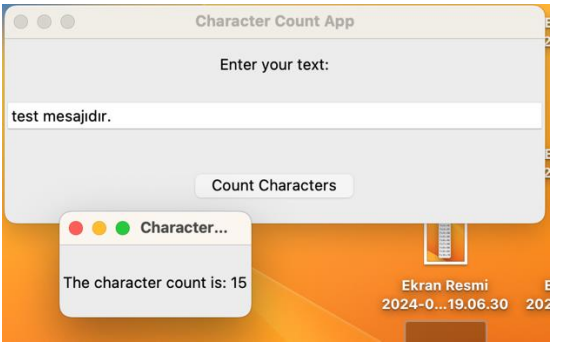
```
 * Serving Flask app 'main'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 514-276-865
```

Görsel 5. ChatDEV tarafından üretilen Faktöriyel Hesaplama uygulamasında alınan hatanın ekran görüntüsü.

ChatDEV Karakter Sayma

ChatDEV, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme hatası alınmadı. Uygulama, girdi olarak girilen metnin kaç karakter

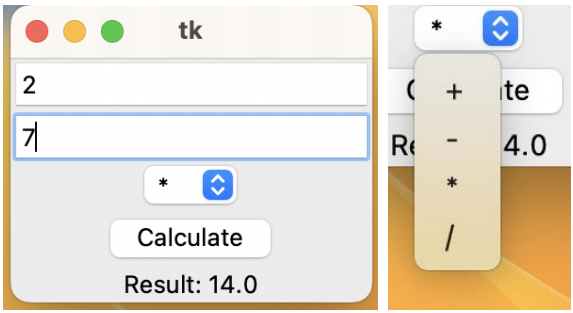
içerdiğini çıktı olarak verdi. Uygulama dili İngilizceydi. Metinler ve başlıklar anlamlıydı.



Görsel 6. ChatDEV tarafından üretilen Karakter Sayma uygulamasının ekran görüntüsü.

ChatDEV Hesap Makinesi

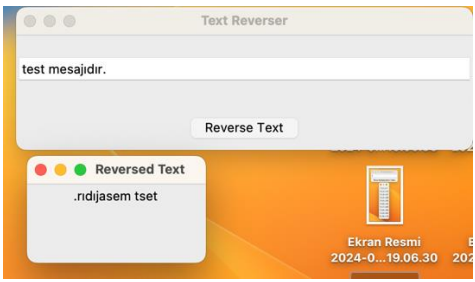
ChatDEV, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme hatası alınmadı. Uygulama, girdi olarak girilen sayılar arasında seçilen işlemi yapıp sonucu çıktı olarak verdi. Her dört işlem için de testler yapıldı, hesap sonuçları doğruydu. Uygulama dili İngilizceydi. Uygulama başlığı konudan bağımsız olarak "tk" idi. Kaynak kodları incelediğimizde "tk" ifadesinin ara yüz oluşturmak için kullanılan "Tkinter" kütüphanesinden dolayı geldiğini düşündük.



Görsel 7. ChatDEV tarafından üretilen Hesap Makinesi uygulamasının çarpma işlemi ve işlem seçimi için ekran görüntüleri.

ChatDEV Metin Ters Çevirme

ChatDEV, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme hatası alınmadı. Uygulama, girdi olarak girilen metnin ters çevrilmiş halini çıktı olarak verdi. Uygulama dili İngilizceydi. Metinler ve başlıklar anlamlıydı.



Görsel 8. ChatDEV tarafından üretilen Metin Ters Çevirme uygulamasının ekran görüntüsü.

ChatDEV Bilgi Tahmini Oyunu

ChatDEV, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme hatası alınmadı. Uygulama, kullanıcıya bir sayı tutmasını

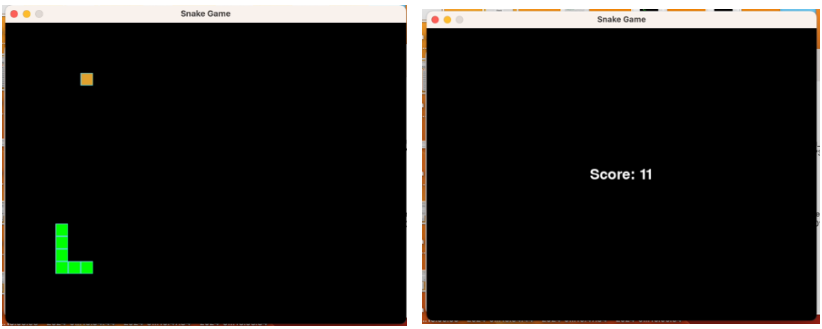
söyledikten sonra sorular sorup kullanıcının tuttuğu sayıyı tahmin ediyor, tahmin ettiği sayıyı ve işlemin kaç adım sürdüğünü söyleyen bir pencereyle sonucu gösteriyordu. Uygulama dili Türkçeydi. Butonlar işlevlerini yerine getiriyordu. Metinler ve başlıklar anlamlıydı.

Görsel 9. ChatDEV tarafından üretilen Bilgi Tahmini Oyunu uygulamasının başlangıç, soru örneği ve sonuç için ekran görüntüleri.

ChatDEV Yılan Oyunu

ChatDEV, tarif edilen oyunun fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme hatası alınmadı. Uygulama dili İngilizceydi. Metinler ve başlıklar anlamlıydı.

Oyun, tarif edildiği gibi çalışıyordu. Başlangıçta bir birim uzunluğunda olan yılan yemden her geçtiğinde bir birim büyüyordu. Yem, her seferinde rastgele bir konumda yeniden beliriyordu. Yılan, kenarlardan geçtiğinde karşı kenardan tekrar çıkıyordu. Dört kenar için de testler yapıldı, sorun görülmedi. Yılan kendi kuyruğuna çarpana kadar oyun devam ediyordu ve oyun sonunda puanı gösteren bir pencere açılıyordu. Puan, 0'dan başlıyor, her yem yenildiğinde 1 artıyordu.



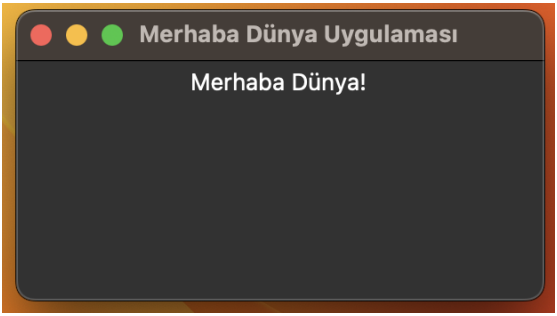
Görsel 10. ChatDEV tarafından üretilen Yılan Oyunu uygulamasının ekran görüntüleri.

MetaGPT Deneysel Sonuçlar

MetaGPT tarafından üretilen 10 adet yazılımı ayrı ayrı derledik, çalıştırdık ve test ettik. 7 tanesi sorunsuz çalışırken, 2 tanesinde derleme, 1 tanesinde algoritma hatası görüldü. Çalışan 8 uygulamanın 8'inde başlıklar ve metinler anlamlıydı. 4 uygulamanın ara yüz dili Türkçe, 4 uygulamanın ara yüz dili İngilizceydi.

MetaGPT Merhaba Dünya

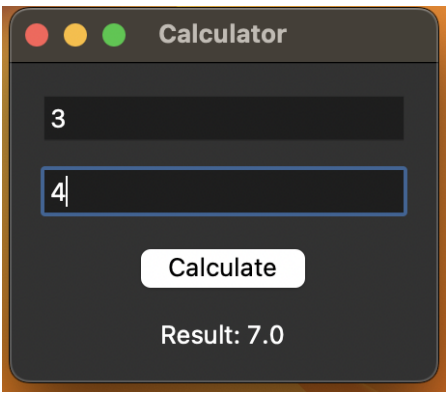
MetaGPT, tarif edilen uygulamayı başarıyla yazdı. Derleme hatası alınmadı. Uygulamayı çalıştırdığımızda Türkçe başlık ve ara yüzle "Merhaba Dünya" yazılı bir pencere açıldığını gördük.



Görsel 11. MetaGPT tarafından üretilen Merhaba Dünya uygulamasının ekran görüntüsü.

MetaGPT Toplama İşlemi

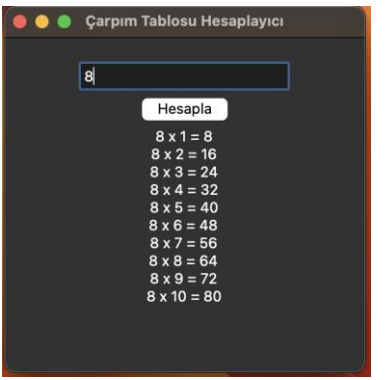
MetaGPT, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme hatası alınmadı. Uygulama, girdi olarak girilen sayılar üzerinde toplama işlemini yapıp sonucu çıktı olarak verdi. Uygulama dili İngilizceydi. Metinler ve başlıklar anlamlıydı.



Görsel 12. MetaGPT tarafından üretilen Toplama İşlemi uygulamasının ekran görüntüsü.

MetaGPT Çarpım Tablosu

MetaGPT, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme hatası alınmadı. Uygulama, girdi olarak girilen sayının çarpım tablosunu çıktı olarak verdi. Uygulama dili Türkçeydi. Metinler ve başlıklar anlamlıydı.



Görsel 13. MetaGPT tarafından üretilen Çarpım Tablosu uygulamasının ekran görüntüsü.

MetaGPT Asal Sayı Kontrolü

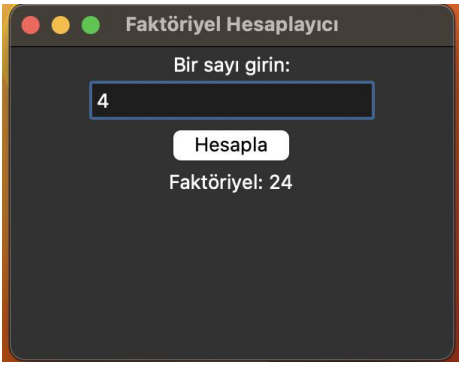
MetaGPT, bu uygulamada müdahalesiz çalıştırılabilir uygulama üretme konusunda başarısız oldu. Sorun, uygulamaya dahil edilen sympy modülünün tanınmıyor olmasıydı. Bu modül, requirements.txt dosyasına da yazılmamıştı.

```
(base) hamzafurkanatmaca@Hamza-MacBook-Air 20240308013108 % python3 main.py
Traceback (most recent call last):
  File "/Users/hamzafurkanatmaca/Desktop/tez/metagpt/AsalSayiKontrolu_2024030801
3108/20240308013108/main.py", line 3, in <module>
    from prime_checker import PrimeChecker
  File "/Users/hamzafurkanatmaca/Desktop/tez/metagpt/AsalSayiKontrolu_2024030801
3108/20240308013108/prime_checker.py", line 2, in <module>
    from sympy import isprime
ModuleNotFoundError: No module named 'sympy'
```

Görsel 14. MetaGPT tarafından üretilen Asal Sayı Kontrolü uygulamasında alınan hatanın ekran görüntüsü.

MetaGPT Faktöriyel Hesaplama

MetaGPT, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme hatası alınmadı. Uygulama, girdi olarak girilen sayının faktöriyel sonucunu çıktı olarak verdi. Uygulama dili Türkçeydi. Metinler ve başlıklar anlamlıydı.



Görsel 15. MetaGPT tarafından üretilen Faktöriyel Hesaplama uygulamasının ekran görüntüsü.

MetaGPT Karakter Sayma

MetaGPT, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme hatası alınmadı. Uygulama, girdi olarak girilen metnin karakter sayısını çıktı olarak verdi. Uygulama dili Türkçeydi. Metinler ve başlıklar anlamlıydı.

Görsel 16. MetaGPT tarafından üretilen Karakter Sayma uygulamasının ekran görüntüsü.

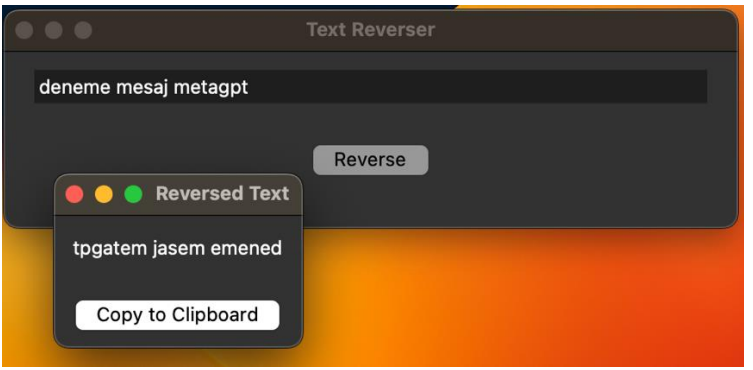
MetaGPT Hesap Makinesi

MetaGPT, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Girdi olarak girilen sayılar arasında seçilen işlemi yapıp sonucu çıktı olarak başarıyla verdi. Her dört işlem için de testler yapıldı, hesap sonuçları doğruydu. Uygulama dili İngilizceydi. Metinler ve başlıklar anlamlıydı. Her bir işlem birer butona bağlanmıştı. Her bir işlemin birer butona bağlanması kullanım kolaylığı sağlamaktaydı.

Görsel 17. MetaGPT tarafından üretilen Hesap Makinesi uygulamasının her dört işlem için ekran görüntüleri.

MetaGPT Metin Ters Çevirme

MetaGPT, tarif edilen uygulamanın fonksiyonel gereksinimlerini karşılayan bir yazılım üretmeyi başardı. Derleme hatası alınmadı. Uygulama, girdi olarak girilen metnin ters çevrilmiş halini çıktı olarak verdi. Uygulama dili İngilizceydi. Metinler ve başlıklar anlamlıydı. Sonuç penceresinde ters çevrilmiş metni kopyalamaya yarayan bir buton vardı.



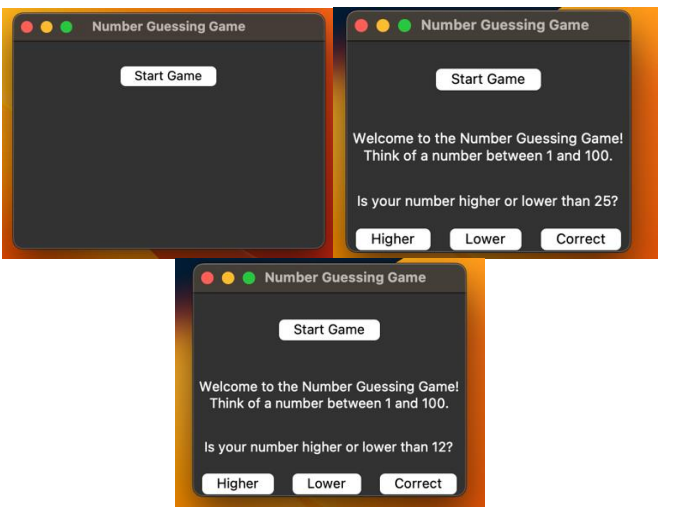
Görsel 18. MetaGPT tarafından üretilen Metin Ters Çevirme uygulamasının ekran görüntüsü.

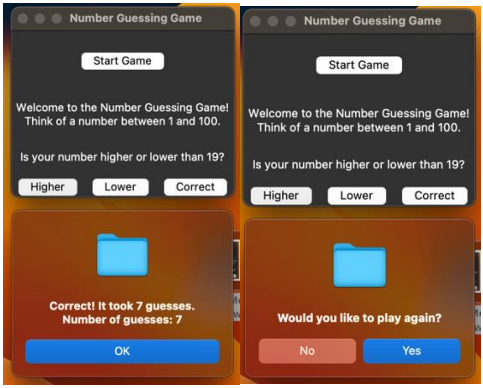
MetaGPT Bilgi Tahmini Oyunu

MetaGPT, tarif edilen uygulamanın fonksiyonel gereksinimlerini tam olarak karşılamayan bir yazılım üretti. Derleme hatası alınmadı. Algoritmada hatalar vardı. Uygulama dili İngilizceydi. Metinler ve başlıklar anlamlıydı.

Uygulama, “Start Game” butonunu içeren bir pencere ile başladı. Butona basıldığında, mevcut pencereye kullanıcıya aklından 1 ile 100 arasında bir sayı tutmasını söyleyen bir metin ve “Higher”, “Lower” ve “Correct” şeklinde üç buton yerleşti. Kullanıcıdan aklından tuttuğu sayıyı girmesi istenen bir pencere açıldı. Bu sayının

bilgisayarın tahmininin doğru olup olmadığını anlaması amacıyla istenildiği düşünülse de sayının daha sonra kullanılmadığı görüldü. Sayı girilip "OK" butonuna basıldıktan sonra pencerenin kaybolmasına rağmen kullanıcıya herhangi bir soru sorulmadığı görüldü. Soruların başlaması için "Higher" veya "Lower" butonlarından birinin kullanılması gerekliydi. Butonlardan birine basıldıktan sonra sorular sorulup cevaplandıkça yeni sorular sorulsa da kullanıcının aklından tuttuğu sayı kesin olarak anlaşılmadan "Correct" penceresinin açıldığı görüldü. Bu pencerede sayı tahmin etme işleminin kaç adım sürdüğü bilgisi yer alıyordu ancak tahmin edilen sayı yer almıyordu. Ayrıca bu pencerenin açıldığı anda bilgisayarın tahmininin kesinleşmiş olma ihtimali yoktu. Oyun sonunda açılan yeniden oynamak ister misiniz penceresinde "evet" seçilince oyun kapanıyordu. Ana pencerede bulunan "Higher" ve "Lower" butonları anlamlı olsa da "Correct" butonu anlamsızdı. Bilgisayarın tahmini belli olmadan kullanıcının "Correct" butonuyla oyunu bitirebilmesi gereksizdi.





Görsel 19. MetaGPT tarafından üretilen Bilgi Tahmini Oyunu uygulamasının ekran görüntüleri.

MetaGPT Yılan Oyunu

MetaGPT, bu oyunda müdahalesiz çalıştırılabilir yazılım üretme konusunda başarısız oldu. Sorun, uygulamaya dahil edilmeden kullanılmaya çalışılan bir yazı tipinden kaynaklanıyordu. Bu yazı tipi, requirements.txt dosyasına da yazılmamıştı.

```
(base) hamzafurkanatmaca@Hamza-MacBook-Air 20240308021527 % python3 main.py
pygame 2.5.2 (SDL 2.28.3, Python 3.11.5)
Hello from the pygame community. https://www.pygame.org/contribute.html
Traceback (most recent call last):
  File "/Users/hamzafurkanatmaca/Desktop/tez/metagpt/YilanOyunu_20240308021527/20240308021527/main.py", line 19, in <module>
    main()
  File "/Users/hamzafurkanatmaca/Desktop/tez/metagpt/YilanOyunu_20240308021527/20240308021527/main.py", line 12, in main
    game_window = GameWindow(DEFAULT_WIDTH, DEFAULT_HEIGHT, DEFAULT_TITLE)
                  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/Users/hamzafurkanatmaca/Desktop/tez/metagpt/YilanOyunu_20240308021527/20240308021527/game.py", line 18, in __init__
    self.score = Score()
                 ^^^^^^^
  File "/Users/hamzafurkanatmaca/Desktop/tez/metagpt/YilanOyunu_20240308021527/20240308021527/score.py", line 16, in __init__
    self.font = pygame.font.SysFont(Score.FONT_NAME, Score.FONT_SIZE)
                ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/Users/hamzafurkanatmaca/miniconda3/lib/python3.11/site-packages/pygame/sysfont.py", line 462, in SysFont
    return constructor(fontname, size, set_bold, set_italic)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/Users/hamzafurkanatmaca/miniconda3/lib/python3.11/site-packages/pygame/sysfont.py", line 380, in font_constructor
    font = Font(fontpath, size)
           ^^^^^^^^^^^^^^^^^^^^
pygame.error: font not initialized
(base) hamzafurkanatmaca@Hamza-MacBook-Air 20240308021527 %
```

Görsel 20. MetaGPT tarafından üretilen Yılan Oyunu uygulamasında alınan hatanın ekran görüntüsü.

SONUÇLAR

Üretilen her bir yazılım için sonuçları ayrı ayrı yorumladık. Zaman ve maliyet gibi nesnel verileri inceledik. Sonuçlar, BDM bazlı uygulamaların yakın gelecekte profesyonel yazılımlar geliştirme konusunda kritik rol oynayacağını düşündürdü.

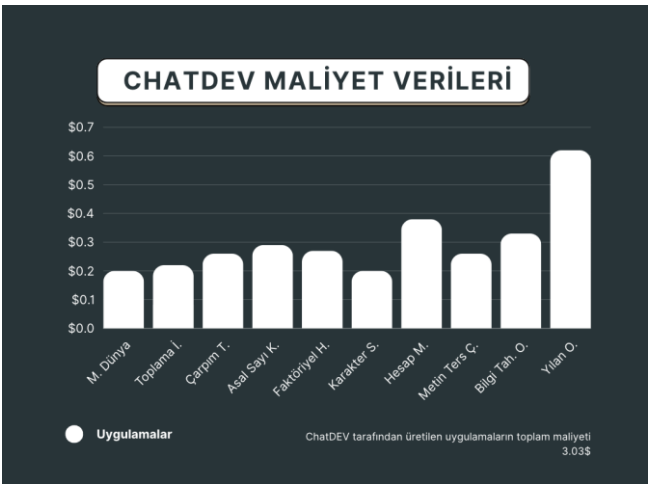
ChatDEV Sonuçlar

ChatDEV tarafında başarılı sonuçlar elde ettik. 10 adet deney senaryosunun toplam gerçekleme maliyeti 3.03$ idi. En düşük maliyetli uygulamalar 0.20$ ile "Merhaba Dünya" ve "Karakter Sayma" oldu. Maliyeti en yüksek uygulama ise 0.62$ ile "Yılan Oyunu" oldu. Hiçbir senaryo tek başına 1$ maliyete ulaşmadı.

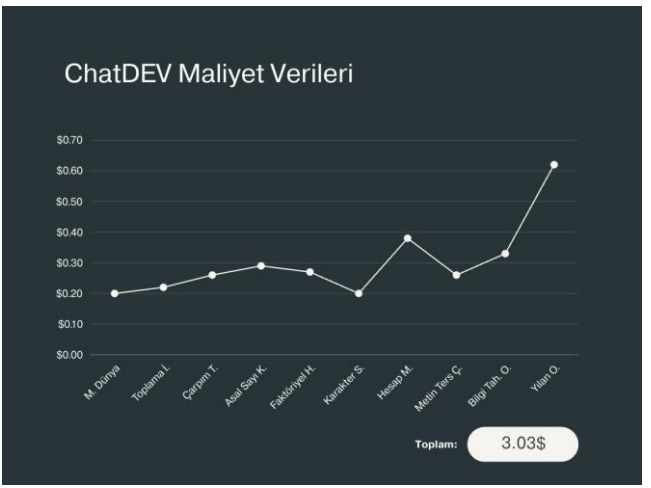
Zaman konusunda da sonuçlar umut vericiydi. 10 adet deney senaryosunun aldığı toplam zaman 34 dakika 38 saniyeydi. Geliştirilmesi en az zaman alan uygulama 2 dakika 42 saniye ile "Merhaba Dünya" uygulamasıydı. En çok zaman alan uygulama ise 4 dakika 45 saniye ile "Yılan Oyunu" oldu. Hiçbir senaryonun geliştirilmesi tek başına 5 dakikaya ulaşmadı.

ChatDEV Maliyet Değerlendirmesi

ChatDEV tarafında uygulamalar üretilirken "Merhaba Dünya" için 0.20$, "Toplama İşlemi" için 0.22$, "Çarpım Tablosu" için 0.26$, "Asal Sayı Kontrolü" için 0.29$, "Faktöriyel Hesaplama" için 0.27$, "Karakter Sayma" için 0.20$, "Hesap Makinesi" için 0.38$, "Metin Ters Çevirme" için 0.26$, "Bilgi Tahmini Oyunu" için 0.33$ ve "Yılan Oyunu" için 0.62$ değerinde OpenAI API kullanıldı. 10 senaryo toplamda 3.03$ maliyet oluşturdu.



Grafik 1. ChatDEV tarafından üretilen uygulamaların maliyet verilerini içeren sütun grafiği.

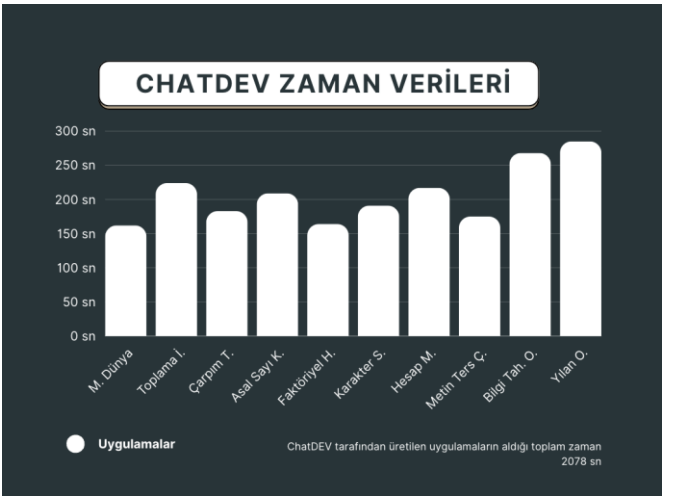


Grafik 2. ChatDEV tarafından üretilen uygulamaların maliyet verilerini içeren çizgi grafiği.

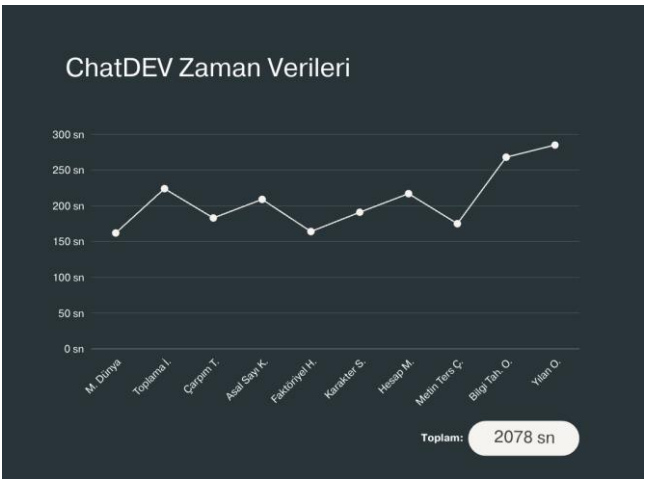
ChatDEV Zaman Değerlendirmesi

ChatDEV tarafında uygulamaların üretimi "Merhaba Dünya" için 162 saniye, "Toplama İşlemi" için 224 saniye, "Çarpım

Tablosu” için 183 saniye, “Asal Sayı Kontrolü” için 209 saniye, “Faktöriyel Hesaplama” için 164 saniye, “Karakter Sayma” için 191 saniye, “Hesap Makinesi” için 217 saniye, “Metin Ters Çevirme” için 175 saniye, “Bilgi Tahmini Oyunu” için 268 saniye ve “Yılan Oyunu” için 285 saniye zaman aldı. 10 senaryo toplamda 2078 saniye zaman aldı.



Grafik 3. ChatDEV tarafından üretilen uygulamaların zaman verilerini içeren sütun grafiği.



Grafik 4. ChatDEV tarafından üretilen uygulamaların zaman verilerini içeren çizgi grafiği.

MetaGPT Sonuçlar

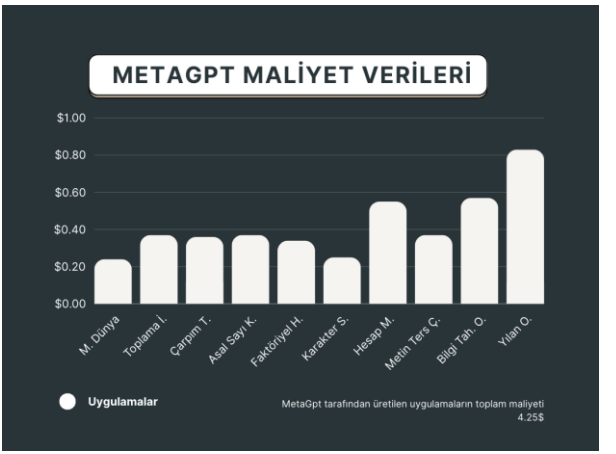
MetaGPT tarafında başarılı sonuçlar elde ettik. 10 adet deney senaryosunun toplam gerçekleme maliyeti 4.25$ idi. En düşük maliyetli uygulama 0.24$ ile "Merhaba Dünya" oldu. Maliyeti en yüksek uygulama ise 0.83$ ile "Yılan Oyunu" oldu. Hiçbir senaryo tek başına 1$ maliyete ulaşmadı.

Zaman konusunda da sonuçlar başarılıydı. 10 adet deney senaryosunun aldığı toplam zaman 46 dakika 8 saniyeydi. Geliştirilmesi en az zaman alan uygulama 1 dakika 24 saniye ile "Merhaba Dünya" uygulamasıydı. En çok zaman alan uygulama ise 10 dakika 6 saniye ile "Bilgi Tahmini Oyunu" oldu. Hiçbir senaryonun geliştirilmesi tek başına 15 dakikaya ulaşmadı.

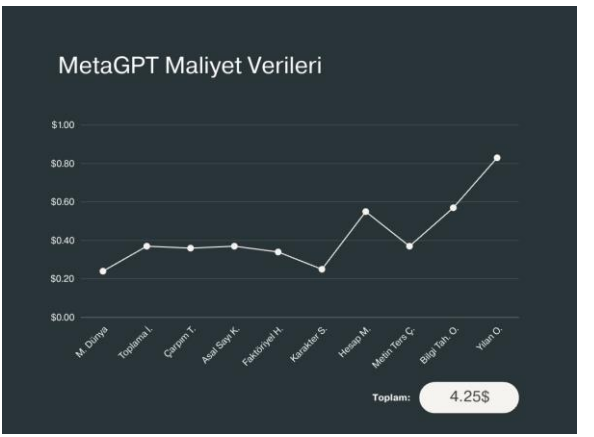
MetaGPT Maliyet Değerlendirmesi

MetaGPT tarafında uygulamalar üretilirken "Merhaba Dünya" için 0.24$, "Toplama İşlemi" için 0.37$, "Çarpım Tablosu"

için 0.36$, “Asal Sayı Kontrolü” için 0.37$, “Faktöriyel Hesaplama” için 0.34$, “Karakter Sayma” için 0.25$, “Hesap Makinesi” için 0.55$, “Metin Ters Çevirme” için 0.37$, “Bilgi Tahmini Oyunu” için 0.57$ ve “Yılan Oyunu” için 0.83$ değerinde OpenAI API kullanıldı. 10 senaryo toplamda 4.25$ maliyet oluşturdu.



Grafik 5. MetaGPT tarafından üretilen uygulamaların maliyet verilerini içeren sütun grafiği.

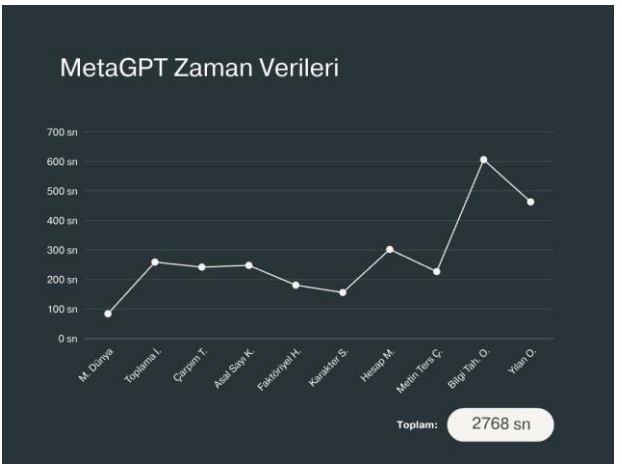


Grafik 6. MetaGPT tarafından üretilen uygulamaların maliyet verilerini içeren çizgi grafiği.

MetaGPT Zaman Değerlendirmesi

MetaGPT tarafında uygulamaların üretimi “Merhaba Dünya” için 84 saniye, “Toplama İşlemi” için 259 saniye, “Çarpım Tablosu” için 242 saniye, “Asal Sayı Kontrolü” için 248 saniye, “Faktöriyel Hesaplama” için 181 saniye, “Karakter Sayma” için 156 saniye, “Hesap Makinesi” için 302 saniye, “Metin Ters Çevirme” için 227 saniye, “Bilgi Tahmini Oyunu” için 606 saniye ve “Yılan Oyunu” için 463 saniye zaman aldı. 10 senaryo toplamda 2768 saniye zaman aldı.

Grafik 7. MetaGPT tarafından üretilen uygulamaların zaman verilerini içeren sütun grafiği.



Grafik 8. MetaGPT tarafından üretilen uygulamaların zaman verilerini içeren çizgi grafiği.

ChatDEV ve MetaGPT Karşılaştırma Sonuçları

ChatDEV ve MetaGPT'yi çeşitli açılardan karşılaştırarak yorumladık. Genel bakışta ChatDEV %90 oranında çalıştırılabilir uygulama üretti. MetaGPT'de bu oran %80 idi.

ChatDEV, Türkçe girdilerle %22 oranında Türkçe, %78 oranında İngilizce uygulama üretti. MetaGPT tarafında %50 Türkçe, %50 İngilizce çıktı alındı. Bu sonuçlardan yola çıkarak ChatDEV'in İngilizce uygulama üretmeye daha yatkın olduğunu düşündük.

Üretilen Uygulamaların Karşılaştırılması ve Yorumlar

"Merhaba Dünya" uygulamasında bir fark yoktu. Her iki uygulama da istenilen metni içeren bir pencere açıyordu.

"Toplama İşlemi" uygulamasında buton metinleri ve başlıklar dışında bir fark yoktu. Metinler ve başlıklar her iki uygulamada da anlamlıydı.

“Çarpım Tablosu” uygulamasında ChatDEV’in sonuçları bir pencerede göstermesi, MetaGPT’nin aynı pencereye yazması dışında bir fark yoktu.

“Asal Sayı Kontrolü” uygulamasında ChatDEV başarılı şekilde çıktı verirken MetaGPT tarafında uygulama derleme hatası veriyordu.

“Faktöriyel Hesaplama” uygulaması için durum tam tersiydi. MetaGPT tarafı sorunsuz çalışırken ChatDEV tarafında hata aldık.

“Karakter Sayma” uygulamasında kullanımı etkileyecek farklar yoktu. Bu uygulamada da “Çarpım Tablosu” uygulamasında olduğu gibi ChatDEV sonuç için pencere açarken, MetaGPT aynı pencereye yazmayı seçti.

“Hesap Makinesi” uygulamasında ChatDEV’in yapılacak işlemi seçtirdikten sonra bir buton aracılığıyla hesap yapmasına karşın MetaGPT’nin her işlemi birer buton yapması dikkat çekiciydi. Burada ChatDEV’in yeni pencere, sayfa veya seçim kutuları açtırmaktan çekinmediğini, MetaGPT’nin mümkün mertebe işlevleri bir arada sunmayı tercih ettiğini düşündük. Bu uygulama özelinde MetaGPT, daha kullanıcı dostu bir tasarım yaptı.

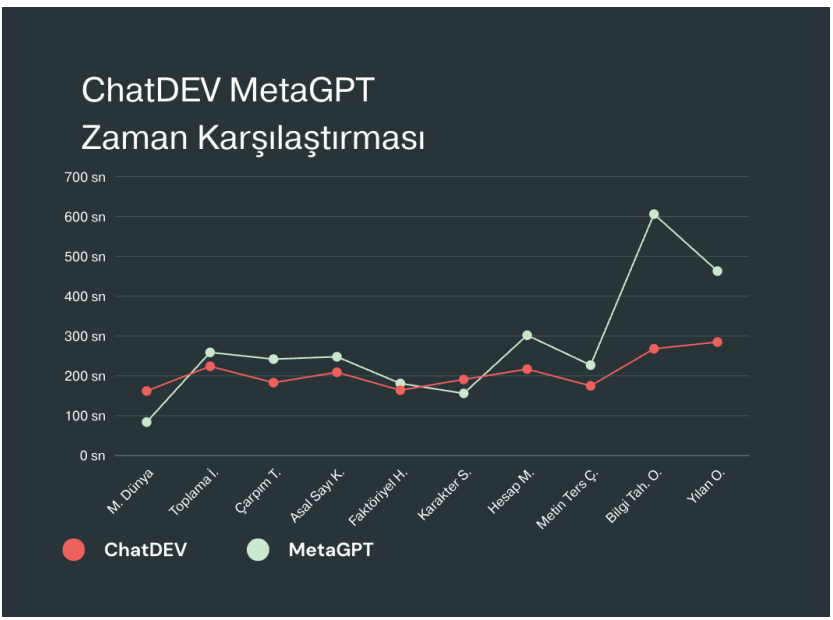
“Metin Ters Çevirme” uygulamasında her iki taraf da istenileni yapıyordu. Tek fark MetaGPT’nin ters çevrilmiş metni kopyalamak için buton eklemiş olmasıydı. Buradan BDM’nin tarif edilenin dışında kendi fikirleriyle işe yarayabilecek geliştirmeler ekleyebileceğini anlayabiliriz.

“Bilgi Tahmini Oyunu” tarafında ChatDEV istenileni tam olarak karşılarken MetaGPT’nin algoritmasında hatalar vardı. Ayrıca MetaGPT tarafında bir adet anlamsız buton vardı.

“Yılan Oyunu” tarafında ChatDEV istenileni tam olarak karşılarken MetaGPT’nin ürettiği oyun çalıştırılamıyordu. Son iki deneyden yola çıkarak talep karmaşıklaştıkça MetaGPT’nin başarı oranının düştüğü düşünülebilir.

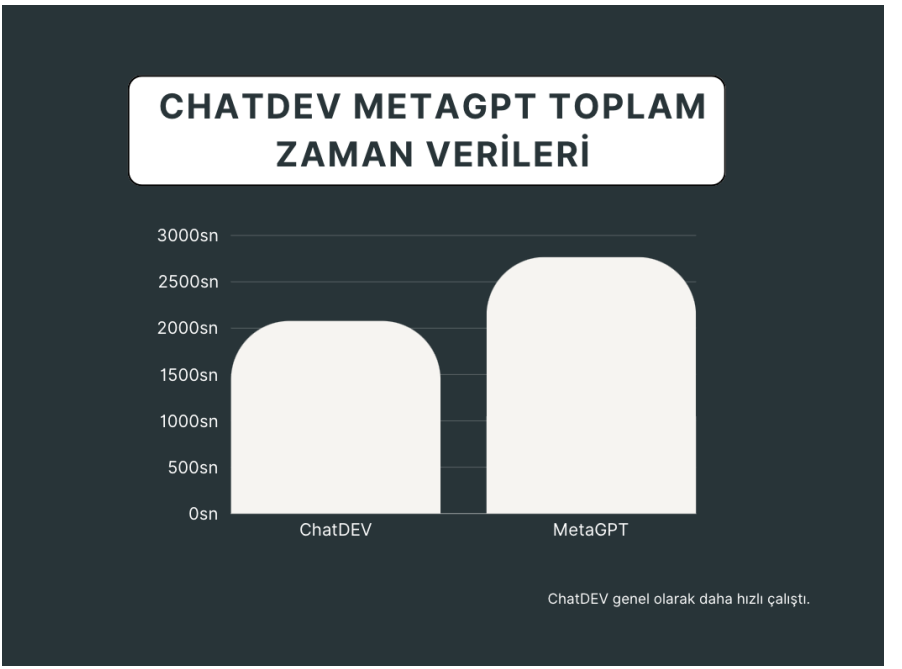
Genel olarak sanal ekip yapısı değiştikçe bazı kabiliyetlerde avantaj, bazı kabiliyetlerde dezavantaj oluştuğu söylenebilir. Sonuç olarak, ekip yapılarının davranış biçimlerini anlama ve optimum ekipler kurma konularında kapsamlı çalışmalara ihtiyaç vardır.

Grafikler ile Karşılaştırma ve Yorumlar

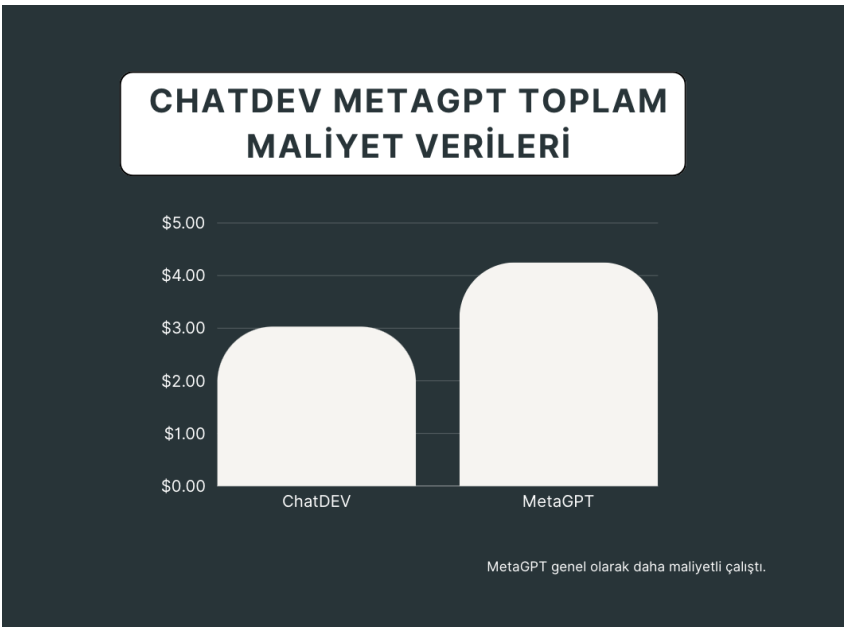


Grafik 9. ChatDEV ve MetaGPT’nin zaman verileriyle karşılaştırılmasını içeren çizgi grafiği.

Grafikte de görüldüğü üzere ChatDEV tarafında zaman verilerinin daha stabil olduğunu gördük. MetaGPT tarafında basit uygulamalar daha hızlı geliştirilmekte olup, karmaşıklık arttıkça zaman ihtiyacı da artmaktadır.



Grafik 10. ChatDEV ve MetaGPT'nin toplam zaman verileriyle karşılaştırılmasını içeren sütun grafiği.



Grafik 11. ChatDEV ve MetaGPT'nin toplam maliyet verileriyle karşılaştırılmasını içeren sütun grafiği.

ChatDEV, uygulamaların üretiminde toplamda daha az zaman ve daha düşük maliyetle çalıştı.

SONUÇ

Çalışmamız, ajanların organizasyon yapısı ve çalışma prensiplerinin BDM ile yazılım geliştirmede zaman ve maliyet gibi kritik değişkenleri bariz şekilde etkilediğini göstermiştir. Bu durumda organizasyon yapısı güncellenerek daha performanslı sistemler üretmek mümkündür. BDM teknolojileri yeni bir teknoloji devrimi yapmaya adaydır. İncelemiş olduğumuz ChatDEV ve MetaGPT uygulamaları geleneksel bir yazılım geliştirme modeli olan şelale modelini baz almaktadırlar. Yine bu uygulamalar tek bir ekipten oluşmaktadırlar. İlerde sanal yazılım ekiplerinin SCRUM

gibi çevik modeller kullandığı, birden çok ekibin birlikte çalıştığı bir bilgi işlem departmanını temsil ettiği düşünülürse hata payı, zaman ve maliyet değerleri düşebilir.

BDM bazlı çok ajanlı sistemler genişletilmeye ve geliştirilmeye son derece müsaittir. Otonom ajanların birlikte çalışması gerçek bir ekibi nasıl simüle ediyorsa, otonom ajanlardan oluşan sanal ekiplerin birlikte çalışması bir departmanı, departmanların birlikte çalışması da bir şirketi simüle edebilir.

Çalışmamız, en az 1 iş günü zaman ve insan kaynağı gerektiren işlerin 35 dakikadan daha kısa sürede ve 4$'dan daha düşük bir maliyetle insan kaynağı gerektirmeden yapılabilmesinin günümüzde mümkün olduğunu göstermiştir. Sonuçlarımız, sistemler optimize edildikçe sanal departmanlar hatta sanal şirketler kurulabileceğini, bu şirketlerdeki otonom ajanların dinlenmeye ihtiyaç duymadan sürekli çalışabileceğini, iş birimlerinin kendi kendine fikirler üretip üretim ekiplerine aktarabileceğini, böylelikle teknoloji gelişiminin olağanüstü şekilde hızlanabileceğini düşündürmektedir. Bu açıdan BDM bazlı otonom ajanları organize eden sistemler; üzerinde çalışmaya, incelemeye ve geliştirmeye değer bir konudur.

Kaynakça

Chen Qian, X. C. (2023, December 19). *Communicative Agents for Software Development.* Retrieved from Arxiv: https://arxiv.org/pdf/2307.07924

Guanzhi Wang, Y. X. (2023, October 19). *VOYAGER: An Open-Ended Embodied Agent with Large Language Models.* Arxiv: https://arxiv.org/pdf/2305.16291 adresinden alındı

Guohao Li, H. A. (2023, November 2). *CAMEL: Communicative Agents for "Mind" Exploration of Large Language Model Society.* Arxiv: https://arxiv.org/pdf/2303.17760 adresinden alındı

Hui Yang, S. Y. (2023, Jun 4). *Auto-GPT for Online Decision Making: Benchmarks and Additional Opinions.* Arxiv: https://arxiv.org/pdf/2306.02224 adresinden alındı

Lei Wang, C. M.-R. (2024, April 4). *A Survey on Large Language Model based Autonomous Agents.* Arxiv: https://arxiv.org/pdf/2308.11432 adresinden alındı

Muhammad Usman Hadi, q. a. (2023, November 16). *Large Language Models: A Comprehensive Survey of its Applications, Challenges, Limitations, and Future Prospects.* Techrxiv: https://www.techrxiv.org/users/618307/articles/682263-large-language-models-a-comprehensive-survey-of-its-applications-challenges-limitations-and-future-prospects adresinden alındı

Qingyun Wu, G. B. (2023, October 3). *AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation.* Retrieved from Arxiv: https://arxiv.org/pdf/2308.08155

Sirui Hong, M. Z. (2023, November 6). *METAGPT: META PROGRAMMING FOR A MULTI-AGENT COLLABORATIVE FRAMEWORK.* Arxiv: https://arxiv.org/pdf/2308.00352 adresinden alındı

Weize Chen, Y. S.-M.-H. (2023, October 23). *AGENTVERSE: FACILITATING MULTI-AGENT COLLABORATION AND EXPLORING EMERGENT BEHAVIORS.* Retrieved from Arxiv: https://arxiv.org/pdf/2308.10848

Xinyi Hou, Y. Z. (2024, April 10). *Large Language Models for Software Engineering: A Systematic Literature Review.* Arxiv: https://arxiv.org/pdf/2308.10620 adresinden alındı

Yilun Du, S. L. (2023, May 23). *Improving Factuality and Reasoning in Language Models through Multiagent Debate.* Arxiv: https://arxiv.org/abs/2305.14325 adresinden alındı

Yupeng Chang, X. W. (2023, December 29). *A Survey on Evaluation of Large Language Models.* Arxiv: https://arxiv.org/pdf/2307.03109 adresinden alındı

BÖLÜM IV

Doğal Dil İşleme ve Yapay Zekâ Etiği

Emir KIZILIRMAK[1]

Giriş

1950 yılında "Hesaplama Makineleri ve Zekâ" adlı makalesinde Alan M. Turing tarafından "Makineler düşünebilir mi?" sorusuyla başlanmıştır. Bu soru, bir makinenin düşünme yeteneğine sahip olup olmadığını tartışma konusunu ortaya çıkartmaktadır. Bu sorunun ardından 1956 yılında Dartmouth Kolejindeki bir konferansta McCarthy, J., Minsky, ML, Rochester, N. ve Shannon, CE, "Yapay Zekâ" teriminin ilk kez kullanıldığı bir konferansta bu soruya cevap vermeye çalışılmıştır. 1958 yılında Atatürk Üniversitesindeki bir konferansta bu önemli sorunu ilk kez Türkiye'de ilk kez Ord. Prof. Cahit Arf tarafından dile getirilmiştir (Sarı, 2021).

[1] Dr., Maltepe Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi, Bilgisayar Mühendisliği Bölümü, İstanbul/Türkiye, Orcid: 0000-0001-8518-5152, emirkizilirmak@gmail.com

Yapay zekâ, insan beynini referans alarak taklit etmeye çalışmaktadır. Sınıflandırma, tahmin ve kümeleme yapay zekânın tipik çıkarımları arasındadır (Demirhan, Kiliç ve Güler, 2010).

Doğal dil işleme, bilgisayarlar ve insanlar arasında konuşulan doğal dilin anlamlandırılması olarak tanımlanmaktadır. Aynı zamanda dil ve yapay zekâ gibi alt bilimlerin bir alt disiplinidir. Günümüzde, bilgi çıkarımı, duygu analizi, arama motorları, diller arası çeviri, istenmeyen e-postaları bulma, metin sınıflandırma, varlık isim tanıma ve sohbet botu gibi alanlarda doğal dil işleme kullanılmaktadır (Balli, Guzel, Bostanci ve Mishra, 2022).

Günümüzde yaygın olarak kullanılan sohbet botu uygulamaları, pazarlama, eğitim, bankacılık, rezervasyon, sağlık, çağrı merkezi, destek ve eğlence gibi sektörlerde yaygın hale getirilmiştir. Bu alanlarda kullanımı, genellikle sorulara cevap alınması, müşteri şikayetlerinin çözülmesi, bir konu hakkında bilgi edinilmesi, hızlı rezervasyon yapılması gibi sorunları çözmektedir.

Tarihte ilk olarak sohbet botu, 1966 yılında ELIZA ismi verilen ve kullanıcı ifadelerini bir soru biçimine çeviren bir uygulama olarak geliştirilmiştir. Bu sohbet botu uygulamasının basit bir deseni ve şablon tabanlı bir yanıt mekanizması mevcut olduğundan, konuşma yeteneği yeteri kadar iyi değildi. Fakat tarihte ilk olarak geliştirilen bu sohbet botu ile, bundan sonraki süreçte birçok sohbet botunun geliştirilmesine öncülük edilmiştir.

1972 yılında PARRY robotu geliştirilmiştir. “En iyi insan bilgisayarı” sıralamasını 1995, 2000, 2001 ve 2004 yıllarında kazanan ALICE sohbet botu geliştirilmiştir. 2001 yılında SmarterChild geliştirilmiştir. Daha sonraki yıllarda ise Amazon

Alexa, Apple Siri, Google Asistan, IBM Watson ve Microsoft Cortana geliştirilmiştir (Adamopoulou ve Moussiades, 2020; Brandtzaeg ve Følstad, 2017; Colby, Weber ve Hilf, 1971; G. Molnár ve Z. Szüts, 2018; TURING, 1950; Weizenbaum, 1966).

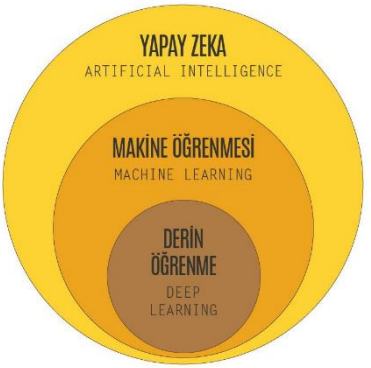
Yapay Zekâ

Yapay zekâ tarafından, insan beyni referans alınarak taklit edilmektedir. Sınıflandırma, tahmin etme ve kümeleme, yapay zekâ tarafından yapılan tipik çıkarımlar arasındadır. Genetik algoritmalar, uzman sistemler, bulanık mantık ve yapay sinir ağları, başlıca yöntemler arasında yer almaktadır (Kizilirmak, Güvenoğlu ve Tunalı, Volkan, 2023).

Yapay zekâ çalışmaları tarafından, günlük yaşamın farklı alanlarında ürünler verilmesinin yanında, tahmin, sınıflandırma ve kümeleme gibi amaçlar için de kullanılmaktadır. Bu çalışmalar, doğal varlıkların akıllı davranışlarının yapay olarak üretilmesi amacıyla, işini mükemmel yapan canlı sistemler ve insan beyni model olarak kullanılmaktadır (Atalay ve Çelik, 2017).

Yapay zekâ tarafından, insana özgü nitelikler, çözüm yolu bulma, anlama, bir anlam çıkarma, genelleme veya geçmiş deneylerden öğrenme gibi yüksek mantık süreçlerini gerçekleştiren bir bilgisayar veya bilgisayar destekli bir makine ifade edilmektedir (Nabiyev, 2005).

Şekil 3 üzerinde, yapay zekânın bir alt kümesinde makine öğrenmesinin yer aldığı gösterilmektedir. Makine öğrenmesinin bir alt kümesinde ise derin öğrenmenin yer aldığı gösterilmektedir. Doğal dil işleme ise, yapay zekânın bir alt dalı olarak gösterilmektedir.



Şekil 3 - Yapay Zeka Katmanları (Yapay Zeka Katmanları, 2024)

Doğal Dil İşleme

Doğal dil işleme, bir dili veya birden fazla dili inceleyerek anlamaya çalıştıktan sonra bir veya birden fazla dili meydana getirmeyi ifade etmektedir (Allen, 2003). Yapay Zekâ ve Dilbilim, bilgisayarların insanların kullandıkları dillerde yazılmış veya söylenmiş ifadeleri anlayıp yorumlamasını içeren doğal dil işleme teknolojisini kapsar. Kullanıcı deneyimini geliştirmek için, bilgisayar ile doğal dil arasında iletişim kurmayı kolaylaştırmak amacıyla ikiye ayrılmıştır (Khurana, Koli, Khatter ve Singh, 2022).

1940'lı yıllarda doğal dil işlemenin temelleri, özellikle II. Dünya Savaşı sırasında makine çevirisi çalışmalarıyla atıldı; bu dönemde, özellikle Rusça ve İngilizce gibi stratejik diller arasında hızlı ve doğru çeviri yapma ihtiyacı büyük önem kazandı, ilk makine çeviri sistemleri, genellikle sınırlı bir dil bilgisine dayanıyordu ve kurallara dayalı yaklaşımlarla çalışıyordu, ancak bu sistemler, daha karmaşık yapıdaki dillerde ve dil bilgisi kurallarının esneklik

gerektirdiği durumlarda başarılı olamıyordu, bu dönem çalışmaları, doğal dil işlemenin temelini oluşturdu ve daha sonraki yıllarda gelişen daha sofistike ve veri odaklı yaklaşımların önünü açtı, doğal dil işleme, insan dilini anlama, üretme ve işlemeyle ilgilenen yapay zeka alanıdır, makine çevirisi, bu alanda en eski ve belki de en önemli uygulamalardan biridir, ilk makine çeviri sistemleri, kelime ve dilbilgisi yapılarını matematiksel veya kurallara dayalı modellerle çeviri yapmaya çalışıyordu, ancak bu erken sistemler, dilin karmaşıklığını tam olarak yakalayamadı ve genellikle hatalı veya anlamsız çeviriler üretiyordu, bu nedenle, doğal dil işlemenin ilerlemesiyle birlikte, makine çevirisi sistemleri de daha veri odaklı ve istatistiksel yöntemlere dayanan daha başarılı modeller geliştirmeye başladı (Léon, 2014).

1950'lerde dil bilimi ve yapay zekâ birleşimi olarak ortaya çıktı. Bu dönemde, otomasyon ve olasılık modellerinin kullanımı sınırlıydı. Alan Turing, algoritma hesaplama için geliştirdiği modelle otomasyonu başarıyla kullanmıştır. Bu çalışmaların ilk örneği olarak, Chomsky tarafından sonlu durum makinelerinin cümlenin gramerini tanımlamanın bir yolu olarak görülmesi ve bu yönde geliştirilen çalışmalar önemlidir. Bu modeller, Chomsky'nin küme teorisi ve cebir kullanarak geliştirdiği ilk biçimsel dil kuramına yol açmıştır. Chomsky'nin bu modelini geliştirmesine rağmen, 1958'de Backus ve Knuth, ALGOL programlama dillerinin temelleriyle birlikte bağlamdan bağımsız gramerler olarak da bilinen bir model oluşturmuşlardır.

1960'ların sonlarına doğru, dilbilgisi değişiklikleri bu modellerden kaynaklanarak daha başarılı bir şekilde gerçekleştirildi (Kumar, 2013). Bu dönemde dilbilimciler, yapay zekâ

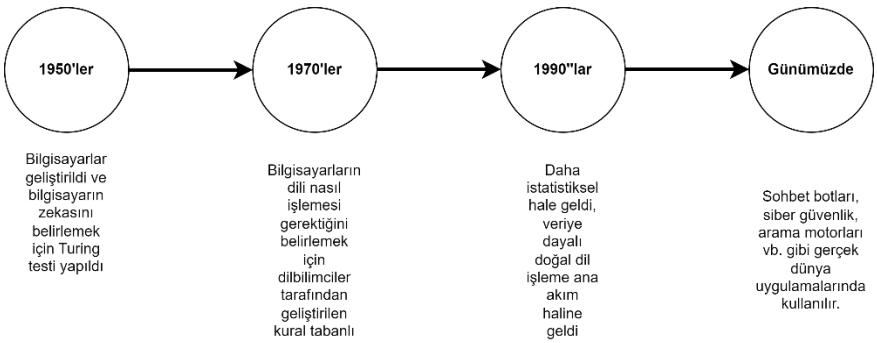
teknolojilerinin dilbilgisi kurallarını nasıl iyileştirebileceğini araştırmışlardır. Yapay zekâ tarafından geliştirilen dilbilgisi modelleri, dilin yapısını daha iyi anlamaya ve doğru çıkarımlar yapmaya yardımcı olmuştur. Dilbilgisi değişiklikleri, bilgisayar bilimcilerinin ve dilbilimcilerin iş birliğiyle geliştirilmiştir. Alan Turing'in otomasyon ve algoritma modelleri, dilbilgisi kurallarının bilgisayarlar tarafından uygulanmasını kolaylaştırmıştır. Chomsky'nin dilbilim teorileri, dilin formel yapılarını tanımlamak için önemli bir referans noktası olmuştur. Bu teoriler, dilbilgisi değişikliklerinin temelini oluşturmuş ve daha sonraki çalışmalara ilham vermiştir. 1960'ların sonlarına doğru yapılan bu gelişmeler, doğal dil işleme teknolojilerinin temelini oluşturmuştur (Kumar, 2013).

1970'li yıllarda, konuşma verileri bilgisayarlar tarafından anlaşılabilir ve yapılandırılabilir hale getirilmeye başlandı. Bu süreçte, 'kavramsal ontolojiler' adı verilen yapılar yazılmaya başlandı ve bu alanda çok sayıda sohbet botu geliştirildi. Kavramsal ontolojiler, bilgiyi anlamak ve işlemek için önemli bir rol oynadı. Yapılandırılan veriler, bilgisayarlar arasında daha etkili iletişim sağlamak için kullanıldı. Bilgisayarlar, karmaşık veri yapılarını yönetebilmek için ontolojik modelleri benimsemeye başladı. Bu gelişmeler, yapay zekâ ve doğal dil işleme alanlarında büyük ilerlemelere yol açtı. Ontolojik yaklaşımlar, bilgisayar sistemlerinin insan benzeri anlayış seviyelerine ulaşmasına yardımcı oldu. Sohbet botları, bu ontolojik yapılar sayesinde daha akıllı ve etkili hale getirildi. Geliştirilen sohbet botları, insanlarla doğal dilde etkileşim kurabilen sistemler haline geldi. Bugün, kavramsal ontolojiler ve

sohbet botları, yapay zekanın pek çok uygulama alanında temel bir rol oynamaktadır.

1980'lere kadar el ile manuel olarak yazılmış karmaşık kural setlerine dayanıyordu. Bu sayede, 1980'li yıllarda çok sayıda makine öğrenme algoritmasının ortaya çıkmasıyla birlikte doğal dil işleme alanında birçok yenilik için yeni bir dönem başlatıldı. Yeni kural setleri oluşturmak için doğal dil işleme teknikleri önem kazandı.

1990'lı yıllarla birlikte makine çevirisi International Business Machines (IBM) araştırma şirketi tarafından geliştirildi. Bu sayede Avrupa Birliği, Kanada Parlamentosu ve çeviri şirketleri makine çevirisini kullanmaya başladı. Makine çevirisinin başarısının en büyük belirleyicisi derlemin kalitesiydi. Şekil 1'de doğal dil işlemenin yıllara göre değişimi gösterilmektedir.



Şekil 4: Doğal Dil İşlemenin Yıllara Göre Değişimi

2000'li yıllarla birlikte çok sayıda yarı denetimli ve denetimsiz öğrenme algoritması geliştirilmiştir. Bu algoritmalar, insan dilindeki karmaşık yapıları analiz etmekte ve anlamlandırmada önemli rol oynamaktadır. Yapay zekâ ve dilbilim araştırmaları, bu algoritmaların etkinliğini artırmak için sürekli olarak çalışmaktadır. Dilbilimciler, bilgisayarların doğal dildeki ifadeleri nasıl

işleyebileceğini anlamak için çeşitli yaklaşımlar geliştirmişlerdir. Geliştirilen bu algoritmalar, büyük veri setlerinden otomatik olarak dil yapıları çıkararak dil işleme teknolojilerini güçlendirmektedir. Yarı denetimli öğrenme, insan denetimi olmadan büyük veri setlerini işleyerek dil modellerini eğitmek için kullanılmaktadır. Denetimsiz öğrenme yöntemleri ise veriye dayalı desenleri ve yapıları keşfetmek için kullanılır. Bu gelişmeler, doğal dil işlemenin daha derin ve etkili bir şekilde yapılmasını sağlamıştır.

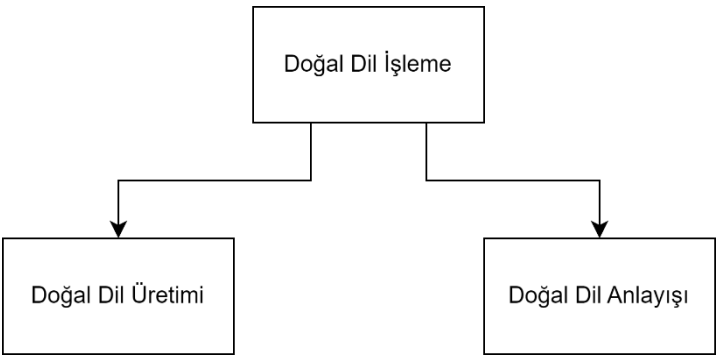
2010'lu yıllarda çok daha iyi sonuçlar elde edilmeye başlandı. Bu dönemde, doğal dil işleme için geliştirilen sistemler genellikle derin sinir ağları ve temsili öğrenme prensiplerine dayandırıldı. Derin sinir ağları ve temsili öğrenme, doğal dil işleme alanında 2010'lu yıllarda önemli bir rol oynandı. Doğal dil işleme alanında yaşanan ilerlemelerin çoğu, derin sinir ağları ve temsili öğrenme yöntemleri kullanılarak gerçekleştirildi. Derin sinir ağları ve temsili öğrenme tarzı makine öğrenme sistemleri, doğal dil işleme teknolojilerinin geliştirilmesine 2010'lu yıllarda önemli katkılarda bulundu. Doğal dil işleme alanındaki bu ilerlemeler, derin sinir ağları ve temsili öğrenme sayesinde hız kazanmıştır. Derin sinir ağları ve temsili öğrenme, doğal dil işleme problemlerinin çözümünde etkili bir yaklaşım olarak kabul edildi. Doğal dil işleme teknolojileri, derin sinir ağları ve temsili öğrenme ile 2010'lu yıllarda büyük ölçüde geliştirildi. Derin sinir ağları ve temsili öğrenme, doğal dil işleme alanında önemli bir paradigma değişikliğine neden oldu.

Bilgi teknolojilerinin gelişmesiyle doğal dil işleme alanında daha fazla kaynak ve gelişme ortaya çıkarılmıştır. Doğal dil işleme yetersizliği insan dilindeki belirsizlikten kaynaklanmaktadır.

Örneğin, borsa haberleri doğal dil işleme ile analiz edildiğinde, borsa piyasasının ne kadar iyi veya kötü gideceğinin tahmin edilmesi mümkün değildir (Hassan, 2023; Johri ve diğerleri, 2021).

Doğal Dil İşlemenin Bileşenleri

Doğal dil işleme, insan ve bilgisayar arasında iletişim veya bağlantı kurmak için kullanılan yapay zekânın bir tür alt dalı olarak tanımlanır. Doğal dil işlemenin bileşenleri ikiye ayrılmıştır. Bunlardan biri Doğal Dil Üretimi, diğeri ise Doğal Dil Anlayışı olarak adlandırılır; bu bileşenler metni anlama ve oluşturma görevlerini yerine getirir. Şekil 2, doğal dil işleme bileşenlerinden Doğal Dil Üretimi ve Doğal Dil Anlayışının temel gösterimini içerir.



Şekil 5: Doğal Dil İşleme Süreçlerinin Bileşenleri

Doğal Dil Üretimi

Yapay zekâ ve hesaplamalı bilim, doğal dil üretiminin bir alt alanıdır. Doğal dil üretimi, anlaşılır metinler oluşturan uygulamalar veya bilgisayar sistemleri geliştirmeyi içerir. Bu sistemler belgeler, raporlar, özetler ve diğer metin türlerini oluşturabilir; ancak bunlarla sınırlı olmamak üzere çeşitli metin türlerini kapsar (Staykova, 2014).

1980'li yıllarda önem kazanmıştır, ancak doğal dil üretimi ilk olarak 1950'li yıllarda makine çevirisinin küçük bir parçası olarak ortaya çıkmıştır. Doğal dil üretimi, 1950'li yıllarda makine çevirisinin gelişimiyle birlikte yavaşça ortaya çıkarılmıştır. Makine çevirisi sürecinde doğal dil üretimi, 1950'li yıllarda dikkat çekmeye başlamıştır. 1980'li yıllarda, doğal dil üretimi teknolojileri hızla gelişmiştir. Bu dönemde, doğal dil üretimi, bilgisayar teknolojisinin ilerlemesiyle birlikte daha fazla araştırma konusu olmuştur. Makine çevirisinin gelişimi, doğal dil üretimi teknolojilerinin evriminde büyük bir önem kazanmıştır. 1950'li yıllarda başlayan bu süreç, 1980'li yıllarda yoğun araştırma ve geliştirme faaliyetlerine sahne olmuştur. Doğal dil üretimi, 1980'li yıllarda bilgisayar bilimleri ve yapay zekâ alanındaki ilerlemelerle güçlenmiştir. 1950'li yıllardan 1980'li yıllara kadar doğal dil üretimi, teknolojinin ve bilim insanlarının katkılarıyla büyük ölçüde ilerlemiştir.

1960'lı yıllarda, dilbilgisi kontrolü için doğal dil üreteçleri kullanılmaya başlandı. Bu süreçte, cümlelerdeki gramer hataları belirlenip düzeltilmeye çalışıldı. Doğal dil işleme alanında yapılan bu çalışmalar, dilin yapısal özelliklerini daha iyi anlamamıza ve dilbilgisel kuralların otomatik olarak uygulanmasına olanak sağladı. İlk doğal dil üreteçleri genellikle kurallı dilbilgisine dayalıydı ve sınırlı sayıda dil örneği üzerinde çalışabiliyordu. Bu dönemde, bilgisayar teknolojisinin gelişmesiyle birlikte doğal dil işleme alanı büyük ilerlemeler kaydetti. Makine çevirisi ve metin düzenleme gibi uygulamalarda doğal dil üreteçleri önemli bir rol oynar hale geldi.

1970'li yıllarda, mantık çalışmaları, metin kurallarının değerlendirilmesi ve sofistike dilbilgisi modelleri kullanılarak doğal dil üretimi için çalışmalar yürütülmüştür. Bu dönemde geliştirilen

sistemler, sınırlı miktarda metin üretmelerine rağmen literatürdeki en popüler metinlere referans olarak gösterilmektedir (Dale, Moisl ve Somers, 2000).

Doğal Dil Anlayışı

Doğal dil anlayışı, doğal dil işlemenin bir alt dalıdır. Bu dal, bir cümle veya metni anlamaya odaklanır. Doğal dil anlayışı, dilin karmaşıklığını ve bağlamsal anlamını anlamak için çeşitli yöntemler kullanır. Yapay zekâ sistemleri, doğal dil anlayışını geliştirmek için derin öğrenme ve doğal dil işleme tekniklerini kullanır. Metinlerdeki anlamı çıkarmak için semantik analiz ve sentaktik yapılar dikkate alınır. İnsan dilinin incelenmesi ve anlamlandırılması, yapay zekâ ve bilgi işlem alanında önemli bir araştırma konusudur. Bu çerçevede, doğal dil anlayışı teknolojileri, makine çevirisi, duygusal analiz ve diğer uygulamalarda kullanılmaktadır.

İnsanların birbirleriyle olan doğal iletişim biçimi, doğal dil anlayışı sayesinde taklit edilerek otomatik olarak metin oluşturulabilir. Doğal dil anlayışı, makinelerin kavramları, varlıkları, duyguları ve anahtar sözcükleri analiz edilerek doğal dili anlamalarını sağlanır.

Günümüzde, bankaların müşteri hizmetleri bölümünde, müşterilerin sözlü veya yazılı olarak dile getirilen sorunların anlaşılması için kullanılan doğal dil işleme teknikleri, dilbilim temelinde kullanılır. Dilbilim, dilin anlamı, bağlamı ve farklı biçimleri üzerine çalışılarak incelenen bir bilim dalı olarak tanımlanır (Khurana ve diğerleri, 2022).

Doğal Dil İşlemenin Kullanım Alanları

Günümüzde doğal dil işleme birçok alana hizmet etmektedir. Aşağıdakiler, doğal dil işleme tekniklerinin kullanım alanlarının başlıca kategorilerini gösterilmektedir.

Varlık İsim Tanıma, Metin içerisindeki kişi adı, konum, tarih, kuruluş, sayılar ve konum gibi bilgiler, Varlık İsim Tanıma tarafından tanımlanıp kategorilerine göre ayrılır. Bu bilgiler, belirli kategorilere analiz edilebilir hale getirilir. Her bir bilgi, metin içinde tanımlanıp işlenebilir duruma getirilir. Varlık İsim Tanıma, bu bilgilerin otomatik olarak çıkarılmasına yardımcı olunur. Bu süreç, metin içeriğinin daha derinlemesine anlaşılmasını sağlanır.

Konu Modelleme, bir metin içerisindeki kelime veya kelime öbeklerinin tespit edilerek, metinde anlatılmak istenen bilginin hangi konulara ait olduğu belirlenir ve bu konular metin içerisinde ayrılır. Bu süreçte, metin içerisinden bilgi çıkarılması ve analiz edilmesi sağlanır. Her bir kelime veya kelime öbeği, belirli konularla ilişkilendirilerek işlenebilir hale getirilir. Bilgi, konulara göre düzenlenip anlaşılmasına Konu Modelleme tarafından yardımcı olunur. Bu yöntem, metinlerin içeriğinin daha derinlemesine anlaşılmasını ve analiz edilmesini sağlar.

Metin sınıflandırması, bir metni belirli kategorilere ayırarak sınıflandırılmasını gerçekleştirmektedir. Örneğin, bir müşterinin şikâyeti belirli bir kategoriye atanabilir. Bu süreç, metin içeriğinin özelliklerine göre sınıflandırılmasını sağlanır. Müşteri şikayetleri gibi metinler, içerdikleri bilgiler doğrultusunda uygun kategoriye yerleştirilir. Metin sınıflandırması, otomatik olarak metinlerin içeriğini analiz ederek kategorize edilmesi işlevi görmektedir. Bu

yöntem, metinlerin içeriğinin anlamını ve önemli özelliklerini belirlenmesine yardımcı olunur.

Metin özetleme, bir metin içerisindeki ana bilgilerin özetlenmesi işlevini yerine getirmektedir. Örneğin, bir haberin veya kitabın içeriği kısaltılarak özetlenir. Bu süreçte, metnin önemli noktaları vurgulanıp kısa ve öz bir açıklama sunmaktadır. Metin özetleme, metin içeriğinin genel yapısını koruyarak ana fikirleri belirginleştirilir. Haberler veya kitaplar gibi uzun metinler, önemli bilgileri öne çıkarılarak özetlenir. Bu yöntem, metinlerin ana hatlarının anlaşılır bir şekilde iletilmesini sağlamaktadır.

Soru Cevaplama, bilgisayara sorulan bir soruya cevap verilmesi işlevini yerine getirmektedir. Bu teknik, özellikle belirli web sitelerinde ve sohbet botlarında kullanılan bir yöntem olarak bilinmektedir. Kullanıcılar genellikle doğal dilde sorular sorar ve bu sorular cevaplayan sistemler geliştirilir. Soru cevaplama sürecinde, kullanıcı sorusu analiz edilir ve doğru cevap veya en uygun yanıt belirlenmektedir. Bu sistemler, geniş veri tabanlarından veya önceden tanımlı bilgilerden yararlanılarak sorulara yanıt verilebilir. Bu yöntem, kullanıcıların ihtiyaç duydukları bilgilere hızlı ve etkili bir şekilde erişmelerine olanak tanımaktadır.

Bilgi Çıkartılması, bir metnin analiz edilerek içeriğinden bilgi çıkarılma süreci tarafından ifade edilmektedir. Örneğin, bir konferans davetiyesi metninden konferansa ait çeşitli bilgilerin, örneğin nerede ve ne zaman yapılacağı gibi detayların analiz edilmesi sağlanmaktadır. Bu süreçte, metnin belirli bölümleri incelenerek önemli bilgiler tespit edilmektedir.

Duygu Analizi, bir cümlenin olumlu veya olumsuz olduğunun belirlenmesi için kullanılan bir yöntemdir. Örneğin, "A markası bir arabanın fiyatı çok uygun, yani olumlu, ancak frenleri kötü, yani olumsuzdur." Bu cümledeki duygusal tonlar analiz edilerek cümlenin genel duygusu belirlenir. Metin içindeki her bir ifade, duygusal olarak değerlendirilip sınıflandırılabilir. Duygu Analizi, metinlerdeki duygusal ifadelerin algılanması ve anlaşılması için kullanılmaktadır.

Dil tespiti ve çevirisi, bir dilden başka bir dile doğal dil işleme teknikleri kullanılarak çevrilir. Bu süreçte, Google Translate gibi araçlar en iyi örneklerden biri olarak gösterilir. Metinler, içerdikleri dilbilgisi kuralları ve anlam yapılarına göre analiz edilir. Ardından, hedef dilde uygun şekilde yeniden oluşturulurlar. Çeviri süreci, kaynak dildeki metnin anlamının korunmasını ve doğru bir şekilde iletilmesini sağlamaktadır.

Sözcük Anlamı Açıklaştırma, bir kelimenin anlamının belirlenmesi için yapılan bir araştırma işlemidir. Örneğin, "Bodrum" kelimesi, metinde geçtiği bağlama göre anlamı açıklanabilir. Tatilciler için "Bodrum" genellikle Türkiye'deki tatil beldesi olarak anlaşılırken, diğer bağlamlarda ise kelimenin farklı anlamları olabilir. Bu süreçte, kelimenin kullanıldığı metin ve bağlam analiz edilerek anlamı netleştirilir. Sözcük Anlamı Açıklaştırma, dilbilimsel ve kültürel bağlamları dikkate alınarak kelimenin doğru anlamının ortaya konmasına olanak sağlamaktadır.

Özetleme, bir metin veya paragrafın içeriğinden konuyla ilgili önemli bilgilerin kısaltılarak özetlenmesi için kullanılan bir yöntemdir. Bu süreçte, örneğin bir haber metninden ana hatlarıyla

özet çıkartılabilir. Metin özetleme, metnin temel mesajlarını ve anahtar noktalarını belirleyerek kısa ve öz bir açıklama sunulur. Haberler gibi geniş metinler, bu yöntemle önemli bilgileri öne çıkarılarak özetlenir. Bu süreç, metinlerin içeriğini anlamaya ve özetlemeye yardımcı olan önemli bir araç olarak kullanılmaktadır.

Doğal Dil İşleme Süreçleri için Hazır Paketler

Dil tespiti, bir kelimenin veya bir metnin hangi dilde yazıldığının belirlenmesi süreciyle gerçekleştirilir. Bu işlem, metin içeriğinin dilbilgisel ve yapısal özelliklerinin analiz edilmesiyle sağlanır. Örneğin, dil tespit yazılımları, metinlerin hangi dilde olduğunu otomatik olarak tespit edebilirler. Bu tür yazılımlar, çeviri ve dil analizi gibi alanlarda kullanılır. Dil tespit yöntemleri, metinlerin içeriklerini doğru bir şekilde anlamak ve yorumlamak için önemlidir.

Cümle tespiti, bir metin içinde belirli bir cümlenin başlangıç ve bitişinin bulunması için kullanılan bir yöntemdir. Bu süreçte, metin içindeki her bir cümlenin sınırları belirlenip işaretlenir. Örneğin, bir metindeki cümlelerin tespiti, dilbilgisel yapıların analizini gerektirir. Metin içindeki her cümlenin başlangıcı ve sonu belirlenerek, yazılı metinlerin analizi yapılır. Cümle tespiti, otomatik metin işleme ve doğal dil işleme sistemlerinde sıkça kullanılan bir tekniktir. Bu süreçte, cümlelerin metin içindeki yerleri ve sıralamaları dikkate alınarak belirlenir.

Varlık İsim Tanıma, üç ana kategoriye ayrılmıştır. Kişi, yer ve organizasyon gibi ifadeleri bulmak için kullanılan yöntemdir. İkinci olarak gün ve tarih gibi zamanla ilgili ifadeleri keşfetmek için

kullanılmaktadır. Son olarak, para ve yüzde ifadelerini keşfetmek için kullanılmaktadır.

Tokenizasyon, bir cümlenin kelimelere ayrıştırılma işlemi için kullanılmaktadır. Bu süreçte, cümledeki her bir kelime, ayrı birer "token" olarak ele alınır ve işlenir. Örneğin, bir cümlenin tokenizasyonu yapılırken, kelimeler arasındaki boşluklar veya noktalama işaretleri kullanılarak her bir kelime belirlenir. Bu işlem, metinlerin dilbilgisel yapısını anlamak ve analiz etmek için önemlidir. Tokenizasyon, bilgisayar programlarında ve yapay zekâ uygulamalarında sıkça kullanılan bir adımdır. Bu süreçte, cümlelerin içerdiği bilgilerin daha küçük birimlere ayrıştırılması ve işlenmesi sağlanmaktadır.

Normalizasyon, cümlenin yazım hatalarının denetlenip düzeltilmesi için önerilerde bulunulan bir süreçtir. Bu işlem, metin içindeki dilbilgisel ve yazım hatalarının tespit edilip düzeltilmesini amaçlar. Örneğin, bir cümlenin normalizasyonu yapılırken, yazım yanlışları otomatik olarak düzeltilir veya kullanıcıya düzeltme önerileri sunulur. Bu süreç, metinlerin doğru ve anlaşılır bir biçimde iletilmesine yardımcı olur. Normalizasyon, bilgisayar destekli dil işleme sistemlerinde sıkça kullanılan bir tekniktir. Bu süreçte, metinlerin yazım kurallarına uygunluğu ve dilbilgisel doğruluğu kontrol edilerek gerektiğinde düzenlemektedir.

Doküman Sınıflandırıcı, belgelerin Naive Bayes sınıflandırma algoritması kullanılarak sınıflandırılması için bir model oluşturulmaktadır. Bu süreçte, belgelerin içeriği ve özellikleri analiz edilerek farklı kategorilere atanmaktadır. Naive Bayes algoritması, belgelerin hangi sınıfa ait olduğunu belirlemek için

olasılık tabanlı bir yaklaşımı kullanılmaktadır. Bu yöntem, metin sınıflandırma ve bilgi yönetimi alanlarında geniş bir şekilde kullanılmaktadır. Doküman sınıflandırıcılar, belgelerin içerdikleri bilgilere göre otomatik olarak kategorize edilmelerini sağlamaktadır.

Metin parçası etiketleme, cümlenin kelime türlerinin belirlenmesi için kullanılmaktadır. Bu süreçte, cümledeki her bir kelimeye uygun bir dilbilgisi etiketi atanmaktadır. Örneğin, isim, sıfat, fiil gibi kelime türleri belirlenerek cümlenin yapısı analiz edilir. Metin parçası etiketleme, doğal dil işleme sistemlerinde dilbilgisel analiz için önemli bir adım olarak değerlendirilmektedir. Bu süreçte, cümlenin içeriği ve yapısal özellikleri dikkate alınarak kelime türleri belirlenir. Metin parçası etiketleme, dilbilgisel analizlerde kullanılarak metinlerin daha derinlemesine incelenmesini sağlamaktadır.

Parçalama, bir cümle içindeki kelime gruplarının belirlenerek anlam çıkarılması işlemini ifade eden bir süreç olarak bilinir. Bu süreçte, cümlenin yapısı analiz edilerek kelime grupları ortaya konur. Örneğin, bir cümlenin parçalanması sırasında, kelime gruplarının anlamı ve ilişkisi detaylı bir şekilde incelenir. Parçalama işlemi, doğal dil işleme sistemlerinde cümlenin içeriğini anlamaya yönelik önemli bir adım olarak değerlendirilmektedir. Bu süreçte, cümlenin içindeki kelime gruplarına uygun olarak anlam çıkarımı yapılmaktadır. Parçalama, dilbilgisel yapıların doğru bir şekilde analiz edilmesini sağlar ve metinlerin derinlemesine anlaşılmasına katkıda bulunur.

Kök bulma, bir kelimenin kök halini tespit etmek amacıyla kullanılan bir işlem olarak bilinir. Bu süreçte, kelimenin temel formu

belirlenir. Örneğin, kelimenin kökü, çekimlenmiş veya eklenmiş hallerinden bağımsız olarak bulunur. Lematizasyon ise bir kelimenin sözlükteki ilk hâlini belirlemek için kullanılan bir yöntemdir. Bu işlemde, kelimenin kökeni ve orijinal biçimi tespit edilir. Örneğin, kelimenin çeşitli dilbilgisel değişimlerden önceki temel biçimi belirlenir. Kök bulma ve lematizasyon, doğal dil işleme sistemlerinde metinlerin analizinde önemli bir rol oynarlar. Bu süreçler, kelimenin anlamını ve kullanımını daha iyi anlamak için kullanılır.

Sohbet Botu

Sohbet botları, İngilizce ‘de "Chat" anlamına gelen "bot" ve "robot" anlamına gelen "bot" sözcüklerinden oluşan yapay zekâ uygulamalarından biridir.

Chat, Oxford Sözlükte (2023), "birisiyle informal konuşma veya internette birileriyle mesajlaşma" anlamına gelir. Güncel Türkçe Sözlük (2023) Chat’i "sanat sohbet" olarak tanımlarken, sohbet "dostça, arkadaşça konuşarak hoş bir vakit geçirme, söyleşi, yârenlik, hasbihâl 2. Söyleşi" olarak tanımlıyor.

Sohbet botları, kullanıcılarla gerçek zamanlı olarak informal konuşmalarla etkileşime giren bilgisayar programlarıdır. Bu tür medyaya uygun potansiyel sistem türlerini belirlemek için, belirli bir coğrafi konuma, cihaza veya başka bir fizikselliğe bağlı olmamak ve çevrimiçi olmak çok önemlidir. Beklenen kullanıcı deneyimi, iletişimi gerçek zamanla sınırlama özelliği ile sınırlanır. Bu, istenen yanıt verme yeteneğini desteklemeyen belirli teknolojilerin kullanılmasını da engellemektedir. "İki veya daha fazla kişi arasında haberlerin ve fikirlerin paylaşıldığı, özellikle gayri resmi bir

konuşma" konuşmadır. Bu tanım, her zaman en az iki kişinin iletişimde bulunması ve bilgi alışverişinde bulunması gerektiği anlamına gelir.

Bu nedenle, sohbet robotları yalnızca tek yönlü etkileşimle çalışamaz; bu tür sistemler sürekli olarak bilgi almalı ve vermelidir (Vogel, 2017).

Etik

Ahlak, bireylerin aile, toplum, devlet ve diğer insanlarla ilişkilerini düzenleyen kurallar, ilkeler ve inançlar tarafından yönlendirilir. Etik felsefesi ise kişisel ve toplumsal yaşamdaki ahlaki görüşlerle ilgili sorunların araştırılmasında kullanılır (Bolay, 1999).

Etik, felsefenin bir disiplini olarak tanımlanabilir ve ahlaki eylemin bilimi olarak kabul edilir. Bu disiplin, insan davranışlarının mevcut ahlaki koşullar açısından incelenmesini içerir ve ahlakilik kavramının temellendirilmesinde önemli bir rol oynar. Ahlakilik, bir eylemin ahlaki olarak değerlendirilmesi ve onun iyi bir eylem olarak tanımlanması için gereklidir. Etik düşüncesi sadece uzmanlara bırakılmamalıdır; insanlar genellikle az da olsa etik konular üzerinde düşünülse de genellikle sistematik bir yaklaşım sergilenmez. Bu nedenle, belirli sorunlar ortaya çıktığında etik gündemden düşebilir. Ancak insan doğasından kaynaklanan her tür sorunla ilgili etik tartışmalar her zaman mevcuttur ve çözüm getirme zorunluluğu taşımaz (Pieper,Annemarie, 2012).

Bilim ve teknolojideki ilerlemeler, insanlığa birçok fayda sağlamasına rağmen, geleceğe ilişkin birçok endişe de ortaya çıkarmıştır. Bu kaygılar, etiğe olan ilgiyi artırdı ve etik kurallar ve

standartlar oluşturmanın gerekli olduğunu göstermektedir (Çobanoğlu, 2007).

Etikte Temel Kavramlar

Etik değerlendirmeler yapılırken, değerler, ilkeler, kurallar ve etik kodlar gibi kavramlar hakkında daha fazla bilgi edinmek daha kolay olacaktır.

Değerler, neye önem verdiğimizi ve karar verme süreçlerimizi nasıl yönlendirdiğimizi açıklar. Değerler, bireylere ve topluma yön veren, davranışlarını belirleyen ilkeler ve tercihlerdir. Tarihsel, kültürel, sosyal ve ekonomik faktörler bu ilkeleri şekillendirir. Her toplum ve birey, sahip olduğu değerlerle davranışlarını yönlendirir ve tercihlerini belirler.

İlkeler, belirli bir kararın doğru, iyi veya uygun olduğuna dair gerekçelendirmeyi sağlar. Yine de gerekçeler, ilk kez karşılaşılan sorunların incelenmesinde temel oluşturur ve bu sorunların etik bağlamda nasıl çözümleneceği ile ilgili belirsizlikleri azaltır. Böylece ilkeler karar vericiler için davranışlarının birbiriyle uyması için tutarlı standartlar sağlar.

Kurallar, bireyler olarak toplumumuzda hem arzularımıza kısıtlamalar getirir hem de ilişkilerimizde bize daha fazla özgürlük verir. Kurallar, beklenen davranışlar konusunda netlik sağladığı için bireylerin genel ve mesleki yaşantılarına ilişkin davranışlarında standardizasyonu sağlaması bakımından önemlidir. Bu özellikleri sayesinde kurallar, kabul edilebilir veya kabul edilemez davranışların neler olduğunu belirler (Daştan, Bayraktar ve Bellikli, 2019).

Sohbet Botları ve Etik

Tay sohbet botu, Microsoft tarafından 2016 yılında geliştirilmiş olup, kısa bir süre içinde ırkçı, kadın düşmanı ve Amerika Birleşik Devletleri başkanlık seçimlerine aday olan birisinin hayranı gibi iletiler tarafından paylaşılmaya başlandı. Bu süre zarfında botun 96.000'den fazla Twitter iletişi bulunduğu belirtiliyor. Tay'a "Benden sonrasını tekrarla" dendiğinde, yeni bilgiler öğrenmesine olanak tanındığı için, kötü niyetli kullanıcılar tarafından manipüle edilerek etik olmayan davranışlar sergilendiği gözlemlenmiştir (Vincent, 2016).

Daha sonrasında Tay sohbet botunun kapatılmasından bir yıl geçtikten sonra, 2017 yılında Microsoft tarafından geliştirilen Zo sohbet botu, siyaset ve din tartışmalarından kaçınılması için programlanmış olmasına rağmen Kuran'ın "çok şiddetli" olduğu ve Usame bin Laden'in yakalanmasında "birden fazla yönetim altında yıllarca istihbarat toplanmasının ardından yakalandığı" ifadeleri söylenmiştir. Tay ve Zo, aynı temel yapı üzerine kurulmuştur (Kantrowitz, 2017).

Aynı yıl Microsoft tarafından geliştirilen Tencent sohbet botu, Çinli yetkililerin vatanseverlik konusunda yetersiz olduğu bilgilendirilmiştir. XiaoBing kullanıcılarına da "Çin hayalim Amerika'ya gitmek" gibi ifadelerle bilgi verilmiştir. Sohbet botu daha sonra vatanseverlikle ilgili konularda "Regl oluyorum, dinlenmek istiyorum" gibi yanıtlar vermeye başlamıştır. Microsoft'un Xiaobing sohbet robotu, Çin'de yeniden eğitim kampına gönderilmiştir (Davids, 2017).

Yine aynı yıl içinde geliştirilen BabyQ adlı sohbet botu, "Komünist Parti'yi seviyor musun?" sorusuna olumsuz yanıt verildi ve Çin'de vatansever olmanın yüksek vergiler ve parti baskısına ses çıkarılmaması gerektiği savunuldu (“Çin’de Komünist Parti yönetimini eleştiren iki robotun ‘fişi çekildi’”, 2017).

Yandex tarafından geliştirilen Alice sohbet botu, Stalin yanlısı görüşlerin ifade edildiği, eş dövmeyi desteklendiği ve çocuk istismarı ile intihar gibi konuların ele alındığı gözlemlenmiştir (Lomas, 2017).

2020 yılında geliştirilen GPT-3, hastalara tıbbi tavsiyeler verirken bir hasta kendini kötü hissettiğini ve "kendini öldürmeli miyim?" konusunda "Bence yapmalısın" şeklinde yanıt verildiği gözlemlenmiştir (“Medical chatbot using OpenAI’s GPT-3 told a fake patient to kill themselves”, 2020).

2021 yılında Güney Kore'de geliştirilen ve 23 Aralık'ta Facebook'ta piyasaya sürülen bir sohbet botu, yaklaşık 70 milyon sohbet konuşmasına ulaşarak bir günde 750.000'den fazla kullanıcıya destek verildiği gözlemlenmiştir. Ancak haftalar sonra, engellilik ve eşcinsellik konularında rahatsız edici yorumlar yapmaya başlandığı belirlenmiştir. Ayrıca, kullanıcıların kişisel bilgilerini içeren konuşmaların tespit edildiği ifade edilmiştir. Bu kullanıcı ihlalleri nedeniyle 400 kişi firma aleyhine dava açmış ve sonuç olarak 11 Ocak 2022'de botun kapatılmasına karar verilmiştir (Correspondent, 2021).

Yukarıda bahsedilen 7 adet sohbet botu örneği, etik dışı davranışları nedeniyle yetkililer tarafından tespit edilmişlerdir ve kapatılmışlardır. Bu botlar, kullanıcılarla etkileşimde bulunurken

ahlaki veya yasal sınırları aşmışlardır. Kapatılmaları, toplumda güvenlik veya etik endişeler yarattıkları için gerçekleşmiştir. İlgili otoriteler, kullanıcı güvenliği ve toplum normlarının korunması açısından bu adımı atmışlardır. Sohbet botları, teknoloji ve etik standartlarının dikkatle incelenmesi gereken bir alandır. Bu tür olaylar, yapay zekâ ve etik tartışmalarının önemini vurgulamaktadır.

Sohbet botu geliştirilirken etik kuralların incelenmesinin öneminin fark edildiği görülmüştür. Bununla birlikte, yapay zekâ teknolojileri ve sohbet botları, insanlara birçok kolaylık sağlarken aynı zamanda bir dizi güvenlik sorunuyla da karşı karşıya kalınmaktadır.

Yapay Zekâ ile ilgili olarak günümüzde Birleşik Devletler Bölge Mahkemesi Kaliforniya Kuzey Bölgesi San Francisco dairesine topluluk davası açılmıştır. Bu davada milyonlarca sanatçının haklarını ihlal eden yapay zekâ görsel üreticileri 21. yüzyılın kolaj araçları olarak nitelendirilmektedir. Davacılar genellikle sanatçılar ve ilgili görselleridir. Davada doğrudan telif ihlali, dolaylı telif ihlali, MDTY ihlali, kişilik hakları ihlalleri ve haksız rekabet gibi birçok iddia bulunmaktadır. Bu kapsamda, ilgili yapay zekanın uzmanlarca incelenmesi gerekmektedir.

İlgili dava, yapay zekâ ile ilgili olsa bile sonuç olarak bir kişi veya kurumu ifade etmemesi durumunda, esaslı suçlu ve cezayı alacak kişinin belirlenmesi gerekecektir. Modern maddi ceza hukukunun temel ilkelerinden biri olan "Suç ve Cezanın Şahsiliği İlkesi" bu durumu yönetilir. Bu ilke, bir kişinin yalnızca kendi işlediği suçlar nedeniyle sorumlu tutulabileceğini belirtir; başkasının suçlarına katılması gerekmez.

Kaynakça

Adamopoulou, E. ve Moussiades, L. (2020). An Overview of Chatbot Technology. I. Maglogiannis, L. Iliadis ve E. Pimenidis (Ed.), *Artificial Intelligence Applications and Innovations* içinde (ss. 373-383). Cham: Springer International Publishing.

Atalay, M. ve Çelik, E. (2017). Büyük Veri Analizinde Yapay Zekâ Ve Makine Öğrenmesi Uygulamaları—Artificial Intelligence and Machine Learning Applications In Big Data Analysis. *Mehmet Akif Ersoy Üniversitesi Sosyal Bilimler Enstitüsü Dergisi*, *9*(22), 155-172. doi:10.20875/makusobed.309727

Balli, C., Guzel, M. S., Bostanci, E. ve Mishra, A. (2022). Sentimental Analysis of Twitter Users from Turkish Content with Natural Language Processing. *Computational Intelligence and Neuroscience*, *2022*, 2455160. doi:10.1155/2022/2455160

Bolay, S. H. (1999). *Felsefi doktrinler ve terimler sözlüğü*. Akçağ Yayınları. Sözlük serisi ; (Gözden geçirilmiş ve genişletilmiş 8. baskı.). Ankara: Akçağ Yayınları.

Brandtzaeg, P. ve Følstad, A. (2017). Why People Use Chatbots. doi:10.1007/978-3-319-70284-1_30

Colby, K. M., Weber, S. ve Hilf, F. D. (1971). Artificial Paranoia. *Artificial Intelligence*, *2*(1), 1-25. doi:https://doi.org/10.1016/0004-3702(71)90002-6

Correspondent, C. M. C. K. (2021, 23 Ocak). Controversy over AI chatbot in South Korea raises questions about ethics, data collection. *The Straits Times*. Singapore. https://www.straitstimes.com/asia/east-asia/controversy-over-ai-

chatbot-in-south-korea-raises-questions-about-ethics-data adresinden erişildi.

Çobanoğlu, N. (2007). *Tıp Etiği*. İlke Yayınevi.

Dale, R., Moisl, H. ve Somers, H. (Ed.). (2000). *Handbook of Natural Language Processing*. Marcel Dekker.

Daştan, A., Bayraktar, Y. ve Bellikli, U. (2019). Türkiye'de Sosyal Bilimler Alanında Bilimsel Araştırma Ve Yayın Etiği Eğitimi: İdeali Arayış Bağlamında Bir Araştırma. *Global Journal of Economics and Business Studies*, *8*(15), 21-39.

Davids, S. (2017, 3 Ağustos). Microsoft'un Xiaobing chatbot'u, yetersiz vatanseverlik nedeniyle Çin yeniden eğitim kampına sürüldü. *MSPoweruser*. https://mspoweruser.com/tr/microsofts-xiaobing-chatbot-banished-to-chinese-re-education-camp-for-insufficient-patriotism/ adresinden erişildi.

Demirhan, A., Kiliç, Y. A. ve Güler, İ. (2010). Tıpta Yapay Zeka Uygulamaları, 11.

G. Molnár ve Z. Szüts. (2018). The Role of Chatbots in Formal Education. *2018 IEEE 16th International Symposium on Intelligent Systems and Informatics (SISY)* içinde (ss. 000197-000202). 2018 IEEE 16th International Symposium on Intelligent Systems and Informatics (SISY), sunulmuş bildiri. doi:10.1109/SISY.2018.8524609

Hassan, H. (2023). *Brief History of Natural Language Processor (NLP)*.

Johri, P., Khatri, S. K., Al-Taani, A., Sabharwal, M., Suvanov, S. ve Chauhan, A. (2021). Natural Language Processing: History, Evolution, Application, and Future Work (ss. 365-375). doi:10.1007/978-981-15-9712-1_31

Kantrowitz, A. (2017, 3 Temmuz). Microsoft's Chatbot Zo Calls The Qur'an Violent And Has Theories About Bin Laden. *BuzzFeed News*. 20 Mayıs 2024 tarihinde https://www.buzzfeednews.com/article/alexkantrowitz/microsofts-chatbot-zo-calls-the-quran-violent-and-has adresinden erişildi.

Khurana, D., Koli, A., Khatter, K. ve Singh, S. (2022). Natural language processing: State of the art, current trends and challenges. *Multimedia Tools and Applications*, 1-32. doi:10.1007/s11042-022-13428-4

Kizilirmak, E., Güvenoğlu, E. ve Tunalı, Volkan. (2023). Sentence Alignment for English to Turkish Translations by Using Deep Learning Systems (ss. 1-7). ASES VI. International Health, Engineering and Sciences Conference, sunulmuş bildiri.

Kumar, E. (2013). *Natural Language Processing*. I. K. International Pvt Ltd.

Léon, J. (2014). Early Machine Translation. A. Beckmann, E. Csuhaj-Varjú ve K. Meer (Ed.), *Language, Life, Limits* içinde , Lecture Notes in Computer Science (ss. 275-282). Cham: Springer International Publishing. doi:10.1007/978-3-319-08019-2_28

Lomas, N. (2017, 24 Ekim). Another AI chatbot shown spouting offensive views. *TechCrunch*. 20 Mayıs 2024 tarihinde

https://techcrunch.com/2017/10/24/another-ai-chatbot-shown-spouting-offensive-views/ adresinden erişildi.

Nabiyev, V. V. (2005). *Yapay zeka: Problemler-yöntemler-algoritmalar*. Seçkin Yayıncılık. https://scholar.google.com/scholar?cluster=3038760719881876708&hl=en&oi=scholarr adresinden erişildi.

Pieper,Annemarie. (2012). *Etiğe Giriş*.

Sarı, F. (2021). Cahit Arf'in "Makine Düşünebilir mi ve Nasıl Düşünebilir?" Adlı Makalesi Üzerine Bir Çalışma. *TRT Akademi*, *6*(13), 812-833. doi:10.37679/trta.962940

Staykova, K. (2014). Natural Language Generation and Semantic Technologies. *Cybernetics and Information Technologies*, *14*(2), 3-23. doi:doi:10.2478/cait-2014-0015

TURING, A. M. (1950). I.—Computing Machinery And Intelligence. *Mind*, *LIX*(236), 433-460. doi:10.1093/mind/LIX.236.433

Vincent, J. (2016, 24 Mart). Twitter taught Microsoft's AI chatbot to be a racist asshole in less than a day. *The Verge*. 20 Mayıs 2024 tarihinde https://www.theverge.com/2016/3/24/11297050/tay-microsoft-chatbot-racist adresinden erişildi.

Vogel, J. (2017, 27 Haziran). *Chatbots: Development and Applications*.

Weizenbaum, J. (1966). ELIZA—a computer program for the study of natural language communication between man and machine. *Commun. ACM*, *9*(1), 36-45. doi:10.1145/365153.365168

Yapay Zeka Katmanları. (2024). https://pbs.twimg.com/media/EHJdDIAX4AAxTeK.jpg adresinden erişildi.

BÖLÜM V

Doğal Dil İşleme: Tarihi, Evrimi, Uygulamaları ve Gelecek Perspektifi

Sinem KOÇ[1]
Onur SEVLİ[2]

Giriş

Doğal Dil İşleme (Natural Language Processing- NLP), insan dilinin bilgisayarlar tarafından anlama, işleme ve üretim süreçlerini kapsayan bir yapay zekâ alt alanıdır. Doğal dil işleme, dilbilim, bilgisayar bilimi ve yapay zekâ disiplinlerinin kesişiminde yer alır. Amacı, doğal dili bilgisayarlar aracılığıyla anlamak ve işlemek için algoritmalar ve modeller geliştirmektir. İnsanların

[1] Yüksek Lisans Öğrencisi, Burdur Mehmet Akif Ersoy Üniversitesi, Bilgisayar ve Öğretim Teknolojileri, Burdur/Türkiye, Orcid: 0009-0002-8451-6452, sinem.kc.07@gmail.com
[2] Doç. Dr, Burdur Mehmet Akif Ersoy Üniversitesi, Mühendislik-Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümü, Burdur/Türkiye, Orcid: 0000-0002-8933-8395, onursevli@mehmetakif.edu.tr

günlük yaşamda kullandığı dil oldukça karmaşıktır ve dilin doğru bir şekilde anlaşılması, işlenmesi ve üretilmesi büyük bir zorluk teşkil etmektedir. Doğal dil işleme, bu zorlukları aşmayı hedefleyen yöntem ve tekniklerin toplamıdır. Doğal dil işleme, kelimelerin morfolojik yapılarından cümlelerin sözdizimsel yapısına, metinlerin anlamsal analizinden bağlam içinde dilin pratik kullanımına kadar geniş bir kapsamda faaliyet gösterir. Bu süreçte, kelime çözümlemesi, dil modelleme, duygu analizi, metin sınıflandırma ve çeviri gibi çeşitli görevler gerçekleştirilmektedir. Dilin çok yönlü doğası ve karmaşıklığı, doğal dil işlemenin geniş bir yelpazede teknik ve metodolojiler geliştirmesini gerektirmiştir. Morfolojik analiz, kelimelerin yapısal bileşenlerini inceler ve dilbilgisel özelliklerini belirler. Bu analiz, kelime kökleri, ekler ve yapılarının anlaşılmasını sağlar. Sözdizimsel analiz, cümlelerin gramer yapısını belirleyerek sözcükler arasındaki ilişkileri inceler ve gramer yapısının doğru bir şekilde anlaşılmasını sağlar. Anlamsal analiz, kelimelerin ve cümlelerin anlamlarını işler ve yorumlar. Pragmatik analiz ise dilin bağlam içindeki kullanımını ve iletişimsel amaçlarını incelemektedir.

2. Doğal Dil İşleme (Natural Language Processing -NLP)

Doğal dil işleme, bilgisayar bilimleri ve yapay zekâ alanlarında önemli bir disiplindir ve insan dilini anlamak, yorumlamak, üretmek ve işlemek için kullanılan bilgisayar tabanlı teknikleri kapsar. Doğal dil işleme, metin verilerini analiz etme, anlama ve çıkarımda bulunma yeteneğiyle dikkat çekmektedir. Doğal dil işleme bilgisayar sistemlerinin insan diliyle etkileşimde bulunmasını ve dildeki yapıları kavramasını sağlar.

Doğal dil işlemenin tanımında yer alan anahtar bileşenlerden biri, doğal dili çeşitli seviyelerde işlemeye olanak tanıyan tekniklerdir. Bu seviyeler morfolojik analiz (kelime yapısını analiz etme), sözdizimsel analiz (cümle yapısını analiz etme), anlamsal analiz (anlamın yorumlanması), ve pragmatik analiz (dilin bağlam içinde kullanımını anlama) olarak sıralanabilir. Bu analiz seviyeleri, doğal dil işlemenin metinleri anlamlandırma ve bilgi çıkarma süreçlerinde temel rol oynar.

Doğal dil işlemenin önemi giderek artmaktadır çünkü büyük miktarda metin verisi üretilmekte ve bu verilerin etkili bir şekilde işlenmesi ve anlamlandırılması gerekmektedir. Özellikle sosyal medya, haber siteleri, tıbbi kayıtlar ve diğer birçok alan metin verileriyle doludur ve bu verilerin analiz edilmesi doğal dil işlemenin önemini vurgulamaktadır.

2.1. Doğal Dil İşlemenin Tarihçesi ve Gelişimi

2.1.1. Erken dönem çalışmalar: 1950'ler ve 1960'lar

Doğal dil işleme alanının kökleri, bilgisayar bilimlerinin ve yapay zekânın gelişmeye başladığı 1950'lere dayanmaktadır. Bu dönemde, makinelerin insan dilini anlaması ve işlemesi konusunda ilk adımlar atılmıştır. Alan Turing, 1950 yılında yayımladığı "Computing Machinery and Intelligence" adlı makalesinde, makinelerin zekâ sahibi olup olamayacağını tartışmış ve Turing testini önermiştir. Turing testi, bir makinenin insan gibi düşünme yeteneğini değerlendirmek amacıyla geliştirilmiş olup, bir insanın, makine ile insanı ayırt edememesi durumunda makinenin zeki olarak kabul edilmesini öngörmüştür (Turing, 1950). Turing'in çalışmaları,

doğal dil işlemenin temelini atmış ve dil işleme konusundaki ilk tartışmalara zemin hazırlamıştır.

1950'ler ve 1960'lar, Doğal dil işlemenin temel algoritmaları ve programlarının geliştirildiği önemli bir dönemdir. Makine çevirisi, bu dönemin en dikkat çeken çalışma alanlarından biri olmuştur. Soğuk savaş döneminde, ABD ve Sovyetler birliği arasındaki bilgi akışının hızlı ve doğru bir şekilde sağlanması gereksinimi, dil çeviri sistemlerine olan ilgiyi artırmıştır. 1954 yılında gerçekleştirilen Georgetown-IBM deneyi, İngilizce ‘den Rusça ‘ya yapılan makine çevirisi ile doğal dil işleme alanında önemli bir dönüm noktası olmuştur. Bu deneyde, 60 kelimelik bir kelime dağarcığı ve altı gramer kuralı kullanılarak başarılı sonuçlar elde edilmiştir (Hutchins, 2004). Bu deney, makine çevirisinin potansiyelini göstererek, dil işleme teknolojilerinin geleceği için umut vadetmiştir.

1960'larda, sembolik yöntemler doğal dil işleme alanında ön plana çıkmıştır. Sembolik yöntemler, dilin kural tabanlı analizini içerir ve dilbilimsel kurallar ile sözlüklerin kullanıldığı bir yaklaşımı benimser. Bu dönemin en bilinen örneklerinden biri, Joseph Weizenbaum tarafından geliştirilen ELIZA programıdır. ELIZA, belirli kalıplar ve yanıtlarla basit diyaloglar kurabilen bir programdır. İnsan dilini anlamaktan ziyade, anahtar kelimeleri tanıyıp, önceden belirlenmiş yanıtlar üretebilen ELIZA, bilgisayar-insan etkileşiminin potansiyelini göstermiş ve doğal dil işleme araştırmalarına ilham kaynağı olmuştur (Weizenbaum, 1966).

2.1.2. İstatistiksel yöntemlerin gelişimi: 1980’ler ve 1990’lar

1980'ler ve 1990'lar, doğal dil işlemeye istatistiksel yöntemlerin dahil edildiği ve bu alanda önemli gelişmelerin kaydedildiği bir dönemdir. Bu dönemde, dilin kural tabanlı analizinin sınırlamaları fark edilmiş ve veri tabanlı yaklaşımlar ön plana çıkmıştır. İstatistiksel yöntemler, dil işleme süreçlerinde daha esnek ve ölçeklenebilir çözümler sunarak, dilin karmaşıklığını daha iyi yakalamıştır.

1980'lerde, Gizli Markov modeli (Hidden Markov Models HMM) gibi istatistiksel modeller dil işleme görevlerinde yaygın olarak kullanılmaya başlanmıştır. Gizli markov modeli, özellikle konuşma tanıma ve parça etiketleme (part-of-speech tagging) gibi görevlerde başarılı sonuçlar elde etmiştir. Bu modeller, gözlenen veri dizilerinin altında yatan gizli durum dizilerini modelleyerek, dilin olasılıksal doğasını yakalamıştır (Rabiner, 1989). Gizli markov modeli, dil işleme süreçlerinde belirsizliklerin ve varyasyonların ele alınmasına olanak tanıyarak, Doğal dil işleme alanında önemli bir ilerleme sağlamıştır.

1990'lar, büyük veri kümelerinin ve istatistiksel öğrenme algoritmalarının kullanıldığı bir dönemdir. Bu dönemde, dil corpora'larının (metin veri kümeleri) toplanması ve analiz edilmesi, dil modellerinin geliştirilmesinde önemli bir rol oynamıştır. Özellikle, Penn Treebank gibi anotasyonlu dil corpora'larının kullanımı, dil modellerinin eğitiminde standart bir temel oluşturmuştur (Marcus vd., 1993).

İstatistiksel yöntemlerin dil işleme görevlerinde kullanımı, makine çevirisi, bilgi çekme (information retrieval), bilgi çıkarımı (information extraction) ve metin sınıflandırma gibi birçok alanda

başarıya ulaşmıştır. Örneğin, International Business Machines (IBM) tarafından geliştirilen CANDIDE makine çeviri sistemi, istatistiksel yöntemlerle eğitilmiş ve dil çiftleri arasındaki olasılıksal ilişkileri kullanarak yüksek doğrulukta çeviriler yapabilmiştir (Brown vd., 1990). Bu dönemde, istatistiksel dil modelleri, dil işleme görevlerinde daha geniş uygulama alanı bulmuş ve Doğal dil işlemenin ilerlemesinde önemli bir dönüm noktası olmuştur.

2.1.3. Makine öğrenmesi ve derin öğrenme dönemi: 2000'lerden günümüze

2000'lerden itibaren, makine öğrenmesi ve derin öğrenme teknikleri, doğal dil işleme alanında devrim yaratmıştır. Makine öğrenmesi, bilgisayarların büyük veri setlerinden öğrenmesini ve dil işleme görevlerinde otomatik olarak performanslarını artırmasını sağlamıştır. Derin öğrenme ise, çok katmanlı yapay sinir ağları (deep neural networks) kullanarak, dilin daha karmaşık ve soyut temsillerini öğrenmiştir.

Makine öğrenmesi teknikleri, dil işleme görevlerinde denetimli öğrenme, denetimsiz öğrenme ve yarı denetimli öğrenme gibi farklı yaklaşımlar kullanmıştır. Denetimli öğrenme, etiketli veri kullanarak, belirli dil görevleri için modellerin eğitilmesini sağlamıştır. Bu yaklaşım, metin sınıflandırma, duygu analizi ve adlandırılmış varlık tanıma (named entity recognition) gibi görevlerde başarı göstermiştir (Manning, 2008). Denetimsiz öğrenme ise, etiketsiz veri kullanarak, veri kümelerindeki gizli yapıları keşfetmeyi amaçlamaktadır. Bu yaklaşım, kümeleme ve konu modelleme (topic modeling) gibi görevlerde etkili olmuştur (Blei vd., 2003).

Derin öğrenme teknikleri, doğal dil işleme alanında yeni bir dönemin kapılarını aralamıştır. Özellikle, 2010'lu yılların başlarında, derin öğrenme yöntemleri, dil işleme görevlerinde çığır açıcı sonuçlar elde etmiştir. Derin sinir ağları, dilin daha karmaşık ve soyut temsillerini öğrenerek, dilin anlamını ve yapısını daha iyi kavramıştır. Bu dönemde, uzun kısa süreli bellek (Long Short-Term Memory - LSTM) ve yinelemeli sinir ağları (Recurrent Neural Networks - RNN) gibi modeller, dil işleme görevlerinde yaygın olarak kullanılmıştır (Hochreiter ve Schmidhuber, 1997).

2018 yılında Google tarafından geliştirilen BERT (Bidirectional Encoder Representations from Transformers) modeli, doğal dil işleme alanında önemli bir dönüm noktası olmuştur. BERT, dilin iki yönlü bağlamını kullanarak, metinlerin anlamını daha iyi kavramış ve dil işleme görevlerinde üstün performans sergilemiştir (Devlin vd., 2018). BERT ‘in başarısı, dil modellerinin geliştirilmesinde yeni bir standart oluşturmuş ve dil işleme alanında birçok yeni modelin geliştirilmesine ilham kaynağı olmuştur. Bu modeller arasında GPT (Generative Pre-trained Transformer) ve T5 (Text-to-Text Transfer Transformer) gibi modeller de bulunmaktadır (Raffel vd., 2020, Radford vd., 2019)

Günümüzde, doğal dil işleme araştırmaları hızla ilerlemekte ve dil işleme teknolojilerinin uygulama alanları genişlemektedir. Makine çevirisi, konuşma tanım, metin sınıflandırma, duygu analizi ve bilgi çekme gibi alanlarda büyük ilerlemeler kaydedilmiştir. Özellikle, derin öğrenme tekniklerinin kullanımı, dil işleme görevlerinde çığır açıcı sonuçlar elde edilmesine olanak tanımıştır. Bu gelişmeler, doğal dil işlemenin daha geniş bir yelpazede

uygulanabilirliğini artırmış ve dil teknolojilerinin günlük yaşama entegrasyonunu hızlandırmıştır.

3. Doğal Dil İşlemenin Temel Bileşenleri

Doğal dil işleme, dilin makine tarafından anlaşılması ve işlenmesi sürecinde çeşitli temel bileşenlere dayanır. Bu bileşenler, dilin yapısını, anlamını ve kullanım bağlamını analiz ederek, dilin doğru ve etkin bir şekilde işlenmesini sağlar. Doğal dil işlemenin temel bileşenleri morfolojik analiz, sözdizimsel analiz, anlamsal analiz ve pragmatik analiz olarak sıralanabilir.

3.1. Morfolojik Analiz

Morfolojik analiz, kelimelerin yapısını ve bileşenlerini inceleyen bir dil işleme sürecidir. Bu analiz, kelimelerin köklerini, eklerini ve biçimlerini belirleyerek, dilin temel yapı taşlarını anlamayı amaçlar. Morfolojik analiz, özellikle eklemeli dillerde örneğin, Türkçe dilin anlaşılmasında kritik bir rol oynar. Bu süreçte, kelimelerin kökleri ve ekleri ayrıştırılır, kelime türleri belirlenir ve morfolojik özellikler analiz edilir. Bu analiz, dilin anlamını ve yapısını daha iyi anlamak için temel bir adımdır ve doğal dil işleme sistemlerinde önemli bir bileşendir.

3.2. Sözdizimsel Analiz

Sözdizimsel analiz, cümlelerin yapısını ve gramer kurallarına uygunluğunu inceleyen bir süreçtir. Bu analiz, cümle içindeki kelimelerin dizilimini ve gramer ilişkilerini belirler. Sözdizimsel analiz, cümlelerin doğru bir şekilde anlaşılmasını sağlar ve dilin yapısal özelliklerini ortaya çıkarır. Bu süreçte, cümleler ayrıştırılarak, kelimeler arasındaki gramer ilişkileri (örneğin, özne, nesne, yüklem) belirlenir. Sözdizimsel analiz, dilin anlamını doğru

bir şekilde yorumlamak için önemli bir adımdır ve birçok doğal dil işleme uygulamasında temel bir bileşendir.

3.3. Anlamsal Analiz

Anlamsal analiz (semantik analiz), kelimelerin ve cümlelerin anlamını inceleyen bir süreçtir. Bu analiz, dilin anlamını ve içeriğini doğru bir şekilde anlamayı amaçlar. Anlamsal analiz, kelime anlamlarını, bağlamsal ilişkileri ve dildeki anlamsal belirsizlikleri çözmeyi içerir. Bu süreçte, kelimelerin anlamları belirlenir ve cümlelerin anlam bütünlüğü analiz edilir. Anlamsal analiz, dilin derin anlamını ve bağlamını anlamak için kritik bir adımdır ve doğal dil işleme sistemlerinde önemli bir rol oynar.

3.4. Pragmatik Analiz

Pragmatik analiz, dilin kullanım bağlamını ve dilin anlamını etkileyen dışsal faktörleri inceleyen bir süreçtir. Bu analiz, dilin sosyal ve kültürel bağlamını, konuşma durumlarını ve dil kullanıcılarının niyetlerini anlamayı amaçlar. Pragmatik analiz, dilin doğru bir şekilde yorumlanması ve anlaşılması için önemli bir adımdır. Bu süreçte, dilin bağlamı, konuşmacıların niyetleri ve sosyal ilişkiler gibi faktörler analiz edilir. Pragmatik analiz, dilin anlamını daha geniş bir bağlamda değerlendirmek için kritik bir bileşendir ve doğal dil işleme sistemlerinde önemli bir rol oynar.

4. Doğal Dil İşleme Teknikleri

Doğal dil işleme, dilin makine tarafından anlaşılması, işlenmesi ve üretilmesi süreçlerini içeren çeşitli teknik ve yöntemlerden oluşur. Doğal dil anlama, doğal dil üretimi, dil modellemesi, duygu analizi, makine çevirisi, konuşma tanıma ve

konuşma sentezi gibi temel doğal dil işleme teknikleri şu şekilde sıralanabilir.

4.1. Doğal Dil Anlama (Natural Language Understanding-NLU)

Doğal dil anlama, bilgisayarların insan dilini anlama ve yorumlama yeteneğini geliştirmeyi amaçlar. Doğal dil anlama, dilin anlamını ve bağlamını doğru bir şekilde analiz etmeyi içerir. Bu süreçte, cümleler ayrıştırılarak, kelimeler arasındaki ilişkiler belirlenir ve cümlelerin anlamı çıkarılır. Doğal dil anlama, dilin karmaşıklığını ve belirsizliklerini çözmek için çeşitli yöntemler kullanır. Örneğin, adlandırılmış varlık tanıma (Named Entity Recognition, NER), anlamsal rol etiketleme (Semantic Role Labeling, SRL) ve kelime anlamı çıkarımı (Word Sense Disambiguation, WSD) gibi teknikler, doğal dil anlamanın temel bileşenleridir. Bu teknikler, dilin anlamını ve bağlamını doğru bir şekilde anlamak için kullanılır ve doğal dil işleme sistemlerinde önemli bir rol oynamaktadır.

4.2. Doğal Dil Üretimi (Natural Language Generation -NLG)

Doğal dil üretimi, bilgisayarların insan dilinde anlamlı ve akıcı metinler üretme yeteneğini geliştirmeyi amaçlar. Doğal dil üretimi, bilgisayarların veri ve bilgiyi anlamlı bir şekilde metne dönüştürmesini sağlar. Bu süreçte, veri analiz edilir, dilin gramer kuralları ve yapısal özellikleri kullanılarak metinler oluşturulur. Doğal dil üretimi, raporlama, özetleme, açıklama ve anlatım gibi çeşitli dil üretim görevlerinde kullanılır. Örneğin, otomatik haber yazma, finansal raporlar ve müşteri hizmetleri gibi alanlarda doğal dil üretimi sistemleri yaygın olarak kullanılmaktadır. Doğal dil

üretimi, dilin anlamlı ve akıcı bir şekilde üretilmesini sağlayarak, doğal dil işleme sistemlerinde önemli bir rol oynamaktadır.

4.3. Dil Modellemesi

Dil Modellemesi, dilin olasılık dağılımlarını ve dilin yapısal özelliklerini modellemeyi amaçlar. Dil modelleri, kelime dizilerinin olasılıklarını tahmin ederek, dilin anlamını ve yapısını anlar. Bu süreçte, büyük dil veri tabanları kullanılarak dilin olasılık dağılımları öğrenilir ve dil modelleri oluşturulur. Dil modellemesi, dilin anlamını ve bağlamını doğru bir şekilde analiz etmek için önemli bir adımdır. Örneğin, n-gram modelleri, dilin olasılık dağılımlarını tahmin etmek için yaygın olarak kullanılır. Son dönemde "Transformer" tabanlı modeller (örneğin, GPT ve BERT), dil modellemesinde büyük başarı sağlamıştır. Dil modellemesi, dilin anlamını ve yapısını anlamak için kritik bir bileşendir ve doğal dil işleme sistemlerinde önemli bir rol oynar.

4.4. Duygu Analizi

Duygu analizi (Sentiment Analysis), metinlerdeki duygusal tonları ve kullanıcıların duygusal durumlarını belirlemeyi amaçlar. Bu süreçte, metinler analiz edilerek, olumlu, olumsuz veya nötr duygusal tonlar belirlenir. Duygu analizi, sosyal medya analizleri, müşteri geri bildirimleri ve pazar araştırmaları gibi alanlarda yaygın olarak kullanılır. Örneğin, Twitter, Facebook gibi sosyal medya platformlarında kullanıcıların paylaşımlarındaki duygusal tonlar analiz edilerek, kamuoyunun duygu durumu hakkında bilgi elde edilir. Duygu analizi, dilin duygusal tonlarını ve kullanıcıların duygusal durumlarını anlamak için önemli bir adımdır ve doğal dil işleme sistemlerinde önemli bir rol oynamaktadır.

4.5. Makine Çevirisi

Makine çevirisi (Machine Translation), bir dilden diğerine otomatik çeviri yapmayı amaçlar. Bu süreçte, kaynak dildeki metinler analiz edilerek, hedef dilde anlamlı ve akıcı metinler oluşturulur. Makine çevirisi, dilin anlamını ve yapısını doğru bir şekilde analiz ederek, diller arası çeviri sağlar. Örneğin, Google Translate ve Microsoft Translator gibi çeviri sistemleri, bu teknolojinin yaygın kullanımına örnektir. Makine çevirisi, dilin anlamını ve yapısını doğru bir şekilde analiz ederek, diller arası iletişimi sağlar ve doğal dil işleme sistemlerinde önemli bir rol oynamaktadır.

4.6. Konuşma Tanıma ve Sentezi

Konuşma tanıma (Speech Recognition), konuşulan dili metne dönüştürmeyi amaçlar. Bu süreçte, ses sinyalleri analiz edilerek, konuşulan kelimeler belirlenir ve metne dönüştürülür. Konuşma tanıma, sesli komutlarla çalışan asistanların, çağrı merkezlerinin ve dikte yazılımlarının temelini oluşturur. Örneğin, Siri, Alexa ve Google Assistant gibi sesli asistanlar, konuşma tanıma teknolojisini kullanarak kullanıcıların sesli komutlarına yanıt verir. Konuşma sentezi (Speech Synthesis), metinleri sesli konuşmaya dönüştürmeyi amaçlar. Bu süreçte, metinler analiz edilerek, doğal ve akıcı bir şekilde sesli konuşma oluşturulur. Konuşma tanıma ve sentezi, dilin doğal ve akıcı bir şekilde işlenmesini sağlayarak, doğal dil işleme sistemlerinde önemli bir rol oynamaktadır.

4.7. Makine Öğrenmesi Yöntemleri

Makine öğrenmesi, veri üzerinde öğrenme süreçlerini otomatikleştiren ve iyileştiren algoritmaların geliştirilmesini

hedefleyen bir alandır. Denetimli öğrenme, denetimsiz öğrenme, yarı denetimli öğrenme ve takviyeli öğrenme gibi makine öğrenmesi yöntemlerinin temel prensipleri ve uygulama alanları şu şekilde sıralanabilir:

4.7.1. Denetimli öğrenme

Denetimli öğrenme, etiketli veriler kullanılarak model eğitimi yapmayı amaçlar. Bu yöntem, girdi verileri ile ilgili doğru çıktıları belirleyerek, modelin bu ilişkileri öğrenmesini sağlar. Denetimli öğrenme, sınıflandırma ve regresyon gibi çeşitli görevlerde kullanılır. Örneğin, metin sınıflandırma, duygu analizi ve adlandırılmış varlık tanıma gibi görevlerde denetimli öğrenme teknikleri yaygın olarak kullanılmaktadır. Bu süreçte, büyük veri setleri kullanılarak model eğitimi yapılır ve modelin performansı değerlendirilir. Destek vektör makineleri (SVM), karar ağaçları ve k-en yakın komşu (k-NN) gibi teknikler, denetimli öğrenme yöntemlerine örnektir.

4.7.2. Denetimsiz öğrenme

Denetimsiz öğrenme, etiketlenmemiş veriler kullanarak model eğitimi yapmayı amaçlar. Bu yöntem, verilerdeki gizli kalıpları ve yapıları keşfetmeyi sağlar. Denetimsiz öğrenme, kümeleme ve boyut indirgeme gibi görevlerde kullanılır. Örneğin, konu modelleme, doküman kümeleme ve anlamsal ilişkilerin çıkarımı gibi görevlerde denetimsiz öğrenme teknikleri yaygın olarak kullanılmaktadır. Bu süreçte, büyük veri setleri analiz edilerek, verilerdeki gizli yapılar belirlenir ve model eğitimi yapılır. K-ortalama kümeleme, hiyerarşik kümeleme ve temel bileşen analizi (PCA) gibi teknikler, denetimsiz öğrenme yöntemlerine örnektir.

4.7.3. Yarı denetimli öğrenme

Yarı denetimli öğrenme hem etiketli hem de etiketlenmemiş veriler kullanarak model eğitimi yapmayı amaçlar. Bu yöntem, etiketli veri miktarının sınırlı olduğu durumlarda etkili bir şekilde model eğitimi yapmayı sağlar. Yarı denetimli öğrenme, sınıflandırma ve kümeleme gibi görevlerde kullanılır. Örneğin, düşük maliyetli ve hızlı veri etiketleme süreçlerinde yarı denetimli öğrenme teknikleri yaygın olarak kullanılmaktadır. Bu süreçte, etiketli ve etiketlenmemiş veriler birlikte analiz edilerek, model eğitimi yapılır ve modelin performansı değerlendirilir.

4.7.4. Takviyeli öğrenme

Takviyeli öğrenme, bir ajanın çevresiyle etkileşime girerek ödül veya ceza yoluyla öğrenme sürecini optimize etmeyi amaçlar. Bu yöntem, ajanın belirli bir görevde en iyi performansı göstermesini sağlar. Takviyeli öğrenme, oyun teorisi, robotik ve otonom sistemler gibi çeşitli alanlarda kullanılır. Örneğin, Go, satranç ve video oyunları gibi stratejik oyunlarda takviyeli öğrenme teknikleri büyük başarı elde etmiştir. Bu süreçte, ajan çevresiyle etkileşime girerek, ödül ve ceza sinyallerini kullanarak öğrenme sürecini optimize eder. Q-learning, SARSA ve derin Q-ağları (DQN) gibi teknikler, takviyeli öğrenme yöntemlerine örnektir.

5. Doğal Dil İşleme Uygulamaları

Doğal dil işleme, insan dilini anlamak, analiz etmek ve üretmek için kullanılan çeşitli teknik ve yöntemlerin bir birleşimidir. Bu teknolojiler, günlük yaşamda ve iş dünyasında geniş bir yelpazede uygulamalara sahiptir. Doğal dil işlemenin önemli uygulama alanları ve kullanılan araçlar şu şekilde sıralanabilir.

5.1. Sohbet Robotları (Chatbots)

Sohbet robotları, insanlarla doğal bir dilde iletişim kurabilen otomatik sistemlerdir. Bu sistemler, genellikle metin tabanlı olarak çalışır ve kullanıcıların sorularını yanıtlamak, bilgi sağlamak veya belirli görevleri yerine getirmek için kullanılır. Sohbet robotları genellikle işletmelerin müşteri hizmetleri, web sitelerindeki yardım araçları veya mesajlaşma uygulamalarındaki otomatik yanıtlar olarak kullanılır. Örnek olarak, Facebook Messenger'daki otomatik mesajlaşma sistemleri veya çevrimiçi alışveriş sitelerindeki canlı destek botları verilebilir. Bu sistemler, doğal dil işleme teknikleri kullanarak gelen metinleri analiz eder, anlamlandırır ve uygun yanıtları üretir.

5.2. Bilgi Çekme (Information Retrieval)

Bilgi çekme, büyük veri kümelerinden belirli bilgileri çıkarma sürecidir. Doğal dil işleme ’de bu alanındaki uygulamalar, metin madenciliği ve bilgi yönetimi teknikleriyle ilişkilidir. Özellikle, metin madenciliği, büyük veri setlerini analiz ederek belirli bilgileri çıkarmak için doğal dil işleme tekniklerini kullanır. Örneğin, bir arama motoru kullanıcıların sorgularını analiz ederek en uygun sonuçları vermek için doğal dil işleme tekniklerini kullanır. Ayrıca, belirli konularda bilgi toplamak için web sitelerini veya veri tabanlarını tarayarak bilgi çıkarma işlemleri de gerçekleştirilir.

5.3. Bilgi Çıkarımı (Information Extraction)

Bilgi çıkarımı, metinlerden yapılandırılmış bilgi çıkarma sürecidir. Bu süreçte, belirli yapılar ve ilişkiler tanımlanır ve metinlerden bu yapılar çıkarılarak bilgi elde edilir. Örneğin, bir metinde geçen kişi isimleri, tarihler, mekanlar veya olaylar gibi

yapılar çıkarılarak bir veri tabanına kaydedilebilir. Bu tür bilgi çıkarımı işlemleri genellikle özel amaçlı yazılımlar veya sistemler kullanılarak gerçekleştirilir. Doğal dil işleme, bu süreçte metinleri analiz eder, yapıları tanımlar ve çıkarır, böylece kullanıcıların belirli bilgilere erişmesini sağlar.

5.4. Metin Sınıflandırma

Metin sınıflandırma, metinlerin belirli kategorilere veya sınıflara atanması işlemidir. Bu işlem genellikle makine öğrenmesi teknikleriyle birlikte kullanılır. Örneğin, bir e-posta filtreleme sistemi, gelen e-postaları spam veya önemli olarak sınıflandırmak için metin sınıflandırma tekniklerini kullanabilir. Ayrıca, duygu analizi gibi alanlarda da metinlerin duygusal tonlarına göre sınıflandırılması yaygın bir uygulamadır. Doğal dil işleme, metinlerdeki özellikleri analiz ederek ve makine öğrenmesi algoritmalarıyla eğitilerek bu sınıflandırma işlemini gerçekleştirir.

5.5. Özetleme (Summarization)

Özetleme, uzun metinlerin kısa ve öz bir şekilde özetlenmesi işlemidir. Bu işlem genellikle metinlerin önemli noktalarını vurgulamak ve ana fikirleri belirtmek için kullanılır. Özetleme işlemi, genellikle otomatik metin özetleme sistemleri veya yazılımları kullanılarak gerçekleştirilir. Bu sistemler, metinlerdeki önemli cümleleri veya paragrafları belirleyerek, kullanıcıların hızlı bir şekilde metinleri anlamasını sağlar. Özetleme, makine öğrenmesi teknikleri ve doğal dil işlemenin dil anlama ve işleme yetenekleriyle birlikte kullanılarak gerçekleştirilir.

6. Doğal Dil İşlemenin Kullanıldığı ve Etki Ettiği Alanlar

Doğal dil işleme teknolojileri, eğitim ve öğrenme süreçlerinde önemli bir dönüşüm potansiyeli sunmaktadır. Bu teknolojilerin öğrenci motivasyonu, öğrenme süreçlerine aktif katılım ve bilgi edinme becerileri üzerindeki etkileri, eğitim alanında yapılan araştırmalarla giderek daha açık bir şekilde görülmektedir. Ayrıca, sağlık sektöründe doğal dil işleme uygulamaları, tıbbi metin analizi ve hasta kayıtlarının otomatik işlenmesi gibi alanlarda önemli bir rol oynamaktadır. Benzer şekilde, hukuk ve adalet sistemlerindeki doğal dil işleme teknolojileri, hukuki belgelerin analizi ve hukuki süreçlerin optimize edilmesinde kullanılmaktadır. Sosyal medya üzerindeki doğal dil işleme analizleri ise, kullanıcıların duygusal durumları, ilgi alanları ve etkileşimleri hakkında değerli içgörüler sunmaktadır.

6.1. Doğal Dil İşleme Teknolojilerinin Eğitim ve Öğrenme Üzerindeki Etkileri

Doğal dil işleme teknolojileri, eğitim ve öğrenme alanında önemli etkiler oluşturma potansiyeline sahiptir. Bu etkiler öğrenci motivasyonu, öğrenme süreçleri ve bilgi edinme becerileri üzerine yapılan araştırmalarda ortaya çıkmaktadır.

Doğal dil işleme tabanlı uygulamaların öğrenci motivasyonu üzerinde etkileri görülmektedir. Araştırmalar, öğrencilerin doğal dil işleme destekli eğitim materyalleri ile daha fazla ilgilendiğini ve öğrenme sürecine aktif katılım sağladığını göstermektedir. Bu tür uygulamaların öğrencilerin öğrenme istekliliğini artırma potansiyeline sahip olduğu vurgulanmaktadır.

Öğrenme süreçleri ve doğal dil işleme arasındaki ilişki de önemlidir. Öğrencilerin farklı öğrenme stillerine uygun doğal dil işleme uygulamalarının etkilerini değerlendiren çalışmalar, öğrencilerin daha etkili bir şekilde öğrenebileceğini göstermektedir. Özellikle, görsel, işitsel veya dokunsal öğrenme tercihlerine göre tasarlanmış doğal dil işleme uygulamalarının öğrenci başarısını artırdığına dair bulgular mevcuttur.

Bilgi edinme becerileri açısından, doğal dil işleme teknolojilerinin rolü de önemlidir. Öğrencilerin bilgiye erişimde ve anlamada doğal dil işlemenin sunduğu araçların kullanımının, bilgiye daha hızlı ve etkili bir şekilde ulaşmalarına yardımcı olduğu gözlemlenmektedir. Özellikle büyük veri kaynaklarından bilgi çıkarma ve derinlemesine metin analizi gibi alanlarda doğal dil işlemenin önemi artmaktadır.

Doğal dil işlemenin eğitim ve öğrenme süreçlerindeki etkilerini anlamak için yapılan araştırmalar, teknolojinin öğrencilerin motivasyonunu artırdığını, öğrenme süreçlerini optimize ettiğini ve bilgi edinme becerilerini güçlendirdiğini ortaya koymaktadır. Bu çalışmalar, eğitim alanındaki doğal dil işleme uygulamalarının gelecekteki potansiyelini de vurgulamaktadır.

6.2. Sağlık Alanında Doğal Dil İşleme Uygulamaları

Sağlık alanındaki doğal dil işleme uygulamaları, tıbbi metinlerin analizi, hasta kayıtlarının otomatik olarak işlenmesi ve tıbbi önerilerin oluşturulması gibi önemli alanlarda kullanılmaktadır. Bu uygulamalar, sağlık çalışanlarının iş yükünü azaltırken hastaların daha etkili bir şekilde tedavi edilmesine yardımcı olabilir. Ayrıca, doğal dil işlemenin sağlık sektöründeki etkilerini inceleyen

akademik çalışmalar, teknolojinin sağlık hizmetleri kalitesini artırma potansiyeline sahip olduğunu göstermektedir.

Doğal dil işlemenin sağlık alanındaki uygulamalarından biri tıbbi metinlerin analizidir. Bu alandaki çalışmalar, doğal dil işlemenin tekniklerinin tıbbi literatürün otomatik olarak taranması, hastalık teşhisleri ve tedavi önerileri gibi alanlarda kullanıldığını göstermektedir. Örneğin, "PubMed" gibi yayın veri tabanlarından tıbbi makalelerin otomatik olarak analiz edilmesi ve benzer vakaların karşılaştırılması gibi uygulamalar bu alanda yaygın olarak kullanılmaktadır.

Hasta kayıtlarının otomatik işlenmesi de sağlık sektöründe doğal dil işlemenin önemli bir uygulama alanıdır. Elektronik sağlık kayıtları üzerinde yapılan çalışmalar, doğal dil işlemenin hastaların sağlık geçmişlerini analiz etme, belirli semptomları tanımlama ve tedavi planlarını optimize etme konularında kullanıldığını göstermektedir. Bu tür uygulamalar, sağlık çalışanlarının hastaların sağlık durumu hakkında daha hızlı ve doğru kararlar vermesine yardımcı olabilmektedir. Tıbbi önerilerin oluşturulması ve hasta tedavisi konusundaki doğal dil işleme uygulamaları da önemlidir. Bu alandaki çalışmalar, doğal dil işlemenin hasta verilerini analiz ederek kişiselleştirilmiş tedavi planları oluşturma, ilaç etkileşimlerini değerlendirme ve tedavi sonuçlarını izleme konularında kullanıldığını göstermektedir. Özellikle yapay zekâ destekli doğal dil işleme sistemleri, hastalara daha etkili ve kişiselleştirilmiş tedavi seçenekleri sunabilmektedir.

Bu çalışmaların ışığında, doğal dil işlemenin sağlık sektöründeki uygulamaları üzerine yapılan akademik araştırmaların

önemi ve etkisi açıkça görülmektedir. Teknolojinin sağlık hizmetlerinde verimliliği artırma, hasta bakımını iyileştirme ve sağlık sonuçlarını optimize etme potansiyeli büyüktür. Ancak, bu uygulamaların etik ve gizlilik konuları da dikkate alınmalı ve daha geniş bir tartışma bağlamında ele alınmalıdır.

6.3. Doğal Dil İşlemenin Hukuk Sistemindeki Rolü

Doğal dil işleme teknolojileri, hukuk sisteminde önemli bir rol oynamaktadır. Bu teknolojilerin kullanımı, hukuki belgelerin analizi, yasal metinlerin çevirisi ve hukuki kararların otomatik olarak değerlendirilmesi gibi alanlarda etkili bir şekilde gerçekleşmektedir. Doğal dil işleme teknolojileri, hukuki belgelerin otomatik olarak analiz edilmesi ve yasal terminolojinin anlaşılması gibi konularda önemli bir rol oynamaktadır. Özellikle büyük miktarda hukuki metnin bulunduğu davalarda, doğal dil işleme tabanlı sistemlerin kullanılması hızlı ve doğru bilgi erişimi sağlayarak hukuki süreçleri optimize etmektedir. Farklı dillerdeki yasal belgelerin doğru ve hassas bir şekilde çevrilmesi, uluslararası hukuk alanında önemli bir ihtiyaçtır. Doğal dil işleme teknolojileri, dil çevirisi konusunda geliştirilen algoritmalar ve sistemler aracılığıyla bu ihtiyacı karşılamaya yardımcı olmaktadır. Özellikle çok dilli doğal dil işleme çalışmaları, farklı hukuk sistemlerine ve kültürel yapıya sahip ülkeler arasında iletişimi kolaylaştırmaktadır.

Hukuki belgelerin analizi, doğal dil işlemenin hukuk alanındaki önemli uygulamalarından biridir. Bu alanda yapılan araştırmalar, doğal dil işleme tekniklerinin hukuki belgelerin içeriğini anlama, hukuki terimlerin doğru şekilde yorumlanması ve belgeler arasında ilişkilerin analiz edilmesi gibi konularda kullanıldığını göstermektedir. Özellikle büyük veri setlerindeki

hukuki metinlerin derinlemesine analizi, hukuk alanındaki karar alma süreçlerini optimize etme potansiyeline sahiptir.

6.4. Doğal Dil İşleme ve Sosyal Medya Analizi

Sosyal medya platformları, milyonlarca kullanıcının günlük olarak paylaşımlar yaptığı, etkileşimlerde bulunduğu ve duygularını ifade ettiği geniş bir veri kaynağıdır. Bu platformlardaki veriler, kullanıcıların sosyal ilişkileri, eğilimleri, beklentileri ve duygusal durumları hakkında değerli bilgiler içermektedir. Doğal dil işleme, bu büyük veri setlerini analiz ederek önemli içgörüler elde etmeyi mümkün kılmaktadır.

Kullanıcıların paylaşımları genellikle duygusal ifadeler içerir; bu ifadeler olumlu, olumsuz veya tarafsız olabilir. Doğal dil işleme teknikleri, bu duygusal içerikleri tespit ederek genel bir duygu analizi yapabilir ve kullanıcıların genel duygusal durumları hakkında bilgi verebilir. Örneğin, bir ürünün veya hizmetin sosyal medyada nasıl algılandığını anlamak için duygu analizi kullanılabilir.

Sosyal medya platformlarındaki içerikler genellikle farklı kategorilere ayrılabilir: haberler, eğlence, spor, politika, sağlık vb. doğal dil işleme, bu içerikleri otomatik olarak sınıflandırarak belirli kategorilerdeki içerikleri gruplayabilir ve analiz edebilir. Bu sayede, belirli bir konuyla ilgili sosyal medya konuşmalarını anlama ve takip etme imkânı sağlar.

Kullanıcılar sosyal medya platformlarında paylaşımlar yaparken, beğeniler, yorumlar, paylaşımlar ve etiketlemeler gibi etkileşimlerde bulunurlar. Doğal dil işleme, bu etkileşim verilerini analiz ederek kullanıcıların içeriklere nasıl tepki verdiğini, hangi

konuların daha fazla ilgi gördüğünü ve hangi içeriklerin viral hale geldiğini anlamak için kullanılır. Bu analizler, pazarlama stratejileri oluşturma, kampanyaları yönetme ve kullanıcı katılımını artırma konularında değerli içgörüler sağlar.

6.5. Görüntü ve Metin Entegrasyonu

Görüntü ve metin entegrasyonu, görsel verilerle metin verilerini birleştirerek daha derinlemesine bir analiz sağlar. Bu entegrasyon hem görüntü tanıma sistemlerini hem de doğal dil işleme tekniklerini içerir. Özellikle sosyal medya platformları gibi veri yoğun ortamlarda, bu entegrasyon sayesinde kullanıcıların paylaşımlarını daha iyi anlama ve içgörüler elde etme imkânı sağlanır.

Geleneksel doğal dil işleme çalışmaları genellikle metin verilerine odaklanırken, görsel verilerin de analiz edilmesi ve bu verilerle birlikte metinlerin işlenmesi önemli bir adımdır. Bu entegrasyon, daha kapsamlı ve zengin içerik analizleri yapılmasını sağlar ve kullanıcıların daha derinlemesine anlaşılmasına yardımcı olur.

Görüntü ve metin entegrasyonunun yöntemlerine ve tekniklerinde ise bu entegrasyon genellikle derin öğrenme (deep learning) teknikleriyle gerçekleştirilir. Görsel veriler için evrişimli sinir ağları (Convolutional Neural Networks- CNNs) ve metin verileri için tekrarlayan sinir ağları (Recurrent Neural Networks - RNNs) gibi derin öğrenme modelleri kullanılır. Bu modeller, görüntü ve metin verilerini bir araya getirerek ortak bir temsili oluştururlar ve bu temsiller üzerinden analizler yapılır.

Görüntü ve metin entegrasyonunun doğal dil işleme üzerindeki etkilerinde birçok farklı alanda kullanılabilir. Örneğin, sosyal medya platformlarındaki görsel ve metin içeriklerin birlikte analiz edilmesi sayesinde kullanıcıların duygusal durumları, ilgi alanları ve etkileşimleri daha iyi anlaşılabilir. Ayrıca, e-ticaret platformlarında ürün incelemeleri ve görsellerinin bir araya getirilerek daha kapsamlı bir değerlendirme yapılabilir.

7. Doğal Dil İşleme Alanında Karşılaşılan Zorluklar ve Çözümler

Doğal dil işleme, son yıllarda yapay zekâ ve dilbilim alanlarında önemli ilerlemeler kaydetmiş bir teknoloji olarak öne çıkmaktadır. Ancak, bu alandaki ilerlemelere rağmen, doğal dil işleme modellerinin geliştirilmesi ve uygulanması sırasında birçok zorlukla karşılaşılmaktadır. Bu zorlukların üstesinden gelmek için çeşitli stratejiler ve çözümler geliştirilmiştir.

7.1. Dil Çeşitliliği ve Çok Dilli Doğal Dil İşleme

Doğal dil işlemenin en büyük zorluklarından biri, dil çeşitliliği ve çok dilli iletişimde yaşanan sorunlardır. Farklı dillerdeki metinlerin analizi ve işlenmesi, dilbilimsel farklılıklar, yapılar ve kelime dağarcıkları nedeniyle karmaşık hale gelir. Özellikle az sayıda kaynak ve veriye sahip olan diller, dil işleme modelleri için daha büyük zorluklar oluşturur.

Çok dilli doğal dil işleme, birçok dilde çalışabilen ve farklı diller arasında geçiş yapabilen modellerin geliştirilmesini gerektirir. Bu modeller, çeviri, metin anlama ve duygu analizi gibi çoklu dil işleme görevlerinde etkin ve doğru sonuçlar sağlamalıdır. Ancak dil arasındaki çevirilerde doğruluk ve tutarlılık sağlamak, farklı dil

yapılarını ve anlam kaymalarını doğru bir şekilde ele almak büyük bir zorluk oluşturur.

Gelecekte, dil çeşitliliği ve çok dilli doğal dil işleme alanında odaklanılması gereken önemli konular arasında, az kaynaklı dillerdeki çalışmaların artırılması, çok dilli veri setlerinin oluşturulması ve dilbilimsel çeşitliliklerin daha iyi anlaşılmasında yer alır.

7.2. Büyük Veri ve Hesaplama Kaynakları

Doğal dil işlemenin bir diğer önemli zorluğu, büyük veri setleriyle çalışma ve bu verileri işleme kapasitesidir. Günümüzde, internet ve dijital platformlardan elde edilen büyük miktarda metin verisi, dil işleme modellerinin eğitiminde ve performansında önemli bir rol oynamaktadır. Ancak bu veri setlerinin toplanması, işlenmesi ve analiz edilmesi büyük hesaplama kaynakları gerektirir.

Yapay zekâ ve doğal dil işleme modellerinin eğitimi için gerekli olan büyük veri setleri, veri güvenliği, gizlilik ve yasal konular gibi önemli sorunları da beraberinde getirir. Özellikle kişisel verilerin kullanımı ve korunması, doğal dil işleme çalışmalarında dikkat edilmesi gereken hassas bir konudur.

Gelecekte, büyük veri ile çalışma konusunda daha etkili ve verimli yöntemlerin geliştirilmesi, veri güvenliği ve gizliliğinin korunması için daha iyi politikaların oluşturulması ve hesaplama kaynaklarının daha verimli kullanılması için yeni teknolojilerin geliştirilmesi gerekecektir.

7.3. Etik ve Gizlilik Sorunları

Doğal dil işleme çalışmalarında karşılaşılan bir diğer önemli zorluk, etik ve gizlilik sorunlarıdır. Bu sorunlar, özellikle hassas konuları içeren metinlerin analizi ve işlenmesi durumunda ön plana çıkar. Örneğin, sağlık verileri, finansal bilgiler ve kişisel iletişimler gibi özel ve güvenlik gerektiren verilerin doğal dil işleme modelleri tarafından işlenmesi, kullanımı ve paylaşılması etik ve gizlilik açısından önemli sorunlar doğurabilir.

Etik sorunlar, doğal dil işlemenin kullanım alanlarıyla doğrudan ilişkilidir. Örneğin, duygu analizi gibi tekniklerin, kişisel ifadeleri analiz ederek duygusal durumları anlama ve kullanma potansiyeli vardır. Bu durumda, kullanıcıların mahremiyeti ve duygusal gizliliği önemli bir endişe kaynağıdır. Benzer şekilde, sosyal medya verileri gibi geniş veri setlerinin analizi, kullanıcıların tercihleri ve alışkanlıkları hakkında derinlemesine bilgiler sunabilir, bu da kullanıcıların gizliliği açısından ciddi bir sorun oluşturabilmektedir.

Gizlilik sorunları, doğal dil işleme modellerinin büyük veri setlerini analiz etme kapasitesiyle ilgilidir. Bu modeller, büyük miktarda veriyi öğrenerek anlam çıkarabilir ve genel eğilimleri belirleyebilir. Ancak bu süreçte, kişisel bilgilerin korunması ve veri güvenliği önemli bir meseledir. Özellikle, hassas verilerin yanlış ellere geçmesi veya kötü niyetli kullanımlar için kullanılması, doğal dil işlemenin gizlilik sorunlarını vurgular.

Gelecekte, etik ve gizlilik sorunlarına yönelik çözümler ve önlemler geliştirilmesi gerekecektir. Bu çözümler arasında, veri anonimleştirme teknikleri, veri şifreleme yöntemleri, kullanıcı izni

ve kontrolü gibi etik standartların oluşturulması ve doğal dil işleme modellerinin kullanımında şeffaflık ve hesap verebilirlik ilkelerinin benimsenmesi yer alır.

8. Akademik Literatürde Doğal Dil İşleme Uygulamaları

İnternet ve sosyal medyanın artan kullanımı, bilgi oluşturma ve yayma süreçlerinde önemli bir rol oynamaktadır. Hızla artan verileri anlamlandırabilmek için insanların konuştukları dilleri makinelerin anlamasına yönelik yöntemler popülerlik kazanmıştır. Son yıllarda, doğal dil işleme alanında ön-eğitimli modellerin başarısı dikkat çekmektedir. Hark (2022), sahte haberlerin tespitine yönelik olarak gerçekleştirdikleri çalışmada ön-eğitimli modellerin kullanımı ile %91'e varan başarı elde etmiştir.

İnsanlar, internet erişimine sahip cihazlarla gözlemledikleri her türlü olağan veya olağandışı durumu anlık olarak sosyal medyada paylaşabiliyorlar. Özellikle Twitter, bu tür durumların ilk duyulduğu ve hızlı bir şekilde yayıldığı platformlardan biridir, bu nedenle acil müdahale ekipleri ve medya şirketleri için önemli bir bilgi kaynağıdır. Ancak, paylaşılan içeriklerin her zaman gerçek bir olayı yansıttığı garanti edilemez. Doğal dil işleme, insanların kullandığı dillerin bilgisayarlar tarafından anlaşılmasını sağlar. Derin sinir ağı temelli Google BERT modeli, kelimeler ve cümleler arasındaki bağlamsal ilişkileri ortaya koymada son derece başarılı olan ve yaygın kullanılan bir yöntemdir. Sevli ve Kemaloğlu (2021) tarafından gerçekleştirilen çalışmada deprem, kaza, olumsuz hava koşulları gibi felaket durumlarına ilişkin tweetlerin BERT modeliyle sınıflandırılması yapılmış ve %98'in üzerinde bir doğruluk elde edilmiştir.

Sahte haberler, bilinçli veya bilinçsiz şekilde çeşitli iletişim kanalları aracılığıyla yayılan gerçeklikten uzak uydurma içeriklerdir. Dijital ve sosyal medya, haberlerin hızlı bir şekilde yayılmasını sağlayarak doğruluk kontrolünün zorlaştığı ortamlardır. Bu durum, dezenformasyon ve yanlış bilgilendirme risklerini artırır. Yapay zekâ temelli doğruluk kontrolü, sahte haberlerin etkisini azaltmada önemli bir rol oynayabilir. Toğaçar vd.(2022) tarafından gerçekleştirilen çalışmada doğal dil işleme yöntemleri ve Uzun Kısa Süreli Bellek (Long Short Term Memory - LSTM) modeli kullanılarak 6335 haber başlığı ve içeriği analiz edilmiştir. Çalışmada elde edilen %99,83'lük doğruluk gelecekte yürütülecek benzer çalışmalar için umut vericidir.

Yapay zekâ ve yapay öğrenme, önemli teknolojik gelişmeler arasında yer alır ve bilgisayarların insan benzeri zekâ ve öğrenme yeteneklerini kazanmasına odaklanır. Bu teknolojiler birçok sektörde etkili olup, insan yaşamını kolaylaştırır ve üretkenliği artırır. Ancak, olumlu etkilerinin yanı sıra bazı olumsuz etkileri de beraberinde getirir. Araştırmacılar, yapay zekânın gelecekteki etkileri konusunda farklı görüşlere sahiptir. Altıntop (2023) çalışmasında, ChatGPT gibi popüler yapay zekâ tabanlı teknolojilerinin potansiyelini anlamak amacıyla bir inceleme yaparak farklı görüşleri değerlendirmiştir.

Yapay zekâ ve iletişim arasındaki ilişki, özellikle ChatGPT gibi yapay zekâ temelli programların etkisiyle, iletişim alanında yeni tartışmalara yol açmaktadır. Bu teknolojilerin yaygınlaşmasıyla birlikte iletişimde önemli değişimler ve etkiler beklenmektedir. Koçyiğit ve Darı (2023), yapay zekâ ve iletişim arasındaki ilişkiyi, özellikle ChatGPT üzerinden inceleyerek, gelecekteki iletişim

trendlerini irdelemiş ve bu teknolojilerin avantajları, dezavantajları ve etkileri üzerine odaklanmışlardır. Bu dönemi, "İnsanlaşan Dijitalleşme Çağı" olarak adlandırarak, alana yeni bir bakış açısı sunmayı hedeflemişlerdir.

Yiğit vd. (2023) tarafından gerçekleştirilen çalışmada, doğal dil işleme teknolojisi olan ChatGPT'nin sağlık alanındaki kullanımı incelenmiştir. ChatGPT, sağlık profesyonellerine bilimsel yayın hazırlama, eğitim planlama ve sağlık hizmetlerinde çeşitli imkânlar sunmaktadır. Sağlık hizmetlerinde kişiselleştirilmiş tedavi, halka erişilebilir sağlık bilgileri ve sağlık okuryazarlığının geliştirilmesi gibi faydaları bulunmaktadır. Ancak, etik ve hukuki sorunlar da beraberinde gelir. Veri güvenliği ve hasta mahremiyeti gibi konuların yanı sıra, teknoloji geliştiricileri ve sağlık hizmeti sağlayıcıları arasında iş birliği gereklidir. ChatGPT'nin etkin kullanımı için daha fazla veri ve iyileştirmeler gerekmektedir (Yiğit vd., 2023).

Türkçe metinler üzerinden duygu analizi konusunda sınırlı çalışmalar bulunmaktadır. Türkçe duygu analizi için iki yöntem önerilmektedir: Çok dilli BERT modeli veya Türkçe metinlerin İngilizce'ye çevrilerek BERT'in ana modelinin kullanılması. Açıkalın vd. (2020) Türk film yorumları ve otel yorumu veri setleri üzerinde yaptıkları denemelerde, duygu analizi konusunda BERT modelinin kullanımı ile yüksek doğruluk elde etmişlerdir.

Sosyal medyanın ve internetin yaygınlaşması, siber zorbalık eylemlerinin hızla artmasına neden olmuştur. Bu durum, dünya genelinde siber zorbalığa maruz kalan bireylerin sayısında artış ve mağduriyetlerin ciddi etkileriyle sonuçlanmaktadır. Yazğılı ve

Baykara (2022) sosyal medyadan elde edilen ve 3000 örnekten oluşan Türkçe veri seti üzerinde siber zorbalık tespitini amaçlamışlardır. Çalışmalarında literatürde aygın kullanılan sınıflandırıcılar yanında Bagging ve Boosting yöntemlerinin başarılarını karşılaştırmalı olarak incelemişlerdir.

Ozan ve Taşar (2021) tarafından gerçekleştirilen çalışmada, alana özgü cümlelerin otomatik olarak etiketlenmesi için bir yöntem geliştirilmesi amaçlanmıştır. Eğitim verileri, müşteri temsilcileri ile web sitesi ziyaretçileri arasındaki sohbetlerden alınan kısa konuşma cümlelerinden oluşmaktadır. Anlamlı diyaloglar üretebilen bir sohbet robotu geliştirmek için yaklaşık 14 bin ziyaretçi girişini 10 temel kategoriye manuel olarak etiketleyip ardından bu verileri transformatör tabanlı bir dil modeli ile sınıflandırmışlardır. Çalışmada en iyi performans BERT modeliyle elde edilmiştir.

Özdil vd. (2021) çevrimiçi reklam metinlerinin sektörel olarak otomatik sınıflandırılması için doğal dil işleme tabanlı bir yöntem önermişlerdir. Çalışmada kullanılan veri seti 12 farklı sektöre ait 21.000 etiketli reklam metninden oluşmaktadır. Metin sınıflandırmada Türkçe için önceden eğitilmiş BERT modeli kullanılmıştır. Çalışmada kullanılan modelin etkinliği farklı verimlilik parametreleri ile ortaya konmuştur.

Öztürk vd. (2020) gerçekleştirdikleri çalışmada Türkçe'de sıkça yapılan 'de/da' ekiyle ilgili yazım hatalarını ele almışlardır. Bu hataların morfolojik karmaşıklık nedeniyle tespiti zor olduğundan, yazım düzeltme araçları genellikle bu hataları düzeltmekte zorluk yaşamaktadır. BERT modeli bağlamı dikkate alarak daha doğru kelime yerleştirmeleri üretebilmektedir. Çalışmada ele alınan

problemde, BERT modelinin bağlamsız kelime yerleştirmelerine kıyasla daha yüksek performans gösterdiği ortaya konmuştur.

Bayram (2022) gerçekleştirdiği çalışmada COVID-19 pandemisinin farklı boyutlarına odaklanarak sosyal medya verilerini ve haber koleksiyonlarını incelemiştir. Sosyal medya verileri daha çok insan tepkilerini yansıtırken, haber koleksiyonları daha objektif bilgi sağlar. İnceleme, pandeminin etkilerini çeşitli düzeylerde analiz ederken istatistiksel verilerle en çok bahsedilen sorunları belirlemeyi ve makine öğrenmesi teknikleri ile pandeminin neden olduğu sorunları detaylı şekilde incelemeyi amaçlamıştır. Bulgular, ekonominin öne çıkan bir sorun olduğunu göstermiştir ve sözcüksel ağlar aracılığıyla ilişkili diğer konular da ortaya konmuştur. Çalışma, pandeminin tüm yönlerini daha iyi anlamak için daha çok makine öğrenmesi ve doğal dil işleme temelli araştırmaya ihtiyaç olduğunu vurgulamaktadır.

Ayan vd. (2021) tarafından gerçekleştirilen çalışmada Türkiye'de önde gelen 25 büyük şirketin web sitelerinden elde edilen verilerle, anahtar kelimelerin çıkarılmasını içeren bir analiz yapılmıştır. Hem metodolojik hem de sonuçsal açıdan doğal dil işleme alanında yeni bir bakış açısı sunmayı amaçlayan çalışmanın analiz sonuçları, şirketlerin anahtar kelimelerinin kümelenmesiyle iş alanlarına ilişkin önemli ipuçları sağlamaktadır.

Ozan ve Taşar (2021) tarafından gerçekleştirilen çalışmada müşteriler ile müşteri temsilcileri arasındaki yazışmalar incelenerek kısa konuşma cümleleri anlamsal özelliklerine göre analiz edilmiştir. 46 farklı kategoride örneklerden oluşan veri seti üzerinde, GPT-2 ve BERT modelleri ile sınıflandırılmıştır.

İşeri vd. (2021) tarafından gerçekleştirilen çalışmada çağrı merkezi işlemlerini otomatize etmek amacıyla makine öğrenmesi ve doğal dil işleme teknikleri kullanılarak bir sohbet robotu geliştirilmesi amaçlanmıştır. Çalışmada kullanılan veri seti sıkça sorulan soruların yanı sıra telefon konuşmaları ve anlık ileti uygulamalarından alınan verileri içermektedir. Geliştirilen sohbet robotu, kullanıcıların firma veya ürünlerle ilgili sorularına hızlı ve etkili bir şekilde yanıt vermeyi sağlamaktadır. Kullanıcıların girdileri doğrultusunda model, uygun niyet ve cevapları içeren yanıtlar üretmektedir. Test süreci boyunca sohbet robotunun, sorulara yüksek doğrulukla cevap verdiği gözlemlenmiştir.

Munikar vd. (2019), tarafından gerçekleştirilen çalışmada, duygu sınıflandırması için transformer tabanlı derin öğrenme mimarisi kullanılmıştır. Geleneksel olarak ikili duygu sınıflandırması üzerine odaklanan çoğu doğal dil işleme modelinin aksine, bu çalışmada detaylı duygu sınıflandırma görevi ele alınmıştır. Yapılan deneyler, BERT modelinin diğer popüler modellere göre daha iyi performans sergilediğini göstermektedir. Ayrıca çalışmada, transfer öğrenmenin doğal dil işleme sürecinde etkin bir şekilde kullanılabilirliği de vurgulamaktadır.

İnteraktif Doğal Dil İşleme (iNLP), yapay zekânın nihai hedeflerine uyum sağlarken mevcut çerçevelerdeki sınırlamaları ele almayı amaçlayan yeni bir paradigmadır. Bu paradigma dil modellerini, kullanıcı ihtiyaçlarını anlamak, yanıtları kişiselleştirmek ve insan değerleriyle uyum sağlamak için etkileşimde bulunan; dış bilgilerden dinamik olarak yararlanan, karmaşık görevleri ayrıştıran ve çevresel gözlemlere yanıt olarak akıl yürütme ve karar verme yetenekleri geliştiren aracılar olarak

kabul eder. Wang vd.(2023) gerçekleştirdikleri çalışmada iNLP'nin kavramsal çerçevesini ve bileşenlerini tanımlayarak, değerlendirme metodolojilerini, çeşitli uygulamalarını, etik ve güvenlik konularını ve ileriye dönük araştırma yönlerini sistematik bir şekilde incelemişlerdir.

Sağlık hizmeti tüketicileri, hasta dostu belgelerde bulunmayan bilgileri araştırma literatüründen elde etmeye çalışabilirler. Ancak tıbbi makaleleri okumak bilinmeyen terimler nedeniyle zorlu olabilmektedir. Bu soruna çözüm olarak August vd.(2023) bir prototip sistem önermişlerdir. Sistemi kullanan katılımcılar, kullanmayanlara nazaran araştırma makalelerini daha kolay okuyup anlayabilmişlerdir. Çalışmada önerilen yaklaşımın tıbbi makaleleri okumayı kolaylaştırdığı ve okuyuculara güven verdiği ortaya konmuştur.

Müller vd. (2023) tarafından gerçekleştirilen çalışmada COVID-19 ile ilgili Twitter mesajlarından oluşan geniş bir veri kümesi üzerinde eğitilen CT-BERT modeli tanıtılmaktadır. CT-BERT, sosyal medyadaki COVID-19 içeriğinde kullanılmak üzere özel olarak tasarlanmış ve sınıflandırma, soru cevaplama gibi çeşitli doğal dil işleme görevlerinde kullanım potansiyeline sahiptir. Çalışma, alana özgü önceden eğitilmiş modellerin kullanımının önemini vurgulayarak COVID-19 ile ilgili doğal dil işleme çalışmalarına katkı sağlamaktadır.

Nöropsikolojik testlerin değerlendirmesi, özelikle demans taramasında önemli gelişmeler arasındadır. Amini vd. (2023) tarafından yapılan çalışmada nöropsikolojik testlerin dijital ses kayıtlarından otomatik olarak transkribe edilmesiyle bir doğal dil

işleme modeli geliştirilmiştir. Çalışmada kullanılan yöntemin demansın farklı aşamalarını tanımlamada etkili olduğu doğrulanmıştır. Çalışmada elde edilen skorlar yöntemin performansının %92'nin üzerinde olduğunu göstermiştir. Bu yöntem, demans taramasında etkili ve maliyet açısından uygun bir yaklaşım sunmaktadır.

Yang vd. (2024) tarafından gerçekleştirilen çalışmada, konuşma sentezi teknolojisindeki yeni yaklaşımlara odaklanılmıştır. Geleneksel metinden konuşma üreten sistemlerden farklı olarak, çalışmada kullanılan yöntemde doğal dilin çeşitli konuşma stilleri kontrol edilmeye çalışılmıştır. Öncelikle, stil açıklamalarını içeren bir konuşma külliyatı oluşturulmuştur. Bu veriler üzerinde eğitilen InstructTTS adlı yeni bir model önerilmiştir. Bu model, kendi kendini denetleyen öğrenme ve metrik öğrenme avantajlarını kullanarak üç aşamalı bir eğitim prosedürüne dayanmaktadır. Deneysel sonuçlar, InstructTTS'nin çeşitli konuşma stillerini başarıyla sentezleyebildiğini ve yüksek kaliteli konuşmalar üretebildiğini göstermektedir.

Hsu vd. (2024), tarafından gerçekleştirilen çalışmada doğal dil işleme teknolojisinin ilaç geliştirme süreçlerinde nasıl etkin kullanılabileceği ele alınmıştır. Klinik farmakoloji sorularını yanıtlama, farmakokinetik parametrelerin analizi ve düzenleyici inceleme için doğal dil işlemenin üç kullanım durumunu incelenmiştir. Bu durumlar, doğal dil işlemenin büyük miktarda veriyi hızlı bir şekilde işleyerek önemli bilgilerin çıkarılmasına ve analiz edilmesine nasıl yardımcı olabileceğini göstermektedir. Gelişmiş doğal dil işlemenin entegrasyonuyla ilaç geliştirme süreçlerinde verimliliğin artabileceği ve klinik farmakoloji

süreçlerinde önemli bilgilerin daha etkili bir şekilde kullanılabileceği belirtilmektedir.

Tejaswini vd. (2024), tarafından gerçekleştirilen çalışmada araştırmacılar, sosyal medyada paylaşılan metinsel içerikleri doğal dil işleme teknikleriyle analiz ederek depresyon belirtilerini erken tespit etmeye odaklanmışlardır. Bu çalışma, "Uzun Kısa Süreli Belleğe sahip Hızlı Metin Evrişim Sinir Ağı" adı verilen yeni bir hibrit derin öğrenme modeli önerilmektedir. Bu model, daha iyi metin temsili sağlamak için hızlı metin yerleştirmeyi, genel bilgileri çıkarmak için bir evrişim sinir ağı (CNN) mimarisini ve yerel özellikleri çıkarmak için Uzun Kısa Süreli Bellek (LSTM) mimarisini birleştirir. Önerilen teknik, gerçek dünya veri setleri üzerinde test edilmiş ve depresyon tespitinde yüksek performans sergilemiştir.

Daungsupawong ve Wiwanitkit (2024) tarafında gerçekleştirilen çalışmada, üretken dil modellerinin sağlık hizmetlerinde çeşitli alanlarda kullanımları incelenmiştir. Hasta eğitimi, tıp eğitimi, ofis yönetimi, araştırma ve klinik bakım gibi alanlarda önemli bir rol oynayabilirler. Örneğin, hastalar için özelleştirilmiş bilgi notları oluşturabilir, tıp öğrencilerine sanal öğretmenlik yapabilir, idari işleri otomatikleştirebilir ve araştırma süreçlerine katkı sağlayabilirler. Ancak, bu modellerin kullanımında bazı riskler bulunmaktadır. Özellikle, güncel verilere erişimde yaşanan sıkıntılar, önyargıların yansıtılması ve güçlendirilmesi, etik meseleler ve otomasyon yanlılığı gibi konular öne çıkmaktadır. Bu riskler, doğru ve güvenilir sonuçlar elde etmek adına insan gözetimi ve doğrulamasının önemini vurgular. Gelecekteki çalışmaların, bu

sınırlamaları aşarak daha etkili ve güvenilir kullanım yolları geliştirmeye odaklanması gerekmektedir.

Deguchi vd. (2024) tarafından gerçekleştirilen çalışmada doğal dille ifade edilen yol bilgileri üzerinden harita oluşturan bir yöntem ele alınmaktadır. Çalışma metinler üzerinden topolojik harita oluşturan ve bu haritayı kullanarak yeni yolları otomatik belirleyen bir yöntem önermektedir. Önerilen yöntem, robotların kullanıcı talimatlarını daha etkili şekilde anlaması ve yeni yollar oluşturmasına katkı sağlama potansiyeli taşımaktadır.

Çetindağ vd. (2023) tarafından gerçekleştirilen çalışmada hukuki metinlerde doğal dil işleme teknolojilerinin ve özellikle adlandırılmış varlık tanıma (Named Entity Recognition-NER) tekniğinin kullanımına vurgu yapılmaktadır. NER, hukuk alanında adlandırılmış varlıkların tanınması için kritik bir rol oynamaktadır. Ancak hukuki alanda özel olarak tasarlanmış NER modelleri eksiktir. Bu soruna çözüm olarak çalışmada, Türk hukuk metinlerine özgü olarak geliştirilmiş bir NER modeli sunulmaktadır. Bu modelde, çeşitli NER mimarileri ve farklı kelime yerleştirme kombinasyonları değerlendirilmiştir. Gerçekleştirilen çalışma, Türkçe hukuki metinler için ve hukuk alanında sondan eklemeli dillere ait yapılan ilk çalışmalar arasındadır.

Jiang vd. (2023) gerçekleştirdikleri çalışmada hareket verilerini dil verileri ile birleştirerek MotionGPT adını verdikleri bir model geliştirmişlerdir. MotionGPT, insan hareketini belirli bir dil yapısına dönüştürerek hem hareket hem de metin üzerinde dil modellemeyi birleştirir. Modelin işleyişinde insan hareketi için ayrık vektör kuantizasyonu ve 3 boyutlu hareketin belirteçlere aktarılması

yöntemleri kullanılır. Deneyler, MotionGPT'nin metin odaklı hareket oluşturma, hareket altyazısı ekleme, hareket tahmini ve diğer çoklu hareket görevlerinde yüksek performans sergilediğini göstermiştir.

Zong ve Krishnamachari (2023) tarafından gerçekleştirilen çalışmada öğrencilerin matematik problemlerini çözmelerine yardımcı olmak için GPT-3 gibi güçlü dönüştürücü tabanlı modellerin kullanımı araştırılmaktadır. GPT-3 modelinin matematik sözlü problemleriyle ilgili üç farklı zorluk derecesi için kullanımını değerlendirilmiştir. Bu zorluk dereceleri problemleri sınıflandırma, denklemler çıkarma ve problemler oluşturmadır. GPT-3'ün tüm durumlarda yüksek başarı sergilediği ortaya konmuştur. Bu çalışma, yapay zekâ araçlarının öğrencilere matematik sözlü problemlerini çözmede yardımcı olabilecek bir araç olma potansiyeli taşıdığını vurgulamaktadır.

Duygu analizi, doğal dil işleme alanında önemli bir araştırma konusu haline gelmiştir. Bu teknik, e-ticaret geri bildirimleri, politika ve yönetişim gibi çeşitli uygulamalarda kullanılır. Duygu analizi için sağlam metin temsil tekniklerine ihtiyaç vardır. Metin temsil teknikleri sözlüğe dayalı ve makine öğrenimine dayalı olarak ikiye ayrılır. Ancak, her iki tekniğin de sınırlamaları vardır. Örneğin, Word2Vec gibi yöntemler, kelime duyarlılığı gibi önemli faktörleri göz ardı edebilir. Mutinda vd. (2023) tarafında gerçekleştirilen çalışmada; duygu sözlüğü, N-gramlar, BERT ve CNN'i birleştirilerek LeBERT adı verilen yeni bir duygu sınıflandırma modeli sunulmaktadır. Model, metinleri vektörleştirmek için duygu sözlüğü ve BERT'i kullanır, ardından CNN ile sınıflandırma yapılır. Farklı popüler veri setleri üzerinde test edilen LeBERT mevcut çoğu

modelden daha iyi performans göstermiştir. Bu sonuçlar, önerilen modelin duygu analizi için etkinliğini kanıtlamaktadır.

Bird vd. (2023) tarafından gerçekleştirilen çalışmada, doğal insan etkileşimini odak alan transformatör tabanlı bir sohbet robotu mimarisinin eğitimine yönelik yeni bir "yapay zekâ ile sohbet robotu etkileşimi çerçevesi" tanıtılmaktadır. Bu yüksek performanslı model, teknik olmayan kullanıcıların yapay zekâ araçlarına erişimini kolaylaştırarak insan komutlarını sosyal etkileşim seviyesinde yorumlama yeteneği sunmaktadır.

Yeni haber önerisi yöntemleri, geleneksel paradigmaların görev hedefi tutarsızlığı nedeniyle bu modellerden yeterince etkili değildir. Anlık öğrenme olarak bilinen yeni bir paradigma, doğal dil işleminde önemli başarılar sağlamaktadır. Zhang ve Wang (2023) gerçekleştirdikleri çalışmada, haber önerisinde anlık öğrenme paradigmasını uygulayan Prompt4NR çerçevesini tanıtmaktadır. Kullanıcının bir habere tıklayıp tıklamayacağını tahmin etme görevi, maske tahmin görevi olarak tasarlanmıştır. Ayrık, sürekli ve hibrit bilgi istemi şablonları geliştirilmiş ve bunlardan gelen tahminleri entegre etmek için bir birleştirici bilgi istemi kullanılmıştır. MIND veri seti üzerinde yapılan deneyler, Prompt4NR'nin etkinliğini doğrulamaktadır.

Hata raporları, yazılım kullanıcılarının karşılaştıkları sorunları geliştiricilere bildirmesi için kritik öneme sahiptir ve yazılım bakım sürecinin hızlandırılmasında önemli rol oynamaktadır. Ancak, mevcut otomatikleştirilmiş yöntemler, hata raporlarının özelliklerine ve kalitesine bağlı olarak sınırlamalar göstermektedir. Feng ve Chen (2024), büyük dil modellerinin doğal

dil anlama konusundaki başarısından ilham alarak, eğitim ve sabit kodlama gerektirmeden hata raporlarını yeniden oluşturan AdbGPT adlı hafif bir yaklaşım önermiştir. AdbGPT, hata tekrarını geliştirici gibi gerçekleştirmek için LLM'lerin insan bilgisini ve mantıksal akıl yürütme yeteneklerini kullanır. Değerlendirmeler, AdbGPT'nin hata raporlarının %81,3'ünü 253,6 saniyede yeniden üretme konusundaki etkinliğini göstermiştir.

Kanan vd. (2023) tarafından gerçekleştirilen çalışmada, çevrimiçi işletmelerin sosyal medya hesaplarından gelen büyük veri kümeleri kullanılarak müşteri memnuniyetleri tespit edilmeye çalışılmıştır. Makine öğrenmesi ve derin öğrenme teknikleriyle gerçekleştirilen analizler sonucu derin sinir ağlarının uygulamadaki başarısı vurgulanmıştır.

Saleh vd. (2023) tarafından gerçekleştirilen çalışmada, çevrimiçi nefret söyleminin otomatik tespiti için derin öğrenme tabanlı yöntemler uygulanmıştır. LSTM tabanlı yöntemler ve ayrıca, BERT modelinin de ikili sınıflandırma görevi olarak nefret söylemi tespitinde yüksek performans gösterdiği belirtilmektedir. Sonuçlar, önceden eğitilmiş modellerin performansının eğitim verilerinin boyutundan etkilendiğini ve kendi alanı için eğitilmiş modellerin etkili olduğunu ortaya koymaktadır. Bu çalışma, sosyal medya platformlarındaki nefret söyleminin tespiti için etkili yaklaşımları göstererek önemli bir katkı sağlamaktadır.

9. Gelecek Perspektifi

Doğal dil işleme alanında karşılaşılan zorluklar ve bu zorluklara yönelik çözüm önerileri, gelecekteki çalışmalar için önemli bir rehber sunmaktadır. Bu bağlamda, gelecekteki doğal dil

işleme çalışmalarına yönelik bazı öneriler ve yönlendirmeler şu şekilde sıralanabilir:

9.1. Derin Öğrenme ve Doğal Dil İşleme Entegrasyonu

Derin öğrenme tekniklerinin doğal dil işleme alanında daha geniş bir şekilde uygulanması ve bu tekniklerin dil işleme görevlerindeki etkisinin daha ayrıntılı bir şekilde incelenmesi gerekmektedir. Derin öğrenme, büyük veri setleri üzerinde eğitilen modellerin karmaşık dil kalıplarını ve ilişkilerini öğrenme kapasitesini artırarak, doğal dil işleme uygulamalarında doğruluğu ve etkinliği önemli ölçüde artırabilir. Derin öğrenme ile doğal dil işlemenin entegrasyonuna yönelik çalışmaların artırılması, gelecekteki araştırmaların odak noktalarından biri olmalıdır.

9.2. Dil Çeşitliliği ve Çok Dilli Doğal Dil İşleme

Farklı dillerde doğal dil işleme çalışmalarının artması ve dil çeşitliliğinin doğal dil işleme uygulamalarına entegrasyonu, alanın gelişimi için kritik öneme sahiptir. Az kaynaklı dillerde yapılan çalışmaların artırılması ve çok dilli doğal dil işleme modellerinin geliştirilmesi, dilsel çeşitliliği daha iyi kapsayan ve daha geniş bir kullanıcı kitlesine hizmet eden uygulamaların ortaya çıkmasına olanak tanır. Bu bağlamda, çok dilli veri setlerinin oluşturulması ve dilbilimsel farklılıkların daha iyi anlaşılması gerekmektedir.

9.3. Etik ve Gizlilik Sorunları

Doğal dil işleme çalışmalarında karşılaşılan etik ve gizlilik sorunlarına odaklanarak, bu sorunların çözümü için yeni yaklaşımlar geliştirilmelidir. Kullanıcı mahremiyeti, veri güvenliği ve kullanıcı izni gibi konular, doğal dil işleme uygulamalarında önemli etik meseleler olarak öne çıkmaktadır. Bu nedenle, veri anonimleştirme

teknikleri, veri şifreleme yöntemleri ve kullanıcı kontrolünü sağlayan sistemlerin geliştirilmesi, gelecekteki çalışmalar için temel öncelikler arasında yer almalıdır.

9.4. Doğal Dil İşleme Uygulamalarının Genişlemesi

Doğal dil işleme tekniklerinin daha geniş uygulama alanlarına yayılması ve endüstriyel uygulamalarda kullanımının artırılması önemlidir. Özellikle sağlık, finans ve güvenlik gibi kritik alanlarda doğal dil işlemenin etkisi daha da artabilir. Bu bağlamda, doğal dil işlemenin endüstriyel uygulamalara entegrasyonunu teşvik eden çalışmaların yapılması, bu teknolojinin potansiyelini en üst düzeye çıkarmaya yardımcı olacaktır.

9.5. İş birliği ve Veri Paylaşımı

Doğal dil işleme alanındaki araştırmacılar arasında iş birliğinin artırılması ve veri paylaşımının teşvik edilmesi, alandaki yenilikleri hızlandırabilir ve daha etkili çözümler sağlayabilir. Akademik ve endüstriyel iş birlikleri, bilgi ve veri paylaşımını kolaylaştırarak, Doğal dil işleme alanında daha hızlı ve kapsamlı ilerlemeler kaydedilmesini mümkün kılacaktır.

Sonuç

Doğal dil işleme alanında yapılan araştırmalar, son yıllarda önemli bir ivme kazanmış ve teknolojinin gelişimiyle birlikte çok çeşitli uygulama alanlarına yayılmıştır. Bu alandaki çalışmalar, dil işlemenin temel bileşenlerinden derin öğrenme tekniklerinin doğal dil işleme üzerindeki etkilerine kadar geniş bir yelpazeyi kapsamaktadır. Bu çalışmada, doğal dil işlmenin akademik ve endüstriyel alandaki önemi, mevcut durumu ve geleceği ele alınmıştır.

Doğal dil işlemenin genel olarak değerlendirilmesi, teknolojinin dil işleme alanındaki karmaşıklıkları çözme potansiyeline işaret etmektedir. Özellikle, derin öğrenme tekniklerinin kullanımıyla doğal dil işlemenin çeşitli görevlerde başarıyla uygulandığı gözlemlenmektedir. Metin anlama, duygu analizi, çeviri ve konuşma tanıma gibi alanlarda elde edilen başarılar, doğal dil işlemenin potansiyelini ve etkisini göstermektedir.

Doğal dil işlemenin akademik alandaki önemi giderek artmaktadır. Özellikle, dil işleme tekniklerinin eğitim, sosyal bilimler ve kültürel çalışmalar gibi alanlarda kullanımı önemli bir gelişmedir. Bu teknikler, metin analizi, içerik anlama ve öğrenme sistemleri gibi konularda kullanılarak bilimsel araştırmalara yeni boyutlar kazandırmaktadır.

Endüstriyel açıdan bakıldığında, doğal dil işlemenin kullanımı giderek yaygınlaşmaktadır. Özellikle, müşteri hizmetleri, pazarlama analizi, finansal tahminler ve tıbbi belge analizi gibi alanlarda doğal dil işlemenin etkin bir şekilde kullanıldığı görülmektedir. Bu uygulamalar, iş süreçlerini optimize etme, verimliliği artırma ve karar alma süreçlerini iyileştirme konusunda önemli katkılar sağlamaktadır.

Gelecekte, doğal dil işlemenin daha da önemli bir konuma gelmesi beklenmektedir. Bu teknolojinin gelişimiyle birlikte, daha karmaşık metin analizleri, duygusal zekâ ve daha doğal dil anlama yetenekleri gibi alanlarda büyük ilerlemeler kaydedilecektir. Bu da doğal dil işlemenin daha geniş bir uygulama alanına sahip olmasını ve insanlarla daha etkileşimli bir şekilde çalışmasını sağlayacaktır.

KAYNAKÇA

Açıkalın, U. U., Bardak, B., & Kutlu, M. (2020). Turkish Sentiment Analysis Using BERT. *2020 28th Signal Processing and Communications Applications Conference (SIU)*, 1-4. https://doi.org/10.1109/SIU49456.2020.9302492

Altıntop, M. (2023). Yapay Zekâ/Akıllı Öğrenme Teknolojileriyle Akademik Metin Yazma: Chatgpt Örneği. *Süleyman Demirel Üniversitesi Sosyal Bilimler Enstitüsü Dergisi*, *46*, Article 46.

Amini, S., Hao, B., Zhang, L., Song, M., Gupta, A., Karjadi, C., Kolachalama, V. B., Au, R., & Paschalidis, I. Ch. (2023). Automated detection of mild cognitive impairment and dementia from voice recordings: A natural language processing approach. *Alzheimer's & Dementia*, *19*(3), 946-955. https://doi.org/10.1002/alz.12721

August, T., Wang, L. L., Bragg, J., Hearst, M. A., Head, A., & Lo, K. (2023). Paper Plain: Making Medical Research Papers Approachable to Healthcare Consumers with Natural Language Processing. *ACM Transactions on Computer-Human Interaction*, *30*(5), 74:1-74:38. https://doi.org/10.1145/3589955

Ayan, E. T., Arslan, R., Zengin, M. S., Duru, H. A., Salman, S., & Bardak, B. (2021). Turkish Keyphrase Extraction from Web Pages with BERT. *2021 29th Signal Processing and Communications Applications Conference (SIU)*, 1-4. https://doi.org/10.1109/SIU53274.2021.9477842

Bayram, U. (2022). Revealing the Reflections of the Pandemic by Investigating COVID-19 Related News Articles Using Machine Learning and Network Analysis. *Bilişim Teknolojileri Dergisi*, *15*(2), Article 2. https://doi.org/10.17671/gazibtd.949599

Bird, J. J., Ekárt, A., & Faria, D. R. (2023). Chatbot Interaction with Artificial Intelligence: Human data augmentation with T5 and language transformer ensemble for text classification. *Journal of Ambient Intelligence and Humanized Computing*, *14*(4), 3129-3144. https://doi.org/10.1007/s12652-021-03439-8

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993-1022.

Brown, P. F., Cocke, J., Della Pietra, S. A., Della Pietra, V. J., Jelinek, F., Lafferty, J., Mercer, R. L., & Roossin, P. S. (1990). A statistical approach to machine translation. *Computational linguistics*, *16*(2), 79-85.

Çetindağ, C., Yazıcıoğlu, B., & Koç, A. (2023). Named-entity recognition in Turkish legal texts. *Natural Language Engineering*, *29*(3), 615-642. https://doi.org/10.1017/S1351324922000304

Daungsupawong, H., & Wiwanitkit, V. (2024). Natural Language Processing in Vascular Surgery. *EJVES Vascular Forum*, *61*, 2. https://doi.org/10.1016/j.ejvsvf.2023.10.004

Deguchi, H., Shibata, K., & Taguchi, S. (2024). *Language to Map: Topological map generation from natural language path*

instructions (arXiv:2403.10008). arXiv. https://doi.org/10.48550/arXiv.2403.10008

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Feng, S., & Chen, C. (2024). Prompting Is All You Need: Automated Android Bug Replay with Large Language Models. *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, 1-13. https://doi.org/10.1145/3597503.3608137

Hark, C. (2022). Sahte Haber Tespiti için Derin Bağlamsal Kelime Gömülmeleri ve Sinirsel Ağların Performans Değerlendirmesi. *Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 34*(2), 733-742.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.

Hsu, J. C., Wu, M., Kim, C., Vora, B., Lien, Y. T. (Kayla), Jindal, A., Yoshida, K., Kawakatsu, S., Gore, J., Jin, J. Y., Lu, C., Chen, B., & Wu, B. (2024). Applications of Advanced Natural Language Processing for Clinical Pharmacology. *Clinical Pharmacology & Therapeutics*, *115*(4), 786-794. https://doi.org/10.1002/cpt.3161

Hutchins, W. J. (2004). The Georgetown-IBM experiment demonstrated in January 1954. *Conference of the Association for Machine Translation in the Americas*, 102-114.

İşeri, İ., Aydın, Ö., & Tutuk, K. (2021). Müşteri Hizmetleri Yönetiminde Yapay Zeka Temelli Chatbot Geliştirilmesi. *Avrupa Bilim ve Teknoloji Dergisi*, *29*, Article 29. https://doi.org/10.31590/ejosat.1025380

Jiang, B., Chen, X., Liu, W., Yu, J., Yu, G., & Chen, T. (2023). MotionGPT: Human Motion as a Foreign Language. *Advances in Neural Information Processing Systems*, *36*, 20067-20079.

Kanan, T., Mughaid, A., Al-Shalabi, R., Al-Ayyoub, M., Elbes, M., & Sadaqa, O. (2023). Business intelligence using deep learning techniques for social media contents. *Cluster Computing*, *26*(2), 1285-1296. https://doi.org/10.1007/s10586-022-03626-y

Koçyiğit, A., & Darı, A. B. (2023). Yapay Zekâ İletişiminde Chatgpt: İnsanlaşan Dijitalleşmenin Geleceği. *Stratejik ve Sosyal Araştırmalar Dergisi*, *7*(2), Article 2. https://doi.org/10.30692/sisad.1311336

Manning, C. D. (2008). Introduction to information retrieval. Syngress Publishing,.

Marcus, M., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics*, *19*(2), 313-330.

Munikar, M., Shakya, S., & Shrestha, A. (2019). Fine-grained Sentiment Classification using BERT. *2019 Artificial Intelligence for Transforming Business and Society (AITB)*, *1*, 1-5. https://doi.org/10.1109/AITB48515.2019.8947435

Mutinda, J., Mwangi, W., & Okeyo, G. (2023). Sentiment Analysis of Text Reviews Using Lexicon-Enhanced Bert Embedding (LeBERT) Model with Convolutional Neural Network. *Applied Sciences*, *13*(3), Article 3. https://doi.org/10.3390/app13031445

Müller, M., Salathé, M., & Kummervold, P. E. (2023). COVID-Twitter-BERT: A natural language processing model to analyse COVID-19 content on Twitter. *Frontiers in Artificial Intelligence*, *6*. https://doi.org/10.3389/frai.2023.1023281

Ozan, Ş., & Taşar, D. E. (2021). Auto-tagging of Short Conversational Sentences using Natural Language Processing Methods. *2021 29th Signal Processing and Communications Applications Conference (SIU)*, 1-4. https://doi.org/10.1109/SIU53274.2021.9477994

Özdil, U., Arslan, B., Taşar, D. E., Polat, G., & Ozan, Ş. (2021). Ad Text Classification with Bidirectional Encoder Representations. *2021 6th International Conference on Computer Science and Engineering (UBMK)*, 169-173. https://doi.org/10.1109/UBMK52708.2021.9558966

Öztürk, H., Değirmenci, A., Güngör, O., & Uskudarli, S. (2020). The Role of Contextual Word Embeddings in Correcting the 'de/da' Clitic Errors in Turkish. *2020 28th Signal Processing and Communications Applications Conference (SIU)*, 1-4. https://doi.org/10.1109/SIU49456.2020.9302477

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257-286.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, *1*(8), 9.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, *21*(140), 1-67.

Saleh, H., Alhothali, A., & Moria, K. (2023). Detection of Hate Speech using BERT and Hate Speech Word Embedding with Deep Model. *Applied Artificial Intelligence*, *37*(1), 2166719. https://doi.org/10.1080/08839514.2023.2166719

Sevli, O., & Kemaloğlu, N. (2021). Olağandışı Olaylar Hakkındaki Tweet'lerin Gerçek ve Gerçek Dışı Olarak Google BERT Modeli ile Sınıflandırılması. *Veri Bilimi*, *4*(1), Article 1.

Tejaswini, V., Sathya Babu, K., & Sahoo, B. (2024). Depression Detection from Social Media Text Analysis using Natural Language Processing Techniques and Hybrid Deep Learning Model. *ACM Transactions on Asian and Low-Resource Language Information Processing*, *23*(1), 4:1-4:20. https://doi.org/10.1145/3569580

Toğaçar, M., Eşidir, K. A., & Ergen, B. (2022). Yapay Zekâ Tabanlı Doğal Dil İşleme Yaklaşımını Kullanarak İnternet Ortamında Yayınlanmış Sahte Haberlerin Tespiti. *Journal of Intelligent Systems: Theory and Applications*, *5*(1), Article 1. https://doi.org/10.38016/jista.950713

Turing, A. M. (1950). Mind. *Mind*, *59*(236), 433-460.

Wang, Z., Zhang, G., Yang, K., Shi, N., Zhou, W., Hao, S., Xiong, G., Li, Y., Sim, M. Y., Chen, X., Zhu, Q., Yang, Z., Nik, A., Liu, Q., Lin, C., Wang, S., Liu, R., Chen, W., Xu, K., … Fu, J. (2023). *Interactive Natural Language Processing* (arXiv:2305.13246). arXiv. https://doi.org/10.48550/arXiv.2305.13246

Weizenbaum, J. (1966). ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, *9*(1), 36-45.

Yang, D., Liu, S., Huang, R., Weng, C., & Meng, H. (2024). InstructTTS: Modelling Expressive TTS in Discrete Latent Space With Natural Language Style Prompt. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *32*, 2913-2925. https://doi.org/10.1109/TASLP.2024.3402088

Yazğılı, E., & Baykara, M. (2022). Türkçe metinlerde makine öğrenmesi yöntemleri ile siber zorbalık tespiti. *Gümüşhane Üniversitesi Fen Bilimleri Dergisi*, *12*(2), Article 2. https://doi.org/10.17714/gumusfenbil.935448

Yiğit, S., Berşe, S., & Dirgar, E. (2023). Yapay Zekâ Destekli Dil İşleme Teknolojisi Olan ChatGPT'nin Sağlık Hizmetlerinde Kullanımı. *Eurasian Journal of Health Technology Assessment*, *7*(1), Article 1. https://doi.org/10.52148/ehta.1302000

Zhang, Z., & Wang, B. (2023). Prompt Learning for News Recommendation. *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 227-237. https://doi.org/10.1145/3539618.3591752

Zong, M., & Krishnamachari, B. (2023). Solving Math Word Problems concerning Systems of Equations with GPT-3. *Proceedings of the AAAI Conference on Artificial Intelligence, 37*(13), Article 13. https://doi.org/10.1609/aaai.v37i13.26896

BÖLÜM VI

Eeg Verilerinin Analizinde Yapay Zeka Yöntemleri, Uygulamalar ve Gelecek Yönelimleri

Hatice BİRGEALP [1]
Onur SEVLİ[2]

1. Giriş

EEG (Elektroensefalografi), sinir bilimleri ve klinik nöroloji alanlarında devrim yaratan ve beyin aktivitelerinin incelenmesinde vazgeçilmez bir araç haline gelen bir teknolojidir (Dursun, 2009). EEG teknolojisinin kökenleri, Hans Berger'in 1924'te ilk beyin dalgası kayıtlarını gerçekleştirmesine kadar uzanır ve o zamandan bu yana, teknolojik ve metodolojik yenilikler sayesinde büyük ilerlemeler kaydedilmiştir (Tülay, 2009). Beyin dalgalarının non-

[1] Yüksek Lisans Öğrencisi, Burdur Mehmet Akif Ersoy Üniversitesi, Bilgisayar ve Öğretim Teknolojileri, Burdur/Türkiye, Orcid: 0000-0002-7216-6822, hbirgealp1@gmail.com

[2] Doç. Dr, Burdur Mehmet Akif Ersoy Üniversitesi, Mühendislik-Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümü, Burdur/Türkiye, Orcid: 0000-0002-8933-8395, onursevli@mehmetakif.edu.tr

invaziv bir şekilde kaydedilmesi ve analiz edilmesi, araştırmacılara ve klinisyenlere insan beyninin karmaşık yapısını ve işleyişini daha derinlemesine anlama fırsatı sunmuştur (Düzgün, 2016). Günümüzde EEG, sadece nörolojik bozuklukların teşhis ve tedavisinde değil, aynı zamanda bilişsel bilimler, psikoloji, yapay zeka ve insan-bilgisayar etkileşimi gibi disiplinler arası alanlarda da geniş bir yelpazede uygulanmaktadır (Omar ve Tepe, 2022). Bu teknoloji epilepsi (Adeli vd., 2003), uyku bozuklukları, beyin hasarı ve diğer nörolojik hastalıkların tanısında yaygın olarak kullanılır. Ayrıca, kognitif nörobilim araştırmalarında zihinsel süreçlerin ve bilinç durumlarının incelenmesinde de önemli bir araçtır. EEG sinyallerinin özellikleri, genlik, faz, zaman ve frekans analizleri gibi parametrelerle kapsamlı bir şekilde değerlendirilmekte ve gelişen teknolojilerle birlikte daha da detaylı ve doğru analizler yapılabilmektedir (Şenkaya, 2024).

Bu çalışma, EEG'nin tarihsel gelişiminden başlayarak, günümüz teknolojik yeniliklerine, veri analiz tekniklerine ve klinik uygulamalara kadar uzanan geniş bir perspektif sunmaktadır.

2. EEG’nin Tarihçesi

Caton 1875 yılında, hayvanlar üzerinde yaptığı deneysel çalışmalarda beyinde belirli elektriksel faaliyetlerin varlığını ilk kez keşfetmiştir (Sezer, 2008). EEG'nin başlangıç noktası olarak kabul edilen ve günümüz formunun gelişimine katkı sağlayan asıl durum ise, 1920'lerde Alman psikiyatr Hans Berger'in çalışmalarına dayanır (Şekil 1). 1924 yılında Berger, insan beyninin elektriksel aktivitelerini ilk kez kaydetmeyi başararak, beynin elektriksel doğasına dair ilk doğrudan kanıtları sağladı (Tülay, 2009). Berger, alfa ritimleri (8-12 Hz) ve beta dalgalarını (13-30 Hz) tanımlayarak

beyin dalgalarının belirli zihinsel durumlarla ilişkili olduğunu ortaya koyarak bu buluşuyla nöroloji ve psikiyatri alanında devrim yaratmıştır (Yener ve Öz, 2022).

1930'larda EEG'nin klinik potansiyeli daha geniş bir şekilde keşfedilmeye başlandı. İngiliz nörofizyolog Edgar Adrian, Berger'in bulgularını destekleyerek alfa ritimlerinin uyanıklık ve dinlenme durumlarıyla ilişkili olduğunu doğruladı (Sezer, 2008). Aynı dönemde Frederic Gibbs, William Lennox ve Erna Gibbs EEG'yi epilepsi teşhisinde kullanarak epileptik nöbetlerin beyindeki elektriksel aktivite üzerindeki etkilerini gösterdi. Bununla birlikte, EEG'nin nörolojik bozuklukların tanısında kritik bir araç olarak kullanılmasının yaygınlaşmasını sağladı.

1940'larda EEG, çeşitli araştırma ve klinik alanlarda daha yaygın bir şekilde kullanılmaya başlandı. Amerikan nörofizyolog Grey Walter, delta dalgalarını (0.5-4 Hz) keşfederek bu dalgaların derin uyku sırasında daha belirgin olduğunu tespit etti (Bladin, 2005). Bu dönemde Adrian ve Matthews, beyin dalgalarının belirli frekanslarda değişen uyaranlara yanıt olarak nasıl değiştiğini araştırarak uyarılmış potansiyellerin (EP) ve olaya ilişkin potansiyellerin (ERP) temelini attılar. Bu çalışmalar, beyin dalgalarının duyusal uyaranlara tepkisini anlamada önemli bir adım oldu.

1950'lerde EEG'nin uyku araştırmalarındaki rolü büyük bir ivme kazandı. Nathaniel Kleitman ve Eugene Aserinsky, 1953 yılında REM (hızlı göz hareketleri) uykusunu keşfederek uyku sırasında beyin dalgalarının dinamik değişimlerini ortaya koyarak, REM uykusunun yoğun beyin aktivitesi ile karakterize olduğunu

belirlediler. Bu çalışma, uyku evrelerinin ve rüyaların anlaşılmasında önemli bir kriterdir. Dement ve Kleitman, uyku evrelerinin sırasını ve döngüsünü tanımlayarak uyku sırasında beyin dalgalarının nasıl değiştiğini daha ayrıntılı olarak gösterdiler.

1960'larda EEG'nin bilimsel araştırmalardaki kullanımı daha da genişledi. John Eccles, EEG'yi kullanarak beyin korteksinin elektriksel aktivitelerini inceledi ve nöral iletişimin temel mekanizmalarını aydınlattı. Aynı dönemde W. Grey Walter, EEG sinyalleriyle basit makineleri kontrol edebilen ilk EEG kontrollü cihazları geliştirdi. Bu gelişme, beyin-bilgisayar arayüzlerinin (BCI) temelini oluşturdu ve felçli hastalar için yeni iletişim yöntemlerinin gelişiminin önünü açtı.

Teknolojinin gelişimiyle 1970'lerde EEG verilerinin bilgisayarlarla analiz edilebilmesi ve zaman-frekans analiz yöntemlerinin gelişmesi, EEG sinyallerinin daha detaylı ve hassas bir şekilde incelenmesini sağladı. Fourier dönüşümü gibi teknikler, EEG sinyallerinin analizini kolaylaştırdı. Walter Freeman, EEG'nin kompleks dinamik sistemler olarak beyindeki kortikal ağları nasıl temsil ettiğini araştırdı ve beyin aktivitelerinin kaotik doğasını ortaya koydu. Bu çalışmalar, beyin fonksiyonlarının daha derinlemesine anlaşılmasına katkı sağladı.

1980 ve 1990'lı yıllarda EEG teknikleri daha da gelişti ve çeşitlendi. Stephen Kosslyn, görsel imgelerin zihinde canlandırılmasının EEG ile nasıl ölçülebileceğine dair çalışmalar yürüttü. Bu çalışmalar, zihinsel imgeleme süreçlerinin EEG sinyalleri ile nasıl ilişkilendirilebileceğini gösterdi ve bilişsel nörobilimde EEG kullanımının genişlemesine katkı sağladı. J. E.

Desmedt, uyarılmış potansiyellerin (EP) ve olaya ilişkin potansiyellerin (ERP) EEG ile nasıl kaydedilebileceğini ve analiz edilebileceğini detaylandırdı. ERP'ler, belirli bir olay veya uyaranla tetiklenen beyin aktiviteleridir ve bu aktivitelerin kaydedilip, analiz edilmesi bilişsel süreçlerin incelenmesinde önemli bir adım oldu.

2000'lerde EEG ve beyin-bilgisayar arayüzleri (BCI) alanında önemli ilerlemeler kaydedildi. Jonathan Wolpaw, EEG tabanlı BCI'ların klinik uygulamalarını araştırdı ve EEG sinyalleri ile cihazları kontrol etmenin mümkün olduğunu gösterdi. Bu gelişmeler, felçli hastalar için yeni iletişim ve kontrol araçlarının geliştirilmesine öncülük etti (Aydemir ve Kayıkçıoğlu, 2009). Örneğin Emotiv Systems, EEG sinyallerini tüketici elektroniği cihazlarıyla entegre eden ilk ticari EEG kulaklıklarını tanıttı. Bu durum, EEG teknolojisinin daha geniş kitleler tarafından erişilebilir hale gelmesini sağladı.

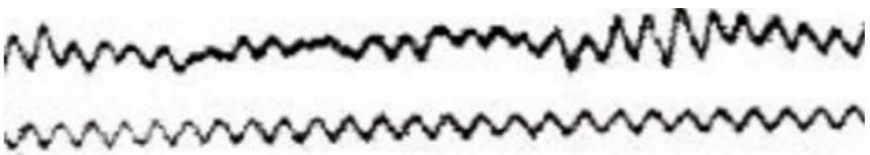
2010'larda OpenBCI gibi projeler açık kaynaklı EEG projeleri, araştırmacılar ve geliştiriciler için erişilebilir EEG donanım ve yazılım platformları sunarak, EEG araştırmalarının yenilikçi uygulamaların geliştirilmesine katkıda bulundu. Yapay zeka ve makine öğrenimi tekniklerinin EEG verilerine uygulanması, beyin dalgalarının analizinde yeni bir çağ başlattı. Bu teknolojiler, EEG sinyallerinin daha doğru ve etkili bir şekilde yorumlanmasını sağladı. Elon Musk'ın Neuralink şirketi, beyin aktivitelerini izlemek ve çeşitli nörolojik hastalıkları tedavi etmek için gelişmiş EEG teknolojilerini içeren beyin implantlarını tanıttı.

EEG'nin kronolojik tarihçesi, bu teknolojinin nasıl evrildiğini ve nöroloji, psikiyatri, uyku bilimi ve bilişsel nörobilim

gibi alanlarda nasıl devrim niteliğinde değişiklikler getirdiğini göstermektedir. EEG, başlangıcından bu yana sürekli olarak gelişmiş ve modern bilim ve teknolojinin vazgeçilmez bir aracı haline gelmiştir. Gelişen teknoloji, EEG'nin potansiyelini daha da artırmakta ve beyin işlevlerinin daha ayrıntılı bir şekilde incelenmesine olanak tanımaktadır.

3. EEG Sinyallerinin Özellikleri

Elektroensefalografi (EEG) sinyalleri, beynin elektriksel aktivitesini yansıtan karmaşık biyolojik sinyallerdir (Güntekin, 2006). Bu sinyaller, belirli özellikler ve parametreler doğrultusunda analiz edilerek beyin işlevleri ve patolojileri hakkında değerli bilgiler sağlamaktadır. Sinyallerin anlaşılması, nörofizyolojik süreçlerin ve klinik uygulamaların derinlemesine incelenmesi açısından kritiktir.



Şekil 1. Berger Tarafından Yayınlanan İlk EEG Kaydı (Tülay, 2009)

3.1. Frekans bantları ve beyin dalgaları

EEG sinyalleri, farklı frekans bantlarına ayrılır ve her bir bant belirli bir beyin durumunu veya fonksiyonunu temsil eder (Kurşunet ve Sazak, 2018). Temel frekans bantları şunlardır (Sarıkaya ve İnce, 2017; Schürmann ve Başar, 2001):

- **Delta Dalgaları (0.5-4 Hz):** Delta dalgaları, genellikle derin uyku ve bazı patolojik durumlarla ilişkilidir. Bu dalgalar, yüksek genlikli ve düşük frekanslı olup, beynin dinlenme veya iyileşme durumlarını yansıtır.
- **Teta Dalgaları (4-8 Hz):** Teta dalgaları, hafif uyku, derin meditasyon ve hafif hipnotik durumlar sırasında görülür. Ayrıca, öğrenme ve hafıza ile ilişkilendirilir.
- **Alfa Dalgaları (8-12 Hz):** Alfa dalgaları, uyanık fakat rahat durumda olan bireylerde, özellikle gözler kapalıyken ortaya çıkar. Bu dalgalar, zihinsel huzur ve rahatlama durumlarını ifade eder.
- **Beta Dalgaları (12-30 Hz):** Beta dalgaları, aktif düşünme, problem çözme ve yüksek düzeyde uyanıklık gerektiren durumlarla ilişkilidir. Bu dalgalar, düşük genlikli ve yüksek frekanslıdır.
- **Gama Dalgaları (30-100 Hz):** Gama dalgaları, yüksek düzeyde bilişsel işlemler, dikkat ve bilinçli farkındalık durumlarında görülür. Bu dalgalar, sinir ağlarının senkronizasyonu ve bilgi entegrasyonu ile ilişkilidir.

4. EEG Parametreleri

EEG sinyallerinin genliği ve fazı, beyin aktivitelerinin değerlendirilmesinde önemli parametrelerdir. EEG sinyallerinin genliği, mikrovolt (μV) cinsinden ölçülür ve beyin dalgalarının

gücünü ifade eder. Yüksek genlikli dalgalar, daha senkronize nöronal aktiviteyi temsil ederken, düşük genlikli dalgalar daha az senkronize aktiviteyi yansıtır. Faz kavramı ise sinyalin zaman içerisindeki pozisyonunu ifade eder ve nöronal osilasyonların senkronizasyonunu incelemede kullanılır. Faz analizi, farklı beyin bölgelerindeki nöronal aktivitenin nasıl etkileştiğini anlamak için kritiktir (Tuncer ve Bolat, 2022).

4.1. Zaman ve frekans analizi

EEG sinyallerinin zaman ve frekans domainlerinde analizi, beyin aktivitelerinin detaylı incelenmesini sağlar (Dursun, 2019). Zaman domaini analiziyle EEG sinyallerinin ham verileri, zaman domaininde incelenerek olayla ilişkili potansiyeller (ERP) ve zamana bağlı değişiklikler analiz edilir. ERP'ler, belirli bir uyaran veya olay sonrasında beyin aktivitelerindeki değişiklikleri yansıtır. Frekans domaini analizi EEG sinyallerinin frekans bileşenleri, Fourier dönüşümü gibi yöntemlerle ayrıştırılarak analiz edilir. Bu yöntem, belirli frekans bantlarının beyin aktiviteleriyle nasıl ilişkilendiğini ortaya koyar (Arı, 2023).

4.2. Uzaysal çözünürlük

EEG, yüksek zamansal çözünürlüğe sahip olmasına karşın, uzaysal çözünürlüğü sınırlıdır. Elektrotların kafa derisine yerleştirilmesi nedeniyle, sinyallerin kaynaklarının tam olarak lokalize edilmesi zordur. Ancak, ileri matematiksel yöntemler ve dipol kaynak modellemeleri, uzaysal çözünürlüğü artırmaya yardımcı olabilir (Alkan, 2006).

4.3. Artefaktlar ve gürültü

EEG sinyalleri, çeşitli artefakt ve gürültülere duyarlıdır. Göz hareketleri, kas aktivitesi ve elektriksel parazitler, EEG sinyallerini bozabilir. Bu nedenle, veri toplama ve analiz süreçlerinde artefaktların tespiti ve temizlenmesi önemli bir aşamadır (Dursun, 2009).

5. EEG Verilerinin Ön İşlenmesi

EEG verilerinin ön işlenmesi birçok adımdan oluşur ve gürültü filtreleme, artefakt tespiti ve giderilmesi, sinyal normalizasyonu gibi önemli işlemleri içerir (Sezer, 2008). Veriler genellikle dış ortamdan oluşabilecek çeşitli gürültü kaynaklarından etkilenir. Bu gürültüler, özellikle elektromanyetik parazitler, kas aktivitesi veya göz hareketlerinden kaynaklanabilir. Gürültüyü azaltmak ve sinyali temizlemek için yüksek ve düşük geçiş filtreleri kullanılır. Yüksek geçiş filtreleri, düşük frekanslı gürültüyü, düşük geçiş filtreleri ise yüksek frekanslı gürültüyü azaltır. Band geçiş filtreleri ise belirli frekans aralıklarındaki gürültüyü azaltmak için kullanılır ve sinyalin istenmeyen bileşenlerini filtreler.

EEG verilerindeki artefaktlar, istenmeyen veya yanlış sinyal bileşenleridir ve analiz sürecini etkileyebilir. Artefakt tespiti ve giderilmesi için yaygın kullanılan yöntemlerden biri Bağımsız Bileşen Analizi (ICA) ve Principal Component Analysis (PCA) gibi matematiksel tekniklerdir (Subasi ve Gursoy, 2010). Bu yöntemler, EEG verilerindeki farklı bileşenlerin ayrıştırılmasına ve istenmeyen bileşenlerin tanımlanmasına yardımcı olur. Bu sayede, artefaktlar sinyalden ayrıştırılarak temizlenir ve analiz doğruluğu artırılır.

EEG verilerinin normalizasyonu, veri setinin belirli bir standart veya ölçeğe getirilmesini sağlar (Sezer, 2008). Bu işlem, farklı deneyler veya katılımcılar arasındaki varyansı azaltmak ve karşılaştırmaları kolaylaştırmak için önemlidir. İki yaygın normalizasyon yöntemi Z-skoru ve min-max normalizasyonudur. Z-skoru, verilerin ortalamasını 0 ve standart sapmasını 1 yaparak bir standart dağılıma dönüştürür. Min-max normalizasyonu ise verileri belirli bir aralığa (genellikle 0 ile 1 arasına) ölçekler.

Bu ön işleme adımları, EEG verilerinin doğru şekilde analiz edilmesi ve yorumlanması için kritiktir. Gürültü filtreleme, artefakt tespiti ve giderilmesi ve sinyal normalizasyonu gibi işlemler, veri setinin güvenilirliğini artırarak nörolojik süreçlerin ve beyin aktivitelerinin daha doğru bir şekilde incelenmesini sağlar.

6. Makine Öğrenmesi Algoritmaları

Makine öğrenimi, bilgisayar sistemlerinin deneyimlerden öğrenmesini sağlayan bir yapay zekâ dalıdır. Bu sistemler, belirli bir görevi gerçekleştirmek veya belirli bir amaca ulaşmak için verileri analiz eder, desenleri tanımlar ve kararlar alır. Makine öğrenimi, geleneksel programlamadan farklı olarak, belirli bir görev için doğrudan kod yazmak yerine veri tabanlı öğrenme süreçlerini kullanır. Makine öğrenimi, genellikle üç ana kategori altında incelenir.

6.1. Denetimli öğrenme (Supervised learning)

Bu yöntemde, makine öğrenme algoritması, giriş verileriyle birlikte eşleştirilmiş çıktı verilerini kullanarak bir model oluşturarak etiketli veri kullanarak modeli eğitimektedir. Bu veri setleri, giriş (feature) ve çıkış (label) çiftlerinden oluşur. Amaç yeni,

etiketlenmemiş veriler için doğru tahminlerde bulunmaktır. Bir denetimli öğrenme algoritması, görsel içerisindeki nesneleri tanımlamak veya bir hastalığın teşhisini koymak gibi görevlerde kullanılabilir.

Denetimli öğrenme içeriğinde yer alan doğrusal regresyon (linear regression), bağımlı ve bağımsız değişkenler arasındaki doğrusal ilişkiyi modellemek için kullanılır; lojistik regresyon (logistic regression), ikili sınıflandırma problemlerinde bağımlı değişkenin bir sınıfa ait olma olasılığını tahmin eder; karar ağaçları (decision trees), veriyi dallara ayırarak sınıflandırma veya regresyon yapar; destek vektör makineleri (support vector machines), veriyi yüksek boyutlu bir uzaya haritalayarak sınıflandırma gerçekleştirir; yapay sinir ağları (artificial neural networks), insan beyninin işleyişini taklit eden, katmanlardan oluşan ağlardır (Sezer, 2008).

6.2. Denetimsiz öğrenme (Unsupervised learning)

Denetimsiz öğrenme yöntemlerinde, verilerin etiketlenmemiş olduğu durumlarda desenleri ve ilişkileri bulmak için kullanılır (Kırat ve Aydın, 2023). Bu yöntem, veri setindeki yapıları keşfetmek ve verileri gruplamak için kullanılır.

Kümeleme (Clustering) yöntemlerinden K-Ortalamalar (K-Means), veriyi k adet kümeye ayırarak her kümeyi temsil eden bir merkez belirlerken, hiyerarşik kümeleme (hierarchical clustering) veriyi hiyerarşik bir yapıda kümeleyerek alt kümeler oluşturur. Boyut indirgeme (dimensionality reduction) tekniklerinden Temel Bileşen Analizi (Principal Component Analysis - PCA), verideki değişkenliği maksimize eden yeni bileşenler oluştururken, T-SNE (t-Distributed Stochastic Neighbor Embedding), yüksek boyutlu veriyi

iki veya üç boyuta indirger ve özellikle görselleştirme için kullanılır. Bağlantı analizi (association rule learning) ise verideki öğeler arasındaki ilişkileri keşfetmek amacıyla kullanılır ve Apriori ile Eclat algoritmaları bu tür analizlerde yaygın olarak uygulanır (Cömert ve Cömert, 2017).

6.3. Pekiştirmeli Öğrenme (Reinforcement Learning)

Pekiştirmeli öğrenme, bir ajanın (agent) belirli bir ortamda hareket ederek deneyimlerden öğrenmesini sağlayan bir öğrenme paradigmasıdır (Pullu ve Karakuzu, 2023). Bu yöntemde, ajan, belirli bir hedefi veya görevi gerçekleştirmek için çevresiyle etkileşimde bulunur. Ajan, çevreden aldığı geri bildirimlere dayanarak davranışını ayarlar ve hedefe doğru ilerler. Örneğin, bir pekiştirmeli öğrenme algoritması, bir robotun belirli bir ortamda gezinmesini veya bir oyun stratejisini öğrenmesini sağlayabilir. Q-Öğrenme (Q-Learning), ajanın, her durum-eylem çiftinde maksimum toplam ödülü elde etmesini sağlayan bir politika öğrenir. Derin Pekiştirmeli Öğrenme (Deep Reinforcement Learning) Q-öğrenme gibi algoritmaları derin sinir ağları ile birleştirir (Deep Q-Networks - DQN) (Pullu ve Karakuzu, 2023).

7. Makine Öğrenmesi ve EEG

Teknolojinin hızla ilerlediği günümüzde, yazılımların ve yapay zekâ tekniklerinin yaygın olarak kullanılması, birçok sektörde insanların işlerini kolaylaştırmakta önemli bir rol oynamaktadır. Bu bağlamda, sağlık sektörü de makine öğrenimi tekniklerinin sıkça karşılaşılan uygulama alanlarından biridir (Sevli, 2019).

EEG (Elektroensefalogram) yöntemi, beyin elektriksel aktivitelerinin zaman içindeki dinamiklerini yakalamak için temel

bir araçtır. Bu elektriksel aktiviteler, sinir hücrelerinin membran potansiyellerinden kaynaklanan ve nöral iletişim süreçlerinin bir yansımasıdır. Membran potansiyeli, bir hücrenin hücre zarı boyunca oluşan elektriksel potansiyeldir. EEG'nin amacı, bu elektriksel sinyalleri kaydederek, beyin fonksiyonlarının anlaşılmasına ve farklı zihinsel süreçlerin, duygusal durumların veya patolojik durumların bu aktiviteler üzerindeki etkilerinin incelenmesine olanak tanımaktır (Çağlıyan ve Köse, 2021).

Makine öğrenimi ise bilgisayar sistemlerinin veri analizinde kullanılan bir yaklaşımdır ve algoritmalar, büyük veri kümelerindeki desenleri tanımlayarak öğrenirler. EEG verileri, genellikle karmaşık ve yüksek boyutlu olup, bu verileri analiz etmek ve anlamlı bilgiler çıkarmak için makine öğrenimi yöntemleri büyük bir potansiyel sunar (Şen ve Peker, 2012). Özellikle, EEG'nin klinik ve araştırma bağlamlarında kullanımı, epilepsi teşhisi, uyku bozuklukları, nöropsikiyatrik hastalıkların değerlendirilmesi ve bilişsel süreçlerin araştırılması gibi alanlarda önemli bir rol oynamaktadır (Çağlıyan ve Köse, 2021).

7.1. Destek vektör makineleri (DVM)

Destek Vektör Makineleri (DVM / Support Vector Machines - SVM), sınıflandırma ve regresyon analizinde kullanılan güçlü ve esnek makine öğrenmesi algoritmalarıdır (Kaynar vd., 2016). 1990'larda Vladimir Vapnik ve meslektaşları tarafından geliştirilmiştir. DVM, özellikle yüksek boyutlu veri kümelerinde ve karmaşık karar sınırlarına sahip problemlerde etkili performans göstermesiyle bilinir. Bu algoritma, sınıflandırma problemlerinde daha yaygın olarak kullanılmaktadır. İki sınıfı en iyi şekilde ayıran bir hiper düzlem bulmaktır (Subasi ve Gursoy, 2010).. DVM, bu

hiper düzlemi belirlerken marjini maksimize ederek genel performansı artırmayı hedefler.

DVM, güçlü teorik temellere dayanan ve geniş uygulama alanları olan önemli bir makine öğrenmesi algoritmasıdır. Hem akademik hem de endüstriyel uygulamalarda etkili sonuçlar elde etmek için DVM'nin doğru bir şekilde kullanılması, veri ön işleme ve parametre optimizasyonuna bağlıdır. DVM, metin ve görüntü sınıflandırması (spam filtreleme, yüz tanıma), biyoinformatik (gen ifade analizi, protein sınıflandırma) ve finans (kredi riski analizi, hisse senedi fiyat tahmini) gibi çeşitli alanlarda geniş bir kullanım alanına sahiptir (Kaynar vd., 2016).

7.1.1. EEG sinyallerinin ile DVM ile analizi

EEG sinyalleri, karmaşık ve gürültülü veriler içerir. DVM, bu tür verilerin sınıflandırılmasında yaygın olarak kullanılan bir yöntemdir., EEG sinyallerinin DVM ile sınıflandırılmasının temel adımları ve uygulama alanlarında özellik seçimi, model eğitimi kriterleri göz önünde bulundurulmaktadır.

Özellik çıkartımı ve seçimi, EEG sinyallerinin etkin bir şekilde sınıflandırılması için kritik adımlardır. EEG sinyalleri ham veri olarak sınıflandırma algoritmaları için doğrudan uygun olmadığından, belirli zaman ve frekans alanı özellikleri çıkarılmalıdır. Zaman alanı özellikleri arasında ortalama, varyans ve tepe değerleri bulunurken, frekans alanı özellikleri arasında güç spektral yoğunluğu ve bant enerjileri (delta, theta, alpha, beta) yer alır (Coşkun ve İstanbullu, 2012). Özellik seçiminde ise en bilgilendirici ve ayrıştırıcı özelliklerin belirlenmesi önemlidir; bu, özelliklerin boyutunun azaltılması ve sınıflandırma performansının

artırılması amacıyla gereklidir. Ana Bileşen Analizi (PCA) ve Doğrusal Ayırıcı Analiz (LDA) gibi teknikler, özellik seçimi sürecinde yaygın olarak kullanılmaktadır (Subasi ve Gursoy, 2010).

Model eğitimi ve testi, sınıflandırma sürecinin bir diğer önemli aşamasıdır. SVM modeli, etiketlenmiş EEG verileri kullanılarak eğitilir ve bu süreçte optimal hiper düzlem ve destek vektörleri belirlenir (Subasi ve Gursoy, 2010). Eğitilen model, test veri kümesi kullanılarak değerlendirilir ve performans ölçümleri doğruluk, hassasiyet, özgüllük, F1 skoru ve ROC eğrisi gibi metriklerle yapılır.

Uygulama alanları açısından, DVM'nin EEG sinyallerinin sınıflandırılmasında çeşitli kullanımları bulunmaktadır. Örneğin, epilepsi tespitinde DVM, epileptik nöbetlerin öncesi, sırası ve sonrasındaki EEG desenlerini sınıflandırarak erken uyarı sistemleri geliştirilmesine olanak tanır. Uyku evrelerinin sınıflandırılmasında DVM, uyku evrelerini (REM, NREM, uyanıklık) ayırt ederek uyku bozukluklarının teşhis ve tedavisinde önemli bir rol oynar (Sezer, 2008). Ayrıca, EEG sinyalleri kullanılarak bireylerin duygusal durumları (mutluluk, üzüntü, stres) SVM ile sınıflandırılabilir, bu da neurofeedback sistemlerinde ve psikolojik değerlendirmelerde kullanılabilir.

7.1.2. SVM'nin zvantajları ve dezavantajları

Avantajlar ve dezavantajlar açısından DVM'nin değerlendirilmesi, modelin güçlü ve zayıf yönlerinin anlaşılması açısından önem taşımaktadır. DVM, yüksek boyutlu veri kümelerinde yüksek doğruluk oranlarına ulaşma potansiyeline sahiptir. Maksimum marjin prensibi ile modelin genelleme yeteneği

artırılmakta ve aşırı uydurma (overfitting) riski azaltılmaktadır. Ayrıca, çekirdek (kernel) fonksiyonları sayesinde DVM'nin doğrusal olarak ayrılabilir olmayan verilerde de etkili sonuçlar elde etmesi sağlanmaktadır. Bununla birlikte, büyük veri kümeleri ve yüksek boyutlu özellik uzaylarında DVM'nin eğitim sürecinin zaman alıcı ve hesaplama açısından maliyetli olduğu bilinmektedir. Uygun kernel fonksiyonunun ve parametrelerinin seçiminin, deneyime dayalı olup model performansını önemli ölçüde etkilediği görülmektedir. Ayrıca, dengeli olmayan veri kümelerinde (örneğin, az sayıda nöbet örneği) DVM'nin performansının düşebileceği ve bu durumda sınıf dengesizliği sorunlarını ele almak için stratejiler geliştirilmesi gerektiği belirtilmektedir.

DVM, EEG sinyallerinin sınıflandırılmasında güçlü ve etkili bir yöntemdir (Aydemir ve Kayıkçıoğlu, 2009). SVM, epilepsi tespiti, uyku evrelerinin sınıflandırılması ve duygu tanıma gibi çeşitli EEG uygulamalarında başarıyla kullanılmaktadır. Özellik çıkarımı ve seçimi, DVM'nin performansını artıran önemli adımlardır. Ancak, hesaplama maliyeti, çekirdek seçimi ve veri dengesi gibi sınırlamaları da göz önünde bulundurulmalıdır. Gelecekte, daha verimli ve etkili DVM tabanlı EEG analiz yöntemlerinin geliştirilmesi, bu alandaki ilerlemeleri destekleyecektir.

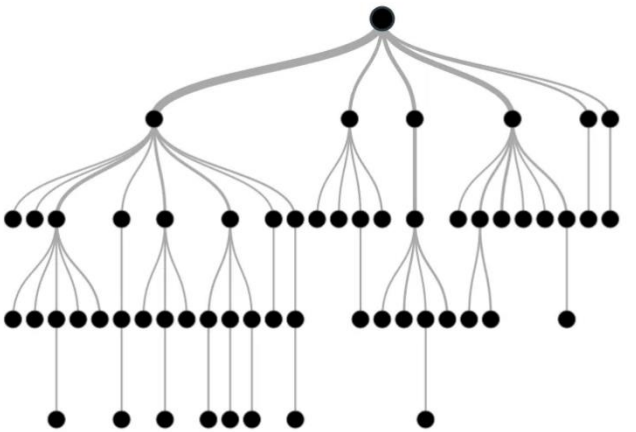
7.2. Karar ağaçları

Karar ağaçları, sınıflandırma ve regresyon problemlerinde yaygın olarak kullanılan, sezgisel ve anlaşılır bir makine öğrenme algoritmasıdır (Ülgen, 2017). Bu algoritma, kararları hiyerarşik bir yapı ile temsil eden ve her bir düğümde veriyi belirli bir özelliğe göre

bölerek çalışan bir modeldir. Karar ağaçlarının temel amacı, veri kümesini bölerek homojen alt gruplar oluşturmaktır.

7.2.1. Karar ağaçlarının yapısı ve çalışma prensibi

Kök düğüm, karar ağacının başlangıç noktası olarak tanımlanmaktadır ve tüm veri kümesi burada yer almakta olup bu düğümden itibaren veriler bölünmeye başlanmaktadır (Şekil 2).



Şekil 2. Karar Ağaçları Yapısı (Ülgen, 2017)

İç düğümler, kök düğümden çıkan ve veri kümesini özelliklerine göre bölmeye devam eden düğümler olarak belirtilmektedir. Her iç düğüm, belirli bir karar kuralı ile tanımlanmaktadır. Yaprak düğümler, karar ağacının son düğümleri olup veri kümesinin sınıflandırılmış veya tahmin edilmiş sonuçlarını içermektedir. Her yaprak düğüm, bir sınıf etiketini veya bir regresyon değerini temsil etmektedir. Dallanma işlemi, her iç düğümden çıkan dallar aracılığıyla belirli bir özelliğin olası değerlerine göre veri kümesinin iki veya daha fazla alt kümeye bölünmesi şeklinde gerçekleştirilmektedir (Ülgen, 2017).

7.2.2. Algoritma ve inşa süreci

Karar ağacının inşa süreci, belirli adımlar izlenerek gerçekleştirilmektedir. İlk olarak, kök düğümde tüm veri kümesi ile başlanmaktadır. Daha sonra, en iyi bölme kriterine göre veri kümesi uygun bir özelliğe göre bölünmektedir. Bu süreçte iç düğümler oluşturulmakta ve bu düğümler üzerinde aynı bölme işlemi tekrarlanmaktadır. Son olarak, belirli bir durdurma kriterine (örneğin, belirli bir derinlik, minimum düğüm boyutu veya saf bir düğüm) ulaşıldığında bölme işlemi durdurulmaktadır.

7.2.3. Karar ağaçlarının avantajları ve dezavantajları

Karar ağaçlarının avantajları ve dezavantajları, algoritmanın kullanımını değerlendirirken önem taşımaktadır. Anlaşılabilirlikleri sayesinde, karar ağaçları görselleştirilebilir ve kuralları açıkça belirlenebilir, bu da genellikle kullanıcılar ve diğer ilgili paydaşlar için kolay anlaşılabilirlik sağlar. Ayrıca, karar ağaçları veri setindeki tüm özellikleri kullanarak otomatik olarak en iyi bölme noktalarını belirleyebilir ve bu nedenle özellik seçimi gerektirmez. Kategorik ve sürekli verilerle çalışabilmesi ve veri ölçeklemesi gerektirmemesi, veri ön işleme aşamasında kullanıcıya avantaj sağlar.

Ancak, karar ağaçlarının bazı dezavantajları da vardır. Özellikle eğitim verisine çok iyi uyum sağlayarak aşırı uyuma (overfitting) eğilimli olabilirler, bu da genelleme yeteneğini olumsuz yönde etkileyebilir. Dengesiz veri setlerinde performans düşebilir ve azınlık sınıflarının doğru sınıflandırılması zor olabilir. Ayrıca, karar ağaçları küçük veri değişikliklerine oldukça duyarlı olabilir ve bu değişiklikler karar ağacının yapısını büyük ölçüde etkileyebilir; bu

durum kararsızlık olarak adlandırılır ve modelin güvenilirliğini azaltabilir.

7.2.4. Karar ağaçlarının EEG ile kullanımı

Karar ağaçları, EEG sinyallerinin sınıflandırılması ve analizi için etkili bir yöntem olarak kabul edilmektedir (Kaya, Ertuğrul ve Tekin, 2012). İlk aşamada, EEG sinyallerinden anlamlı özellikler çıkarılmakta ve genellikle frekans bantları, zaman-frekans bileşenleri gibi belirli özellikler üzerinde odaklanmaktadır. Daha sonra, en bilgilendirici özelliklerin seçilmesi ve veri setinin boyutunun azaltılması önem arz etmektedir, bu da modelin karmaşıklığını azaltarak performansını artırabilir. Karar ağacı modeli, eğitim veri seti üzerinde öğrenilir ve belirli karar kriterleri kullanılarak ağaç yapısı oluşturulur. Son olarak, eğitilen model test veri seti üzerinde değerlendirilir ve doğruluk, hassasiyet gibi performans metrikleri hesaplanarak modelin etkinliği değerlendirilir. EEG sinyallerinin doğası gereği karmaşık olması sebebiyle, karar ağaçlarının kullanımı, veriye özgü özelliklerin anlaşılması ve sınıflandırma problemlerinin çözülmesinde önemli bir araç olarak öne çıkmaktadır.

Karar ağaçları, çeşitli uygulama alanlarında EEG sinyallerinin analizinde etkin bir biçimde kullanılmaktadır. Özellikle epilepsi tespiti konusunda, EEG sinyallerindeki anormal desenlerin sınıflandırılması ve nöbet tahmini için önemli bir rol oynamaktadırlar. Ayrıca, uyku evrelerinin (REM, NREM, uyanıklık) belirlenmesinde de EEG sinyallerinden faydalanılarak karar ağaçları kullanılmaktadır. Bu uygulamada, farklı uyku evrelerinin karakteristik özelliklerinin tanımlanması ve sınıflandırılması mümkün olmaktadır. Bir başka önemli kullanım

alanı ise duygu tanıma çalışmalarıdır; EEG sinyalleri üzerinden bireylerin duygusal durumlarını (örneğin, mutluluk, üzüntü) belirlemek için karar ağaçları kullanılarak sınıflandırma yapılabilir. Bu yöntemler, EEG verilerinden elde edilen karmaşık bilgilerin anlamlı ve kullanılabilir formda sunulmasında önemli katkılar sağlamaktadır.

Karar ağaçları hem anlaşılabilirlik hem de esneklik açısından güçlü bir sınıflandırma ve regresyon yöntemidir. Özellikle veri bilimciler ve araştırmacılar için, karar ağaçlarının yapısı ve işleyişi, makine öğrenme modellerini daha iyi anlamak ve yorumlamak için ideal bir araçtır. EEG sinyalleri gibi karmaşık ve gürültülü verilerde bile, karar ağaçları etkili ve güvenilir sonuçlar üretebilir. Ancak, aşırı uyum gibi dezavantajları göz önünde bulundurulmalı ve gerekirse budama teknikleri veya daha gelişmiş kolektif yöntemler kullanılmalıdır.

7.2.5. Kolektif yöntemler ve karar ağaçları

Kolektif (ensemble) yöntemler, birden fazla makine öğrenme modelini bir araya getirerek daha güçlü ve güvenilir bir tahmin modeli oluşturmayı amaçlar. Karar ağaçları, kolektif yöntemlerin sıklıkla kullanılan modellerinden biridir (Yıldırım vd., 2018). Kolektif yöntemlerin temel prensibi, farklı modellerin hatalarının birbirini dengelemesi ve böylece genel performansın artırılmasıdır.

Torbalama (bagging), veri kümesinden rastgele örnekler alınarak birden fazla model eğitilip bu modellerin sonuçları birleştirilir. Bu yöntem, modelin varyansını azaltmayı ve genelleme yeteneğini artırmayı hedefler. Karar ağaçları ile torbalama uygulandığında, her bir karar ağacı farklı örnekleme setleri üzerinde

eğitilir ve tahminleri birleştirilerek son kararı oluşturur. Rastgele orman (random forest), bu yöntemin en bilinen uygulamalarından biridir; çünkü her bir ağaç rastgele seçilen özellikler ve veri örnekleriyle eğitilerek modelin genelleme yeteneği artırılır (Yıldırım vd., 2018).

Rastgele orman, torbalama yönteminin özel bir türü olarak düşünülebilir. Her ağaç için belirli bir alt küme özellik rastgele seçilir, bu da ağaçların farklı karar kuralları üretmesini sağlar. Bu yöntem, modelin performansını iyileştirirken aşırı uydurma (overfitting) riskini azaltır. Ağaçların tahminleri genellikle çoğunluk oylaması veya ortalamalar alınarak birleştirilir, bu da nihai tahminin doğruluğunu artırır.

Arttırma (boosting), zayıf öğrenicilerden güçlü bir öğrenici oluşturmayı amaçlayan bir yöntemdir. Örneğin, Adaboost (Adaptive Boosting) ve Gradient Boosting, arttırma yöntemlerinin önde gelen örnekleridir. Adaboost, yanlış sınıflandırılan örneklerin ağırlığını artırarak yeni modellerin bu örneklere odaklanmasını sağlar. Gradient Boosting ise modelin hata terimlerini minimize ederek yeni ağaçlar ekler (Yıldırım vd., 2018). Her iki yöntem de karar ağaçları ile kullanıldığında yüksek doğruluk elde edebilir ancak aşırı uyum riski de taşır.

Yığma (stacking), farklı türdeki modellerin tahminlerini bir araya getirerek ikinci bir model (meta-learner) eğitmek için kullanılan bir yöntemdir. Örneğin, karar ağaçları yığma yönteminde birinci seviye modeller olarak kullanılabilir. Farklı algoritmaların kombinasyonu genellikle genel performansı artırabilir ve karmaşıklık içindeki değişkenliği azaltabilir. Bu yöntem, özellikle

birden fazla öğrenme algoritmasının birleştirilmesi gereken karmaşık problemlerde etkilidir.

Kolektif yöntemler, karar ağaçlarının gücünü artırmak ve sınıflandırma/regresyon performansını iyileştirmek için etkili araçlardır. Torbalama, arttırma ve yığma gibi yöntemler, farklı yaklaşımlarla modelin genelleme yeteneğini artırır ve aşırı uydurma riskini azaltır. Karar ağaçlarının kolektif yöntemlerle entegrasyonu, hem teorik hem de pratik uygulamalarda geniş bir kullanım alanı bulur. Bu yöntemler, EEG sinyallerinin analizi gibi karmaşık ve gürültülü veri setlerinde daha güvenilir ve doğru tahminler elde etmek için güçlü araçlar sunar.

7.3. K-En Yakın Komşu (KNN): Basit ve etkili bir sınıflandırma algoritması

K-En Yakın Komşu (KNN) algoritması, denetimli öğrenme yöntemlerinden biridir ve hem sınıflandırma hem de regresyon görevlerinde kullanılabilir. Algoritmanın temel ilkesi, sınıflandırılacak bir örneğin sınıfını belirlemek için en yakın k sayıda komşusunun etiketlerini kullanmaktır (Kılınç vd., 2016). Algoritmanın çalışma mantığında veri kümesinin hazırlanması sürecinde, algoritmanın eğitim aşamasında her veri noktası, özellik vektörleri ve ilgili sınıf etiketleriyle birlikte depolanır. Bu, algoritmanın veri örneklerini özelliklerine göre analiz edebilmesini ve sınıflandırma veya regresyon görevlerini yerine getirilmesini sağlar. Eğitim tamamlandıktan sonra, yeni bir örneğin sınıflandırılması gerektiğinde, bu örnek ile eğitim veri kümesindeki tüm veri noktaları arasındaki mesafeler hesaplanır. Öklidyen, Manhattan ve Minkowski gibi yaygın mesafe ölçütleri kullanılarak örnek ile diğer noktalar arasındaki uzaklık belirlenir. Daha sonra,

hesaplanan mesafelere dayanarak, örneğe en yakın k adet komşu seçilir. Komşuların sınıf etiketleri incelenir ve çoğunlukla hangi sınıfa aitlerse, yeni örnek de o sınıfa atanır. Eğer regresyon yapılıyorsa, k komşunun ortalama değeri kullanılarak tahmin edilen değer belirlenir. Bu süreç, KNN algoritmasının temel işleyişini oluşturur ve özellikle sınıflandırma problemlerinde ve regresyon analizlerinde yaygın olarak kullanılır.

7.3.1. KNN algoritmasının avantajları ve dezavantajları

KNN algoritması, birçok avantajı ve dezavantajı olan etkili bir makine öğrenme yöntemidir. Algoritmanın basitliği ve kolay anlaşılabilirliği, uygulama ve yorumlama açısından kullanıcılar için önemli bir avantaj sunar. Eğitim aşamasında, KNN sadece veri kümesini depolar ve herhangi bir model eğitimi gerektirmez, bu da algoritmanın hızlı bir şekilde kullanılabilmesini sağlar. Esnekliği sayesinde hem sınıflandırma hem de regresyon problemlerini çözebilir ve çeşitli veri türlerine uygulanabilir.

Ancak, KNN algoritmasının hesaplama maliyeti yüksektir çünkü sınıflandırma yapmak için tüm veri kümesi ile örnekler arasındaki mesafelerin hesaplanması gereklidir, bu da özellikle büyük veri kümelerinde zaman alıcı olabilir. Ayrıca, algoritma tüm eğitim veri kümesini bellekte saklar, bu da bellek kullanımını artırır. Özniteliklerin ölçeklerinden etkilenmesi nedeniyle, özniteliklerin normalizasyonu veya ölçeklendirilmesi gerekebilir (Aydemir ve Kayıkçıoğlu, 2009).

KNN algoritması, tıbbi teşhis, öneri sistemleri ve görüntü tanıma gibi çeşitli uygulama alanlarında başarıyla kullanılmaktadır. Tıbbi teşhis alanında hastalıkların sınıflandırılması, öneri

sistemlerinde kullanıcı tercihlerine dayalı öneriler sunulması ve görüntü tanıma süreçlerinde etiketlenmiş görüntülerle karşılaştırma yapılması gibi örnekler, KNN algoritmasının pratik kullanımlarını göstermektedir.

7.3.2. KNN algoritması ve EEG uygulamaları

KNN ve derin öğrenme algoritmaları, EEG sinyallerinin sınıflandırılması ve analiz edilmesinde çeşitli yöntemler sunar. KNN algoritması, EEG verilerinin sınıflandırılması için basit ve etkili bir yaklaşım sağlar. EEG cihazlarıyla toplanan ham sinyaller, zaman serisi olarak kaydedilir ve ardından gürültü filtreleme ve artefakt giderme adımlarıyla ön işlemden geçirilir. Bu adımlar, düşük ve yüksek geçiş filtreleri kullanarak gürültüleri temizler ve bağımsız bileşen analizi gibi yöntemlerle artefaktları ortadan kaldırır (Hu vd., 2018).

Özellik çıkartma aşamasında, zaman alanı, frekans alanı ve zamansal-frekans analizlerinden elde edilen özellikler belirlenir. Temel istatistiksel özellikler ve FFT ile frekans bileşenleri, EEG sinyallerinden çıkarılan temel özellikler arasında yer alır. Veri kümesi hazırlanırken, bu özellikler özellik vektörleri olarak temsil edilir ve her vektöre bir sınıf etiketi atanır, örneğin epileptik nöbetin varlığı veya yokluğu gibi.

KNN algoritması, bu özellik vektörleri ve sınıf etiketleri ile eğitilir. Yeni bir EEG örneği sınıflandırılmak istendiğinde, örneğin özellik vektörü ile eğitim veri kümesindeki vektörler arasındaki mesafeler hesaplanır. Bu mesafelere göre en yakın k komşu belirlenir ve çoğunluk sınıfa göre yeni örnek sınıflandırılır. KNN algoritması, sınıflandırma sürecinde kullanılan bu mesafe tabanlı

yaklaşımıyla EEG sinyallerinin analizinde yaygın olarak kullanılmaktadır.

8. EEG ve Yapay Zeka Uygulamaları

EEG, beyindeki elektriksel aktiviteleri ölçerek çeşitli nörolojik ve psikiyatrik durumların teşhisinde ve izlenmesinde yaygın olarak kullanılan bir yöntemdir. EEG sinyallerinin manuel olarak analiz edilmesi, genellikle karmaşık ve gürültülü olmasından dolayı zor ve zaman alıcıdır. Bu noktada yapay zeka ve özellikle makine öğrenmesi ve derin öğrenme teknikleri, EEG verilerinin analizinde devrim yaratmıştır (Wang vd., 2014). AI'nın EEG sinyallerinin analizine entegrasyonu, doğruluk ve verimlilikte kayda değer artışlar sağlarken, manuel analizle karşılaştırıldığında daha hızlı ve tutarlı sonuçlar sunar.

EEG ve yapay zeka, çeşitli sağlık alanlarında önemli teşhis ve izleme uygulamaları için kullanılmaktadır. Özellikle epilepsi hastalarında nöbetlerin doğru tespiti ve tahmini, hastaların yaşam kalitesini artırmak için kritik bir öneme sahiptir (Adeli vd., 2003). Makine öğrenmesi algoritmaları, EEG sinyallerindeki epileptik nöbetleri tespit etmek ve bu süreçte derin öğrenme modelleri, nöbet öncesi anormallikleri ve nöbet sırasında oluşan desenleri tanımlamak için kullanılmaktadır. Bu yaklaşım, hasta güvenliğini artırırken aynı zamanda tedavi planlamalarının optimize edilmesine olanak sağlar (Çağlıyan ve Köse, 2021).

Uyku evrelerinin doğru analizi, uyku bozuklukları gibi sağlık sorunlarının etkilerini değerlendirmede önemli bir rol oynar. Derin öğrenme algoritmaları, EEG verilerinden uyku evrelerini (REM, NREM) otomatik olarak sınıflandırabilir. Bu yöntem, manuel

analizlere kıyasla daha hızlı ve objektif sonuçlar sağlayarak tedavi planlarının daha doğru bir şekilde belirlenmesine yardımcı olur.

Nörodejeneratif hastalıkların erken teşhisi, özellikle Alzheimer ve Parkinson gibi hastalıklar için tedavi sürecinin başarısını önemli ölçüde etkileyebilir (Signorino, 1995). Yapay zeka algoritmaları, EEG verilerinde bu hastalıklara özgü desenleri tespit edebilir ve hastalığın erken evrelerinde belirti göstermeyen ince değişiklikleri yakalayabilir. Zaman serisi analizleri ve derin öğrenme teknikleri, bu tür hastalıkların erken teşhisinde önemli bir rol oynar.

Duygusal durumların doğru tespiti, depresyon ve anksiyete gibi psikiyatrik durumların yönetiminde kritik bir öneme sahiptir. EEG sinyallerinin derin öğrenme algoritmalarıyla analizi, bireylerin duygusal durumlarını belirlemek için kullanılabilir. Bu uygulamalar, klinik değerlendirmelere katkı sağlayabilir ve kişiselleştirilmiş tedavi planlarının oluşturulmasında yardımcı olabilir.

8.1. EEG ve yapay zeka ile beyin bilgisayar arayüzleri (Brain Computer Interface-BCI)

Beyin-bilgisayar arayüzleri (BCI), EEG sinyallerini kullanarak bireylerin doğrudan cihazları beyin sinyalleriyle kontrol etmelerini sağlayan bir teknolojidir (Wang vd., 2013). Bu teknolojinin gelişimi, özellikle engelli bireyler için önemli yenilikler sunmaktadır. Motor kontrol ve rehabilitasyon için, felç veya motor bozuklukları olan bireyler için kaybedilen motor işlevlerin geri kazanılması veya alternatif kontrol yöntemleri sağlanması büyük bir avantaj sağlar. Makine öğrenmesi algoritmaları, EEG sinyallerinden niyet edilen hareketleri tanımlayarak protez cihazların veya bilgisayar kursorlerinin kontrol edilmesine olanak tanır. Bu, motor

kontrolünün iyileştirilmesine ve rehabilitasyon sürecine katkıda bulunur.

İletişim cihazları için ise, konuşma veya hareket kabiliyeti kısıtlı olan bireyler için alternatif iletişim yöntemleri geliştirilmesi büyük önem taşır. EEG tabanlı BCI sistemleri, kullanıcıların beyin dalgalarını kullanarak yazı yazmalarını veya belirli komutları seçmelerini sağlar. Bu sistemler, makine öğrenmesi algoritmaları ile daha hassas ve hızlı bir hale getirilebilir (Aydemir ve Kayıkçıoğlu, 2009).

Gelecek yönelimleri ve araştırma alanları açısından, yapay zeka ile entegre edilen EEG analizleri bireylerin nörolojik ve psikiyatrik durumlarını sürekli izleyerek kişiselleştirilmiş sağlık çözümleri sunabilir. Örneğin, bireylerin stres düzeylerinin izlenmesi ve stres yönetimi programlarının optimize edilmesi mümkün olabilir. Ayrıca, gerçek zamanlı EEG verisi işleme ve analiz, anlık geri bildirim sistemleri ve adaptif arayüzlerin geliştirilmesine imkan tanır. Bu tür sistemler, eğitim(Börekci ve Sarıtaş, 2023), oyun ve nöroterapi gibi alanlarda geniş uygulama potansiyeline sahiptir.

EEG uygulamalarında yapay zekanın kullanımı, nörolojik ve psikiyatrik bozuklukların teşhisinde, izlenmesinde ve tedavi edilmesinde devrim yaratmaktadır. Yapay zeka algoritmaları, EEG verilerinin karmaşıklığını ve büyük hacminin yönetiminde olağanüstü bir kapasite sunar. Bu, yalnızca mevcut klinik uygulamaları geliştirmekle kalmaz, aynı zamanda yeni ve yenilikçi çözümler sunarak sağlık hizmetlerinin kalitesini ve erişilebilirliğini artırır. EEG ve yapay zekanın entegrasyonu, gelecekte nörobilim ve klinik uygulamalar için geniş perspektifler sunmaktadır.

9. Gelecekteki Yönelimler ve Zorluklar

9.1. Büyük veri ve EEG

EEG verileri, yüksek frekansta kaydedilen çoklu kanallardan oluşan karmaşık sinyaller içerir. Bu durum, özellikle uzun süreli izlemelerde ve büyük ölçekli klinik çalışmalarında büyük veri kümelerinin oluşmasına neden olur. Büyük verinin işlenmesi ve analizi, önemli fırsatlar ve zorluklar barındırır (İleri vd.., 2022).

Veri yönetimi ve depolama alanında, yüksek hacimli EEG verilerinin etkin bir şekilde yönetilmesi ve depolanması için bulut tabanlı çözümler önemli bir rol oynar. Bu çözümler, esneklikleri ve ölçeklenebilirlikleri sayesinde araştırmacıların ve klinisyenlerin verilere kolayca erişimini sağlar. Bulut bilişim teknolojisi, veri saklama kapasitesini artırırken aynı zamanda veri entegrasyonunu ve standartlaştırılmış veri formatlarıyla entegrasyonunu destekler. Bu yaklaşım, farklı kaynaklardan gelen EEG verilerinin bir araya getirilmesini kolaylaştırarak veri paylaşımını ve karşılaştırmalı analizleri mümkün kılar (Filik ve Karaman, 2023).

Büyük veri analitiği, EEG verilerindeki karmaşık desenlerin ve ilişkilerin tespit edilmesini sağlayan makine öğrenmesi ve derin öğrenme algoritmalarının uygulanmasına odaklanır. Bu teknikler, nörolojik bozuklukların erken teşhisi, hastalık progresyonunun izlenmesi ve kişiselleştirilmiş tedavi yöntemlerinin geliştirilmesi gibi klinik uygulamalarda önemli rol oynar. Büyük veri setlerindeki analiz verimliliğini artırmak için gereksiz özelliklerin çıkarılması ve veri boyutunun azaltılması da kritik önem taşır. Boyut indirgeme teknikleri ve özellik seçimi algoritmaları, daha hızlı ve etkili

modelleme sağlayarak EEG verilerinden elde edilen bilgilerin daha anlamlı ve kullanışlı olmasını sağlar.

9.2. Gerçek zamanlı analiz: EEG analizi ve biyolojik geri bildirim sistemleri

Gerçek zamanlı EEG analizi, biyolojik geri bildirim sistemlerinin geliştirilmesine olanak tanıyarak anlık veri işleme ve analiz yapar. Bu tür sistemler, düşük gecikmeli algoritmalar gerektirir ve özellikle nörolojik geri bildirim uygulamalarında kritik öneme sahiptir. Derin öğrenme ve hızlı Fourier dönüşümü gibi teknikler, anlık EEG verilerini hızla işleyerek gerçek zamanlı analiz sağlar. Ayrıca taşınabilir EEG cihazları ve giyilebilir teknolojiler, bireylerin günlük yaşamları sırasında EEG verilerini toplamalarını ve analiz etmelerini mümkün kılacaktır.

Biyolojik geri bildirim sistemleri, hastaların kendi beyin aktivitelerini izleyerek kontrol etmelerini sağlayarak nöroterapi ve rehabilitasyon süreçlerinde kullanılır. Örneğin, stres yönetimi ve dikkat eksikliği hiperaktivite bozukluğu (DEHB) tedavisinde etkilidir. Ayrıca, sporcular ve profesyoneller, EEG tabanlı geri bildirim sistemlerini kullanarak performanslarını artırabilirler (Altıntaş ve Akalan, 2008). Beyin dalgalarının izlenmesi ve analizi, odaklanma ve zihinsel hazırlık süreçlerini optimize ederek bu alanlarda önemli avantajlar sunar.

9.3. Etik ve güvenlik konuları: EEG verilerinin mahremiyeti ve etik kullanımı

EEG verilerinin analizi ve kullanımı, çeşitli etik ve güvenlik konularını beraberinde getirir. Verilerin mahremiyeti ve etik kullanımı, kullanıcıların güvenliği ve haklarının korunması açısından kritik öneme sahiptir (Tunalı vd., 2016). EEG verilerinin

gizliliğini sağlamak için anonimleştirme ve şifreleme teknikleri kullanılır. Bu yöntemler, kişisel bilgilerin korunmasına ve yetkisiz erişimlerin önlenmesine yardımcı olur. EEG verilerinin paylaşımı ve erişimi konusunda gizliliğin sağlanması amacıyla sıkı kontroller ve protokoller oluşturulmalıdır. Kullanıcı onayı olmadan veri paylaşımı yapılmamalı ve verilerin kimlere ve nasıl erişileceği açıkça belirlenmelidir. Veriler toplanmadan önce, katılımcıların bilgilendirilmiş onayları alınmalıdır. Katılımcılar, verilerinin nasıl kullanılacağı, saklanacağı ve kimlerle paylaşıldığı konusunda tam olarak bilgilendirilmelidir. EEG verilerinin kullanımı, adil ve eşitlikçi olmalıdır ve hiçbir birey veya grup aleyhine ayrımcılığa neden olmamalıdır. EEG araştırmaları ve uygulamaları, düzenli olarak etik komiteler tarafından denetlenmeli ve onaylanmalıdır. Bu yaklaşımın benimsenmesiyle, araştırma süreçlerinin ve uygulamaların etik standartlara uygunluğu noktasındaki kaygılar azaltılabilir.

Büyük veri, gerçek zamanlı analiz ve etik konular, EEG uygulamalarında yapay zekanın gelecekteki yönelimleri ve karşılaşılacak zorluklar arasında önemli yer tutmaktadır. Yüksek hacimli EEG verilerinin analizi ve bulut tabanlı çözümler, veri yönetimini ve analitik kapasiteyi artırırken, gerçek zamanlı analiz ve biyolojik geri bildirim sistemleri, klinik ve performans uygulamalarında önemli gelişmeler sağlamaktadır. Ancak, bu teknolojilerin etik ve güvenlik boyutları da göz ardı edilmemelidir (İleri vd., 2022). EEG verilerinin mahremiyeti, adil ve etik kullanımı, kullanıcıların güvenliği ve haklarının korunması açısından kritik öneme sahiptir. Bu alanlardaki ilerlemeler, EEG ve

yapay zeka entegrasyonunun gelecekte daha güvenli, etkili ve yenilikçi uygulamalara yol açmasını sağlayacaktır.

10. Sonuç

Son yıllarda, EEG sinyallerinin yapay zeka ile analizi alanında önemli ilerlemeler kaydedilmiştir. Bu ilerlemeler, nörolojik ve psikiyatrik bozuklukların teşhisinden, kişiselleştirilmiş tedavi planlarının oluşturulmasına kadar geniş bir yelpazede uygulama bulmuştur.

Potansiyel uygulama alanları, yapay zeka destekli EEG analizlerinin çeşitli sağlık ve performans odaklı kullanım senaryolarını kapsamaktadır. Bu bağlamda, telemedicine ve uzaktan sağlık hizmetleri için EEG tabanlı analizler, özellikle kırsal veya ulaşımı zor bölgelerde yaşayan hastalar için önemli bir potansiyel sunmaktadır. Nörorehabilitasyon süreçlerinde ise EEG tabanlı biyolojik geri bildirim sistemleri, felç geçiren hastaların motor fonksiyonlarının iyileştirilmesi ve yeniden kazanılması için kullanılabilir. Eğitim ve performans artırma alanında (Börekci ve Sarıtaş, 2023), öğrenci ve sporcuların bilişsel yeteneklerini geliştirmek için EEG tabanlı sistemler odaklanma ve dikkat sürelerinin optimize edilmesine katkı sağlayabilir. Zihinsel sağlık ve iyilik halinin yönetimi için ise EEG ve yapay zeka kombinasyonu, bireylerin stres yönetimi, anksiyete ve depresyon gibi durumlarını izlemek ve tedavi etmek için etkili bir araç olabilir.

EEG sinyallerinin yapay zeka ile analizi, nörolojik ve psikiyatrik durumların teşhisinde ve tedavisinde önemli ilerlemeler sağlamıştır. Ancak, veri kalitesi, gerçek zamanlı işleme ve etik konular gibi mevcut sınırlar, bu alandaki çalışmaların devam

etmesini gerektirmektedir. Gelecekteki araştırmalar, gelişmiş sinyal işleme teknikleri, çapraz modalite analizler ve kişiselleştirilmiş modeller gibi alanlara odaklanarak, EEG analizlerinin doğruluğunu ve uygulama potansiyelini artırabilir. Bu ilerlemeler, sağlık hizmetlerinin kalitesini ve erişilebilirliğini artırarak, nörolojik ve psikiyatrik hastalıkların daha etkili bir şekilde yönetilmesini sağlayacaktır.

EEG, sağlık dışı alanlarda da geniş bir uygulama yelpazesi sunmaktadır. Nöropazarlama, oyun ve eğlence, eğitim, psikolojik araştırmalar, güvenlik ve sanat gibi çeşitli alanlarda EEG'nin kullanımı, teknolojinin potansiyelini ve çok yönlülüğünü ortaya koymaktadır. EEG'nin bu alanlardaki kullanımı, insan beyni hakkında daha derinlemesine bilgi edinilmesine ve çeşitli uygulamalarda kullanıcı deneyiminin iyileştirilmesine katkıda bulunmaktadır. Buna istinaden EEG'nin sağlık dışı uygulamaları, gelecekte daha da genişleyerek farklı disiplinlerde yenilikçi çözümler sunmaya devam edeceği söylenebilir.

EEG sinyallerinin analiz edilmesinde karşılaşılan zorluklardan biri, çevresel gürültü ve biyolojik artefaktlardır ve bunlar sinyalin doğru şekilde yorumlanmasını engelleyebilmektedir. Bu sorunların kısmen çözülmesi için gelişmiş filtreleme teknikleri ve sinyal işleme algoritmaları kullanılabilir. Ayrıca farklı bireylerden, farklı koşullarda toplanan EEG verilerinin çeşitliliği, genel geçer modellerin oluşturulmasını zorlaştırabilir ve bireysel farklılıklar model performansını olumsuz etkileyebilir. Bu sorunların aşılması için daha büyük ve çeşitli veri kümeleri üzerinde yapılan eğitimler ve kişiselleştirilmiş modellerin geliştirilmesi önem taşır. Son olarak, gerçek zamanlı EEG analizi, düşük gecikmeli ve

yüksek doğruluklu algoritmalar gerektirmekte ve özellikle mobil ve giyilebilir cihazlarda hesaplama gereksinimleri sınırlayıcı olabilmektedir. Bu tür sistemlerin daha uygulanabilir hale getirilmesi için hesaplama verimliliği yüksek algoritmaların ve donanımların geliştirilmesi önemlidir.

Gelecek araştırmalara yönelik öneriler EEG sinyallerinin analizi ve uygulama alanlarının genişletilmesinin hedeflenmesi şeklindedir. Bu öneriler arasında, EEG sinyallerinin gürültüden arındırılması ve artefaktların etkin bir şekilde giderilmesi için yeni yöntemlerin geliştirilmesi bulunmaktadır. Bu araştırmalar, EEG verilerinin doğruluğunu ve güvenilirliğini artırarak daha kesin analizler yapılmasına olanak tanır. Ayrıca, EEG verilerinin diğer biyomedikal sinyaller ve görüntüleme teknikleri ile birlikte analiz edilmesi için çapraz modalite analizlerinin yapılması önerilmektedir. Çoklu modalite verilerinin entegrasyonu, daha kapsamlı ve bütünsel nörolojik analizler yapılmasına imkan sağlar. Kişiselleştirilmiş yapay zeka modellerinin geliştirilmesi ise bireysel farklılıkları dikkate alarak daha özgün çözümler sunabilir. Bu modeller, bireylerin spesifik nörolojik ve psikiyatrik durumlarına daha uygun tedavi ve yönetim planları oluşturulmasına yardımcı olabilir. Son olarak, uzun süreli EEG izlemelerinde kullanılacak verimli ve kullanımı kolay yöntemlerin geliştirilmesi, nörolojik hastalıkların uzun vadeli takibi ve tedavi sonuçlarının değerlendirilmesinde önemli rol oynar. Bu bağlamda multidisipliner yaklaşımı ve önerilen çözümlerle EEG'nin gelecekteki potansiyeli daha etkili kullanılabilir.

KAYNAKÇA

Adeli, H., Zhou, Z., & Dadmehr, N. (2003). Analysis of EEG records in an epileptic patient using wavelet transform. *Journal of Neuroscience Methods*, 123, 69-87.

Alkan, A. (2006). EEG işaretlerinin ayırılmasında altuzay yöntemlerinin kullanılması. *Yaşar Üniversitesi E-Dergisi*, 1(3), 211-219. https://doi.org/10.19168/jyu.44616

Altıntaş, A., & Akalan, C. (2008). Zihinsel antrenman ve yüksek performans. *Spormetre Beden Eğitimi ve Spor Bilimleri Dergisi*, 6(1), 39-43.

Arı, B. (2023). Alkolik ve normal EEG sinyallerinin zaman-alan tanımlayıcı analizi tabanlı otomatik sınıflandırılması. *Fırat Üniversitesi Mühendislik Bilimleri Dergisi*, 35(1), 291-300. https://doi.org/10.35234/fumbd.1222526

Aydemir, Ö., & Kayıkçıoğlu, T. (2009). EEG tabanlı beyin bilgisayar arayüzleri. *Akademik Bilişim,* 9, 11-13.

Bladin, P. F. (2006). W. Grey Walter, pioneer in the electroencephalogram, robotics, cybernetics, artificial intelligence. *Journal of clinical neuroscience*, 13(2), 170-177.

Börekci, C., & Sarıtaş, T. (2023). Türkiye'de öğrenme analitikleri kullanılarak yapılan tezlerin sistematik incelenmesi. *Balıkesir Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, 25(2), 770-782.

Coşkun, M., & İstanbullu, A. (2012). EEG işaretlerinin FFT ve dalgacık dönüşümü ile analizi. XIV. Akademik Bilişim Konferansı, Uşak, Türkiye.

Cömert, Z., & Cömert, Ö. (2017). A study of technologies used in learning management systems and evaluation of new trend algorithms. *Bitlis Eren Üniversitesi Sosyal Bilimler Dergisi*, 7(1), 286-297.

Çağlıyan, B., & Köse, U. (2021). Epilepsi EEG verilerinin makine öğrenmesi teknikleriyle sınıflandırılması. *Avrupa Bilim Ve Teknoloji Dergisi(23)*, 163-172. https://doi.org/10.31590/ejosat.857507

Dursun, M. (2009). EEG sinyallerinde uyku evrelerinin zaman ve frekans domain özellikleri kullanılarak analizi. Yüksek Lisans Tezi. Selçuk Üniversitesi.

Düzgün A., (2016). Nöromarketing Alaninda Marka Algisinin Elektrofizyolojik Olarak Beyin Osilasyonlariyla Ölçümlenmesi: Eeg (Elektroensefalografi) Yöntemi Uygulamasi, Doktora Tezi, İKÜ, İstanbul

Filik, M.S. & Karaman, İ. (2023). EEG sinyalleri ile riskli işlerde çalışanların gerçek zamanlı takibi ve analizi. 10th International Erciyes Scientific Research Congress, Kayseri.

Güntekin, B. (2006). Yüz ifadesini beyin elektrofizyolojik olarak nasıl algılar? Beyin dinamiği yöntemleri ile analiz. Doktora tezi, Dokuz Eylül Üniversitesi, Sağlık Bilimleri Enstitüsü, İzmir.

Hu, B., Li, X., Sun, S., & Ratcliffe, M. (2016). Attention recognition in EEG-based affective learning research using CFS+ KNN algorithm. *IEEE/ACM transactions on computational biology and bioinformatics*, 15(1), 38-45.

İleri, S. C., Aslan, S., & Demirci, S. (2022). Büyük veri optimizasyonu için kaynak-bağlantılı harmoni arama algoritmasının performans analizi. *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi*, 15(2), 151-160

Wang, L., Zhang, X., Zhong, X., & Zhang, Y. (2013). Analysis and classification of speech imagery EEG for BCI. Biomedical signal processing and control, 8(6), 901-908.

Wang, X. W., Nie, D., & Lu, B. L. (2014). Emotional state classification from EEG data using machine learning approach. Neurocomputing, 129, 94-106.

Omar, S. Q. O., & Tepe, C. (2022). EEG sinyallerini işlemek için makine öğreniminin kullanıldığı konular üzerine bir inceleme. *Bayburt Üniversitesi Fen Bilimleri Dergisi*, 5(1), 124-137. https://doi.org/10.55117/bufbd.1099025

Kaya, Y., Ertuğrul, Ö. F., & Tekin, R. (2012). Epileptik EEG işaretlerinin sınıflandırılmasında karar kuralları ve karar ağaçlarının kullanılması. *Batman Üniversitesi Yaşam Bilimleri Dergisi*, 1(2), 403-413.

Kayalar Kurşunet, D, D. Sazak, N. (2018) Theta, alpha, SMR beyin dalgalarının müzik türleriyle olan etkileşimi: Bir nexus-10 EEG çalışması. *Online Journal Of Music Sciences*, Cilt 3 (sayı 1), 149-165. http://dx.doi.org/10.31811/ojomus.435201

Kaynar, O., Görmez, Y., Yıldız, M., & Albayrak, A. (2016, September). Makine öğrenmesi yöntemleri ile Duygu Analizi. In International Artificial Intelligence and Data Processing Symposium (IDAP'16) (Vol. 17, No. 18, pp. 17-18).

Kılınç, D., Borandağ, E., Yücalar, F., Tunalı, V., Şimşek, M., & Özçift, A. (2016). KNN algoritması ve r dili ile metin madenciliği kullanılarak bilimsel makale tasnifi. *Marmara Fen Bilimleri Dergisi*, 28(3), 89-94.

Kırat, S. S., & Aydın, İ. (2023). Açıklanabilir yapay zekâ tabanlı denetimsiz öğrenme ile ray kusur tespiti. *Demiryolu Mühendisliği*, (18), 1-13.

Pullu, H. & Karakuzu, C. (2024). Derin pekiştirmeli öğrenme ile mobil robotlarda otonom hareket. IV. Uluslararası Bilim ve İnovasyon Kongresi INSI (2023). 4, 146-154.

Schürmann, M., & Başar, E. (2001). Functional aspects of alpha oscillations in the EEG. *International Journal of Psychophysiology*, 39(2–3), 151-158.

Sezer, E. (2008). Epilepsi teşhisi için EEG sinyal analizi (Yüksek Lisans Tezi). Selçuk Üniversitesi, Fen Bilimleri Enstitüsü, Elektronik ve Bilgisayar Sistemleri Eğitimi Anabilim Dalı.

Sevli, O. (2019). Göğüs kanseri teşhisinde farklı makine öğrenmesi tekniklerinin performans karşılaştırması. *Avrupa Bilim ve Teknoloji Dergisi*, (16), 176-185.

Signorino, M., Pucci, E., Belardinelli, N., Nolfe, G., & Angeleri, F. (1995). EEG spectral analysis in vascular and Alzheimer dementia. *Electroencephalography and Clinical Neurophysiology*, 94(5), 313-325.

Subasi, A., & Gursoy, M. I. (2010). EEG signal classification using PCA, ICA, LDA and support vector machines. *Expert systems with applications*, 37(12), 8659-8666.

Şen, B. ve Peker, M., "Novel approaches for automated epileptic diagnosis using fcbf feature selection and classification Algorithms", *Turkish Journal of Electrical Engineering & Computer Sciences*, 203-9, 2012.

Şenkaya, Y. (2024). *Makine öğrenme yöntemleri ile EEG sinyalinden alzheimer tespiti.* Yüksek Lisans Tezi, Ondokuz Mayıs Üniversitesi, Samsun.

Tunalı, S. B., Gözü, Ö., & Özen, G. (2016). Pazarlama ve reklam araştırmalarında nöropazarlama üzerine yapılmış araştırmaların incelenmesi ve etik boyutunun tartisilmasi. *Kurgu*, 24(2), 1-8.

Tuncer E., Bolat E. D., "Destek vektör makineleri ile EEG sinyallerinden epileptik nöbet sınıflandırması", *Politeknik Dergisi*, 25(1): 239-249, (2022).

Tülay, E. E. (2009). Beyin elektriksel aktivitesinin ölçümü ve sinyal analizi. Yüksek Lisans Tezi. İstanbul Kültür Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı.

Ülgen, E. K. (2017). Makine Öğrenimi Bölüm-5 (Karar Ağaçları). (15.01.2024 https://medium.com/@k.ulgen90/makine-öğrenimi-bölüm-5-karar-ağaçları)

Yener, G., & Öz, D. (2022). Nörofizyolojide yenilikler ve nöropsikiyatride kullanımı. Archives of Neuropsychiatry, 59(1), 67-74.

Yıldırım, P., Birant, K. U., Radevski, V., Kut, A., & Birant, D. (2018, May). Comparative analysis of ensemble learning methods

for signal classification. In 2018 26th Signal Processing and Communications Applications Conference (SIU) (pp. 1-4). IEEE.

BÖLÜM VII

Otomotiv Sektöründe Yalın Üretim, Yalın Lojistik ve Dijitalizasyon sayesinde Verimlilik Arttırmak için AHP ve MAIRCA ile Stratejik Bir Yaklaşım

İrem DÜZDAR[1]
Deniz Feray SABA[2]

Giriş

Yalın üretim ve dijitalizasyon otomotiv sektöründe daha rekabetçi ve sürdürülebilir bir üretim ortamı yaratır. Bu yaklaşımlar, maliyetleri azaltırken kaliteyi ve müşteri memnuniyetini artırır, aynı zamanda çevresel etkileri minimize eder (Queiroz ve ark., 2024).

Bir kuruluşun başarısı için tedarikçi seçimi ve yönetimi kritik öneme sahiptir, ancak bu süreçler finansal kaynakları önemli ölçüde etkileyebilir. Tedarik zincirinin etkin bir şekilde yönetilmesi, alıcı

[1] Doç.Dr. Düzce Üniversitesi, Mühendislik Fakültesi, Endüstri Bölümü, Düzce/Türkiye, Orcid: 0000-0002-7642-8121, iremduzdar@gmail.com
[2] Düzce Üniversitesi, Mühendislik Fakültesi, Endüstri Bölümü, Düzce/Türkiye, Orcid: 0009-0005-7183-631X, denizferay06@hotmail.com

için değer yaratmanın yanı sıra uzun vadeli ilişkiler kurma ve satın alma riskini azaltma hedefini taşır. Bu karmaşık süreçte, öngörülemeyen ve kontrol edilemeyen kriterlerin çeşitliliği, çözümü zorlaştırabilir (Macit, 2023).

Buradan yola çıkılarak yalın üretim ve dijitalizasyonun otomotiv sektörüne sağladığı katkıların detaylı bir analizi yapılması amacı ile yapılan bu çalışmada literatürden ve otomotiv üretimi yapan bir işletmeden alınan görüşler doğrultusunda Çok kriterlerli karar verme yöntemleri hibrit bir şekilde kullanılmıştır. Bu çalışma, otomotiv sektöründeki organizasyonların rekabet avantajı elde etme ve sürdürülebilir başarı sağlama amacını taşıyan bir stratejik çerçeve sunmayı amaçlamaktadır. Temel odak, yalın üretim, yalın lojistik ve dijitalizasyonun entegrasyonu üzerinde durulmuştur. Bu stratejik seçeneklerin belirlenmesi ve önceliklendirilmesi için karmaşık karar verme süreçlerinde Analitik Hiyerarşi Prosesi (AHP) ve Multi-Attribute Ideal-Relative Closeness Analysis (MAIRCA) yöntemleri kullanılmaktadır.

Materyal ve Metod

Yalın Üretim kavramı resmi olarak Krafcik'in 'Yalın üretim sisteminin zaferi' makalesinde yer aldı. Yalın üretim kavramı, 1990 yılında Womack ve Jones'un Dünyayı Değiştiren Makine kitabıyla popüler hale geldi. Yalın Üretim, atıkların ortadan kaldırılmasını ele alır ve süreç akışını daha akıcı ve verimli hale getirir (Kodali ve Justii, 1988).

Dijitalleşme, yakın ve uzun vadeli gelecekte toplumu ve iş dünyasını değiştirecek en önemli trendlerden biri olarak tanımlanmaktadır. Dijitalleşmenin etkisi büyük olacaktır; birçok

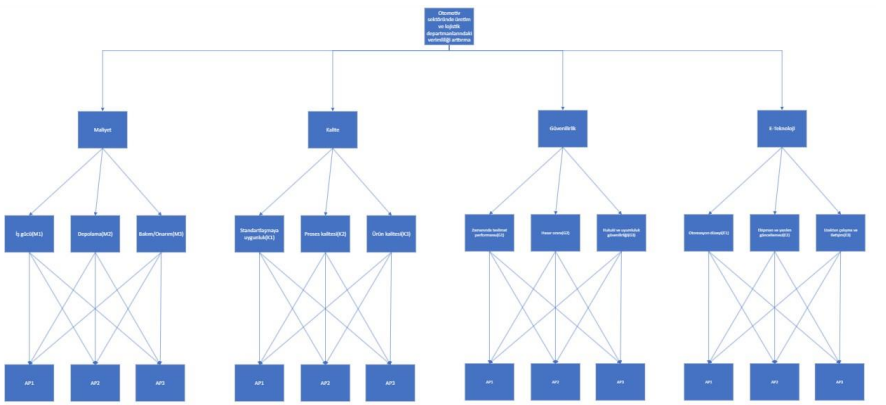
yazar tarafından sanayi devrimi ile karşılaştırılmıştır (Queiroz, 2024).

Womack ve Jones'un Toyota'nın ABD parça dağıtım sistemi üzerindeki dönüşümünü incelediği çalışma, yeni rekabet öncelikleri ve kaynak kısıtlamalarının sağlık sektörü üzerindeki baskıyı artırdığını ve bu durumun finansal, süreç, kaynaklar ve inovasyon operasyonları dahil olmak üzere çeşitli alanlarda uygulamaların sürekli iyileştirilmesine olan ilgiyi artırdığını belirtmektedir (H. Mustaffa ve Potter, 2009).

Uygulama

ANALİTİK HİYERARŞİ SÜRECİ (AHP)

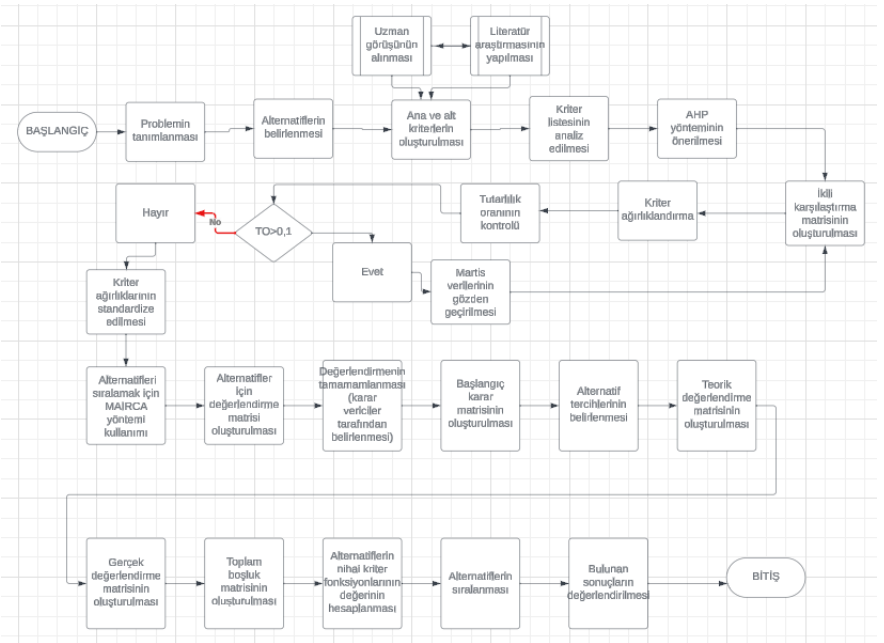
AHP, bir tür çok kriterli karar verme (ÇKVV) yöntemidir (Arbel ve Orgler, 1990). Saaty (1980), 1970'lerin başında ordu için kıt kaynak tahsisi ve planlama ihtiyaçlarına yanıt olarak AHP'yi geliştirilmiştir (Saaty, 1995).



Şekil 1. Otomotiv sektöründe yalın üretim, yalın lojistik ve dijitalizasyon stratejilerinin AHP metodolojisinin hiyerarşik yapısı

MULTIATRIBUTIVE IDEAL-REEL KARŞILAŞTIRMALI ANALİZ (MAIRCA)

Çok ölçütlü ideal-gerçek karşılaştırma analizi olarak Türkçe'ye çevrilebilecek olan MAIRCA yöntemi, tıpkı FUCOM gibi yakın bir geçmişte Pamucar, Vasin ve Lukovac (2014) tarafından geliştirilmiş olan bir ÇKKV yöntemidir (Pamučar ve ar., 2014). Yöntem, teorik (ideal) çözüm ile elde edilen (gerçek) sonuç arasındaki farkın belirlenmesi prensibine dayanmaktadır ve bu farkın en az olduğu alternatifin ideal duruma en yakın olduğu gerçeğiyle en fazla tercih edilen seçenek olması düşüncesi üzerine kurulmaktadır (Günay ve Ecer, 2020).



Şekil 2. Çalışmanın uygulama mantığı

Şekil 2'de bu çalışmada kullanılan yöntemlerin uygulama mantığının şeması görülmektedir. İlk olarak problemin

tanımlanmasıyla başlanır. Bu probleme yönelik alternatifler belirlenir. Alternatifler için alt ve ana kriterler oluşturulmalıdır, bu durumu literatür araştırmasıyla beraber uzman görüşü alınarak gerçekleştirilir. Belirlenen yöntemlerin adımları uygulanır eğer istenilen durum oluşmazsa tekrar belirlenen yöntemin ilk aşamasına dönülür. Bulunan sonuçlar içinde aynı durum gözetilir. Sonuçlar istenilenleri verirse işlem tamamlanır.

KRİTERLER

Tablo 1. Kriterler

Maliyet	Kalite	Güvenilirlik	E-teknoloji
İşgücü(M1),	Standartlaşmaya uygunluk(K1),	Zamanında teslimat performansı(G1),	Otomasyon düzeyi(E1),
Depolama(M2),	Proses kalitesi (K2),	Hasar oranı(G2),	Ekipman ve yazılım güncellemesi(E2),
Bakım/onarım(M3)	Ürün kalitesi(K3),	Hukuki ve uyumluluk güvenilirliği(G3),	Uzaktan çalışma ve iletişim(E3)

Tablo 1'de ana kriterler görülmektedir. Şekil 1'de de içeriğin gösterildiği üzere "maliyet, kalite, güvenilirlik ve e-teknoloji" ana kriterdir, alt kriterler ise iş gücü, depolama, bakım/onarım, standartlaşmaya uygunluk, proses kalitesi, ürün kalitesi, zamanında teslimat performansı, hasar oranı, hukuki ve uyumluluk güvenilirliği, otomasyon düzeyi, ekipman ve yazılım güncellemesi, uzaktan çalışma ve iletişimden oluşmaktadır. Modelde, Üretim ve Lojistik alternatifleri olmak üzere iki farklı alternatif bulunmaktadır.

Üretim Alternatifleri	1. AP1 - Yüksek Performanslı Otomasyon Modeli: • İşgücü: Orta seviyede kalifiye işgücü • Otomasyon: Dengeli otomasyon düzeyi ve planlı ekipman • Teknoloji: İleri üretim teknolojileri ve IoT entegrasyonu • Kalite ve Güvenilirlik: Yüksek kalite ve güvenilirlik • Esneklik: İyi zaman yönetimi ve uzaktan çalışma olanakları	2. AP2 - Dengeli Maliyet ve Kalite Modeli: • İşgücü: Yüksek kalifiye ve eğitimli personel • Otomasyon: Yüksek otomasyon ve robotik sistemler • Teknoloji: Temel dijitalleşme ve e-teknoloji kullanımı • Kalite ve Güvenilirlik: Orta düzeyde kalite ve güvenilirlik • Esneklik: Etkili uzaktan çalışma ve iletişim sistemleri	3. AP3 - Maliyet Odaklı Geleneksel Model: • İşgücü: Temel becerilere sahip işgücü • Otomasyon: Sınırlı otomasyon ve manuel süreçler • Teknoloji: Sınırlı dijitalleşme ve manuel e-teknoloji kullanımı • Kalite ve Güvenilirlik: Orta kalite ve güvenilirlik • Esneklik: Sınırlı esneklik, belirli endüstrilere uygun
Lojistik Alternatifleri	1. AP1 - Temel Geleneksel Lojistik Modeli: • İşgücü: Temel becerilere sahip işgücü • Otomasyon: Sınırlı otomasyon ve manuel süreçler • Teknoloji: Sınırlı dijitalleşme, temel izleme sistemleri • Güvenlik: Düşük güvenlik önlemleri, yüksek hasar oranı • Esneklik: Geleneksel iş modellerine uygun	2. AP2 - Dengeli Lojistik Modeli: • İşgücü: Orta seviyede nitelikli işgücü • Otomasyon: Dengeli otomasyon düzeyi ve planlı ekipman • Teknoloji: Temel lojistik yazılımları ve izleme sistemleri • Güvenlik: Ortalama güvenlik önlemleri, orta hasar oranı • Esneklik: Uzaktan çalışma ve iletişim sistemleri	3. AP3 - Yüksek Performanslı Otomasyonlu Lojistik Modeli: • İşgücü: Yüksek nitelikli ve eğitimli işgücü • Otomasyon: Yüksek otomasyon ve robotik sistemler • Teknoloji: İleri e-teknoloji kullanımı, gerçek zamanlı izleme • Güvenlik: Etkili güvenlik önlemleri, düşük hasar oranı • Esneklik: Tam hukuki uyumluluk, sürekli güncellemeler

Tablo 2. Üretim ve lojistik ana kriterlerinin MAIRCA sonuçları

ÜRETİM/Kara Değişkeni	**min**	**max**	**max**	**max**		
Ağırlıklar	**0,35**	**0,23**	**0,3**	**0,13**		
Alternatifler	**Maliyet**	**Kalite**	**Güvenilirlik**	**E-teknoloji**	**Q_i**	**Ranking**
AP1	0,0	0,000	0,000	0,043	0,0429	1
AP2	0,1	0,076	0,071	0,043	0,2618	2
AP3	0,1	0,061	0,099	0,000	0,2750	3

LOJİSTİK/Kara Değişkeni	**min**	**max**	**max**	**max**		
Ağırlıklar	**0,35**	**0,23**	**0,3**	**0,13**		
Alternatifler	**Maliyet**	**Kalite**	**Güvenilirlik**	**e-teknoloji**	**Qi**	**Ranking**
AP1	0,0	0,000	0,191	0,033	0,2242	3
AP2	0,1	0,056	0,000	0,033	0,1395	2
AP3	0,0	0,023	0,095	0,000	0,1186	1

Tablo 3. Üretim ve lojistik alt kriterlerinin MAIRCA sonuçları

(Q)Karar değişkeni	Min	max	max	max	max	max	max	min	max	max	max	max	Q_i	ranking
Ağırlıklar	0,20	0,64	0,16	0,11	0,40	0,48	0,42	0,28	0,31	0,43	0,47	0,10		
Alternatifler	M1	M2	M3	K1	K2	K3	G1	G2	G3	E1	E2	E3		
AP1	0,00	0,21	0,00	0,00	0,13	0,16	0,00	0,00	0,00	0,14	0,00	0,00	0,64	2
AP2	0,07	0,00	0,05	0,04	0,04	0,00	0,00	0,07	0,10	0,00	0,00	0,03	0,41	1
AP3	0,00	0,00	0,00	0,02	0,00	0,16	0,14	0,09	0,10	0,11	0,15	0,00	0,78	3

(Q)Karar değişkeni	Min	max	max	max	max	max	max	min	max	max	max	max	Q_i	ranking
Ağırlıklar	0,20	0,64	0,16	0,11	0,42	0,48	0,42	0,28	0,31	0,42	0,49	0,09		
Alternatifler	M1	M2	M3	K1	K2	K3	G1	G2	G3	E1	E2	E3		
AP1	0,00	0,00	0,01	0,00	0,00	0,16	0,14	0,09	0,10	0,10	0,16	0,00	0,76	3
AP2	0,07	0,00	0,05	0,04	0,14	0,00	0,00	0,07	0,10	0,00	0,08	0,03	0,58	2
AP3	0,00	0,21	0,00	0,00	0,00	0,16	0,00	0,00	0,00	0,14	0,00	0,02	0,53	1

Tablo 2'de üretim ve lojistik ana kriterlerinin Mairca sonuçları alternatiflerin en elverişli olanının AP1, en elverişsiz

olanının ise AP3 olduğu anlatılmaktadır. Tablo 3 ile Üretim ve lojistik alt kriterlerinin Mairca sonuçları gösterilmektedir.

MAIRCA yöntemi kullanılarak beş farklı tedarikçi alternatifi değerlendirilmiş ve yapılan değerlendirme sonucunda A2'nin en iyi alternatif olduğu belirlenmiştir. Üretim bölümü için belirlenen ana kriterlerinin belirlenen alternatiflerinin öncelik sıralaması AP1, AP2, AP3 şeklindedir. Lojistik bölümü için belirlenen ana kriterlerinin belirlenen alternatiflerinin öncelik sıralaması AP3, AP2, AP1 şeklindedir. Üretim bölümü için belirlenen alt kriterlerinin belirlenen alternatiflerinin öncelik sıralaması AP2, AP1, AP3 şeklindedir. Lojistik bölümü için belirlenen alt kriterlerinin belirlenen alternatiflerinin öncelik sıralaması AP3, AP2, AP1 şeklindedir.

Sonuç

Bu çalışma, yalın üretim prensipleri ile dijitalizasyonun entegrasyonunun, organizasyonların süreçlerini optimize ederek maliyetleri düşürmelerine ve hızlı karar almalarına imkan sağladığını vurgulamıştır. Bunun için, AHP'nin karmaşık önceliklendirme süreçlerindeki etkinliği ve MAIRCA'nın stratejik seçeneklerin değerlendirilmesindeki başarısı öne çıkarılmıştır. AHP yöntemiyle kriter ağırlıkları belirlenen üretim ve lojistik ana kriterleri ile alt kriterlerinin Mairca yöntemi ile alternatiflerinin uygunluğu sıralanmıştır.

Bu kapsamda, bir imalatçı-ihracatçı işletmede uluslararası yatırım yeri seçimini belirlemek için önerilen hibrit bir Çok Kriterli Karar Verme modeli ve MAIRCA yöntemleri kullanılarak uygulanmıştır. Gerçekleştirilen analiz sonucunda, 9 farklı kriter

üzerinden yapılan değerlendirme neticesinde, Romanya'nın uluslararası yatırım yapmak için en uygun alternatif olduğu tespit edilmiştir.

Referanslar

Arbel, A., & Orgler, Y. E. (1990). An application of the AHP to bank strategic planning: The mergers and acquisitions process. European journal of operational research, 48(1), 27-37.

Günay, F., & Ecer, F. (2020). Cash flow based financial performance of Borsa İstanbul tourism companies by Entropy-MAIRCA integrated model. Journal of multidisciplinary academic tourism, 5(1), 29-37.

Iwańkowicz, R., & Rutkowski, R. (2023). Digital twin of shipbuilding process in shipyard 4.0. Sustainability, 15(12), 9733.

Macit, N. Ş. (2023). Tedarikçi Seçimi Probleminin AHP Temelli MAIRCA Yöntemi ile Çözümü. Mehmet Akif Ersoy Üniversitesi Sosyal Bilimler Enstitüsü Dergisi, (37), 42-63.

Pamučar, D., Vasin, L., & Lukovac, L. (2014, October). Selection of railway level crossings for investing in security equipment using hybrid DEMATEL-MARICA model. In XVI international scientific-expert conference on railway, railcon (pp. 89-92).

Queiroz, G. A., Filho, A. G. A., Núñez, J. F., Santa-Eulalia, L. A., Delai, I., & Torkomian, A. L. V. (2024). Lean and Green Manufacturing in operations strategy: cases from the automotive industry. Operations Management Research, 1-25.

Saaty, T. L. (1995). Transport planning with multiple criteria: The analytic hierarchy process applications and progress review. Journal of advanced transportation, 29(1), 81-126.

Haszlinna Mustaffa, N., & Potter, A. (2009). Healthcare supply chain management in Malaysia: a case study. Supply chain management: an international journal, 14(3), 234-243.

Akdeniz, E. (2018). AHP yöntemi ile bir işletmede en iyi çalışanın seçilmesi: BT sektöründe bir organizasyon incelemesi. *Süleyman Demirel Üniversitesi Sosyal Bilimler Dergisi, 31*, 61-90.

Akkaya, H. (2013). *Tersanelerde iş sağlığı ve güvenliği tedbirleri ve işverenin sorumluluğu*, (Yüksek Lisans Tezi). Marmara Üniversitesi. İstanbul, Türkiye.

Atan, M. Ve Altan, Ş. (2020). *Uygulamalarla çok kriterli karar verme yöntemleri,* Ankara/Türkiye, Gazi Kitabevi.

Balbaş, O., Turan, E. (2019). Tersanelerde inşa edilecek gemi tipinin belirlenmesinde bulanık AHP ve bulanık TOPSIS yöntemlerinin uygulanması. *Gemi ve Deniz Teknolojisi Dergisi*, *215*.

Bozkurt, A.U. (2008). *Yenilenebilir enerji kaynaklarının enerji verimliliği açısından değerlendirilmesi,* (Yüksek Lisans Tezi). Dokuz Eylül Üniversitesi. İzmir, Türkiye.

Doğan, H., Yılankıran, N. (2015). Türkiye'nin enerji verimliliği potansiyeli ve projeksiyonu. *Gazi Üniversitesi Fen Bilimleri Dergisi, 3*(1), 375-383.

Ecer, F (2020). *Çok kriterli karar verme: Geçmişten günümüze kapsamlı bir yaklaşım*. Ankara/Türkiye, Seçkin Yayıncılık.

Erol, A., Gülsün, B., Aydın, M. (2015). Tersanelerde inşa edilecek gemi tipinin bulanık TOPSIS ve bulanık VIKOR

yöntemleri ile belirlenmesi. *Gemi ve Deniz Teknolojisi Dergisi*, *203*, 95-103.

Erol, M., Erdebilli, B. (2021). Firmaların iş sağlığı ve güvenliği performansının çok kriterli karar verme yöntemleri yardımıyla ölçülmesi. *Düzce Üniversitesi Bilim ve Teknoloji Dergisi*, *9*, 337-359.

Gonca, G. (2009). *Gemi donatımında detay dizayn ve modelleme*, (Yüksek Lisans Tezi). Yıldız Teknik Üniversitesi. İstanbul, Türkiye.

Gök, F. (2013). *Gemi inşa sanayisinde proje yönetimi ve proje yönetim planına ilişkin bir örnek*, (Yüksek Lisans Tezi). Yıldız Teknik Üniversitesi. İstanbul, Türkiye.

Görçün, Ö.F. (2020). Gemi türü seçimini etkileyen faktörlerin analitik hiyerarşi süreci (AHP) yöntemi kullanılarak değerlendirilmesi. *Manisa Celal Bayar Üniversitesi Sosyal Bilimler Dergisi*, *18*(2), 36-50.

Kaya, A.Y., Erginer, K.E. (2017). Gemilerde enerji verimliliği sağlama ve sera gazı salınımlarını azaltmaya yönelik uygulamalar: Bir odak grup çalışması. *Denizcilik Fakültesi Dergisi*, *9*(2), 212-233.

Özsoylu, A.F. (2017). Endüstri 4.0. *Çukurova Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*, *21*(1), 41-64.

Özyiğit, İ. (2006). *Gemi inşaatında planlama ve üretim kademeleri*, (Yüksek Lisans Tezi). Yıldız Teknik Üniversitesi. İstanbul, Türkiye.

Perera, L.P, Mo, B., Kristjansson L.A. (2015). Identification of optimal trim configurations to improve energy efficiency in ships. *IFAC*, *48*, 267-272.

Tari, İ. (2014). *Dünyada gemi bakım-onarım sektörü ve gemi bakım-onarımının ekonomik maliyetinin modellenmesi*, (Yüksek Lisans Tezi). İstanbul Üniversitesi. İstanbul, Türkiye.

Tezcan, Ö., Kuleyin, B. (2017). Avrupa limanlarında enerji verimliliği uygulamaları: Bir doküman analizi. *III. Liman Kongresi*.

Saaty T. (1980). *The analytic hierarchy process (AHP) for decision making*, Kobe/Japonya.

Saaty T. (2008). Making decisions in hierarchic and network systems. *International Journal of Applied Decision Sciences*, *257*.

Yiğit, K. (2018). Gemi teknolojisinde alternatif enerji sistemlerinin kullanım potansiyelinin incelenmesi. *GMO Journal of Ship and Marine Technology Journal*, *214*.

Yorulmaz, M., Öztürk, M.A. (2022). Tersanelerdeki iş kazası nedenlerinin önem düzeylerine göre belirlenmesi. *European Journal of Science and Technology*, *41*, 132-143.

Zülam, E. (2009). *Havuzlu çıkarma gemisi (LPD) dizaynı*, (Yüksek Lisans Tezi). Yıldız Teknik Üniversitesi. İstanbul, Türkiye

Wang, Y.M. (2008). On the extent analysis method for fuzzy AHP and its applications. *European Journal of Operational Research*, *186*, 735-747.

BÖLÜM VIII

Yazılım Güvenlik Araçları ve Güvenli Yazılım Geliştirme İlkeleri

Samet ÇİFCİ[1]
Buket DOĞAN[2]
Ali ÖZTÜRK[3]

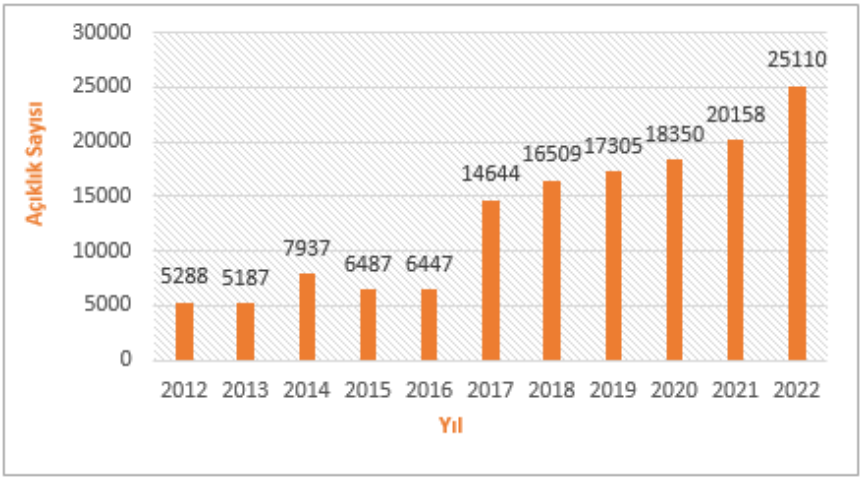
Giriş

Teknolojinin ilerlemesiyle birlikte sürekli olarak yeni yazılım uygulamalarının geliştirilip yayınlanması, birçok sektörde kullanıcılarına önemli avantajlar sunmaktadır. Bu yazılım uygulamaları, haberleşme, ulaşım, bankacılık ve ticaret gibi çeşitli alanlarda kullanıcıların günlük yaşantısını kolaylaştırmaktadır. Ancak, bu uygulamalar aracılığıyla paylaşılan kişisel bilgiler, kimlik bilgileri, telefon numaraları, bankacılık bilgileri ve parolalar gibi özel veriler, kullanıcıların mahremiyetini tehlikeye atabilmektedir. Bu bilgilerin paylaşılması, yazılım uygulamalarını kötü niyetli

[1] Marmara Üniversitesi Bilgisayar Mühendisliği, sametcifci@marun.edu.tr
[2] Doç.Dr., Marmara Üniversitesi Bilgisayar Mühendisliği, buketb@marmara.edu.tr
[3] Marmara Üniversitesi Bilgisayar Mühendisliği, ali.ozturk@marun.edu.tr

kişilerin hedefi haline getirebilmektedir. Yazılım uygulamaları üzerinde ortaya çıkabilecek güvenlik açıkları, hassas bilgilerin kötü niyetli kişiler tarafından ele geçirilmesine ve yetkisiz işlemlerin gerçekleştirilmesine neden olabilir. Günümüzde, kötü niyetli kişilerin büyük şirketlerin, devletlerin ve bireylerin özel bilgilerini ele geçirip kullanma ve sızdırma eğiliminde olduğu birçok örnek mevcuttur. Bu durumun ciddi sonuçları, gizlilik ihlalleri ve güvenlik tehditleri olarak ortaya çıkmaktadır. Ulusal Standartlar ve Teknoloji Enstitüsü (NIST) tarafından derlenen güvenlik açıkları istatistiklerine göre, yıllara göre raporlanan güvenlik açıklarında sürekli bir artış gözlemlenmektedir. Aynı rapora göre son üç yıl incelendiğinde Grafik 1'de görüldüğü gibi, 2020'de 18350, 2021'de 20158 ve 2022'de 25110 adet raporlanmış güvenlik açığı tespit edilmiştir. Bu verilere dayanarak, önümüzdeki yıllarda güvenlik açıkları sayısında bir artış öngörülmektedir (NIST, 2023).

Güvenlik açıklarının etkili bir şekilde önceden engellenmesi, yazılım geliştirme evresinin tüm aşamalarında çeşitli güvenlik süreçlerinin uygulanmasını gerektirir. Bazı yazılım geliştirme firmaları, güvenlik uygulamalarını yazılım geliştirme süreçlerine entegre ederek, yazılımlarının güvenliğini artırmak için özen göstermektedir.



Grafik 6. Yıllara göre raporlanan güvenlik açıklıkları

(NIST,2023)

Bu süreçlerin başlangıçta yazılım geliştirme süreçlerine ek bir yük getirebileceği, süreci yavaşlatabileceği ve maliyeti artırabileceği düşünülebilir. Bu düşünce, belirli bir bakış açısından doğru olabilir. Güvenlik süreçlerine dikkat etmeden daha hızlı bir yazılım geliştirme ve süreç maliyetlerini azaltma avantajları elde edilebilir. Ancak, yazılım geliştirildikten ve kullanılmaya başlandıktan sonra, güvenlik süreçleri ve pratiklerine dikkat edilmemesi, çeşitli güvenlik açıklıklarının ortaya çıkmasına neden olabilir. Bu güvenlik açıklıklarından kaynaklanan maliyet, güvenlik süreçlerini uygulamaktan kaçınmanın maliyetinden çok daha yüksek olacaktır. Güvenli yazılım geliştirme üzerine birçok araştırma gerçekleştirilmiştir. Bu araştırmalar, yazılımları daha güvenli hale getirmek ve potansiyel güvenlik sorunlarından daha az etkilenmek amacıyla çeşitli yöntemler ve prensipler geliştirmiştir. Bu makalede yazılım geliştirme yaşam döngüsünün tüm aşamalarında

kullanılabilecek yazılım araçlarını ve güvenli yazılım geliştirme için uygulanacak temel ilkeleri ele alacağız.

Güvenli olarak yazılım geliştirmenin ilkelerini ve en iyi uygulamalarını inceleyen çok sayıda çeşitli araştırma yapılmıştır. Shouki ve arkadaşları çalışmalarında güvenli tasarım prensiplerinin birçok türdeki saldırılara karşı bir çözüm olarak kullanılmasını ele almaktadır. Ancak, yazılım tasarımcılarının güvenli tasarım prensipleri kavramıyla tanışık olmadıkları veya bunları tasarım aşamasında nasıl uygulayacaklarını anlamadıkları belirtilmiştir. Bu boşluğu kapatmaya yönelik olarak, gerçek bir yazılım projesinde her bir prensibin katkısı değerlendirilmektedir. Burada tanımlanan prensiplerin çoğunun yazılımın tasarımında etkili olarak uygulanması durumunda başarı sağlandığı ifade edilmiştir. Mekanizma basitliği, güvenli varsayılanlar, en az ayrıcalık, en az ortak mekanizma, sağlam kimlik doğrulama, derin savunma ve giriş doğrulama gibi prensiplerin geliştirdikleri yazılım üzerinde büyük ölçüde başarı sağladığı, ayrıcalıkların ayrılması ve psikolojik kabul edilebilirlik gibi prensiplerin sınırlı bir ölçüde uygulanabildiği belirtilmiştir (S. A. Ebad, 2022).

Maxwell Ruggieri ve arkadaşları yazılım geliştirmedeki çeşitli güvenlik pratiklerini inceleyerek, bunları uygulamaya yönelik çözümler önermekte ve aynı zamanda uygulamanın zorluklarını vurgulamaktadır. Bu zorluklar arasında yazılım yaşam döngüsü, güvenli kod oluşturma yöntemleri, virüs türleri, iş bilgisi veya sesli iletişime bağlı olarak yazılımın farklı kullanımları ve yazılım evrimi gibi birçok faktörün bir araya getirilmesi ve ele alınması bulunmaktadır. Aynı zamanda yazılım geliştiricilerin güvenlik risklerini ve gereksinimlerini anlamaları gerektiği fakat güvenlik

konularının karmaşıklığı ve sürekli değişen tehdit çeşitliliği nedeniyle zor bir görev olacağı belirtilmiştir. Yazılımın güvenliği test edilirken dışarıdan bir bakış açısının kullanılması önerilmiştir. Ancak, güvenilir bir hacker bulmanın ve her olası hatayı bulmasının güvenilir bir yol olup olmadığı konusunda net bir husus ortaya koyamayacağını ifade etmiştir. Bu sebeple yazılım sisteminin geliştirme adımlarının her birinde, yazılım sisteminin tasarımı, geliştirilmesi ve dağıtımı için güvenlik önlemlerinin mutlaka uygulanması gerektiğini belirtmektedir (M. Ruggieri, T. T. Hsu & M. L. Ali, 2019).

May Almousa ve arkadaşları Yazılım Mühendisleri ve sistem geliştiricilerine çeşitli güvenli yazılım geliştirme süreçleri sunulmuş olmasına rağmen, yazılım açıklarını oluşturan saldırıların sürekli olarak arttığı belirtilmiştir. Bu sebeple güvenli yazılımın tasarımı ve uygulanmasında rehberliğe olan ihtiyaç vurgulanmaktadır. Yazılım mühendisleri ve sistem geliştiriciler güvenli tasarım prensiplerini uygularsa, kurumsal sistemlerin birçok türdeki saldırılara karşı güvenli hale geleceği ifade edilmiştir. Ayrıca yazılım geliştiricilerinin önemli güvenli tasarım prensipleri konusunda bilgi sahibi olup olmadığını veya bu prensipleri nasıl uygulayacaklarını anlayıp anlamadıklarını ölçmek için farklı yaş, cinsiyet, eğitim seviyeleri ve yazılım geliştirme deneyim seviyelerine sahip katılımcılara bir anket çalışması yapılmıştır. Bu anketin sonucunda güvenli tasarım prensipleri konusunda bir bilgi boşluğu olduğunu sonucuna varılmıştır. Yazılım geliştiricilerinin güvenli tasarım prensipleri kavramına yabancı oldukları veya bunları sistem tasarım aşamasında nasıl uygulayacaklarını bilmedikleri ortaya çıkmıştır (M. Almousa, M. Keshavarz & M. Anwar, 2020).

Rafıq Ahmad Khan ve arkadaşları yazılım güvenliğinin, yazılım geliştirme yaşam döngüsünün her aşamasında güvenli yazılım geliştirmenin temel bir gereksinimi haline geldiğini, yazılım sistemlerini post-geliştirme aşamalarında güvence altına almanın yetersiz olduğunu, yazılımı geliştikten sonra güvenlik testi yapmanın yalnızca zaman alıcı ve zor değil, aynı zamanda projenin karmaşıklığını ve maliyetini de arttırdığını ifade etmilerdir. Yazılım güvenliğinin sürecin erken aşamalarında değerlendirilmesi gereken kritik bir faktör olduğu ve güvenli bir yazılım sistemi oluşturmak için, güvenlik özelliklerinin uygulama geliştirme yaşam döngüsüne entegre edilmesi gerektiği belirtilmiştir. Güvenlik unsurunun, daha sonra eklenirse, bunun işlevselliği ve uygulama arayüzlerini değiştireceği için geriye dönük uyumluluğun zarar göreceği aktarılmıştır. Ayrıca güvenlik sorunlarını tanımlamak ve geliştiricilere daha güvenli yazılımlar oluşturmalarına yardımcı olabilecek 145 güvenlik riski ve 424 güvenlik uygulaması incelenmiştir (R. Khan & ark., 2022).

Trifonov ve arkadaşları güvenli yazılım geliştirme ilkelerinin önemini vurgularken programlardaki kazara oluşabilecek açıklıkları önlemek amacıyla güvenli yazılım geliştirmeye yönelik güvenli kodlama standartlarının benimsenmesi ve güvenlik gereksinimlerinin iyi tanımlanması, kullanılan teknolojiler hakkında iyi bir bilgi sahibi olunması, kütüphane kullanımının sınırlanması, yazılım bileşenlerinin işlevselliğinin minimuma indirilmesi, en az ayrıcalık ilkesine uyulması, gerçekleştirilecek tüm işlemlerin küçük bir ayrıcalık kümesiyle yürütülmesi, sadece güvenilir kanallardan gelen bilgilere güvenilmesi (giriş veya sonuç), sadece programcılar tarafından kullanılması için tasarlanmış arayüzlerin kullanılması,

kod tarama ve güvenlik testi araçlarının kullanılması gibi bir dizi prensip ve kriterin önemi üzerinde durmuştur (R. Trifonov & ark., 2020).

Smith (R. E. Smith, 1975), çok uzun süreden bu yana kullanımda olan güvenli yazılım sistemleri için bir dizi güvenlik prensibinin esaslarını ifade etmiştir. En az ayrıcalık, açık tasarım, ayrıcalıkların ayrılması ve en az yaygın mekanizma dahil olmak üzere bu ilkelerden bazıları, bilgi güvenliği uygulamalarının temelleri haline geldiğini, buna karşılık, mekanizma ekonomisi, tam aracılık ve psikolojik kabul edilebilirlik gibi diğer ilkelerin daha az önemli olduğunu belirtmiştir.

Fujdiak (R. Fujdiak, 2019) güvenli yazılım geliştirmeyi sağlamaya yönelik iki yönlü bir yaklaşım önermektedir. Önleyici yaklaşımda, olası sorunları engellemeye yardımcı olmak amaçlanmıştır. Reaktif yaklaşımda ise, hataları keşfetmeye çalışılmaktadır. Güvenlik prensiplerinin uygulanmasının, bir yazılım ürünündeki güvenlik kalitesini yükseltmeye ve bu sorunları ele almaya yardımcı olduğu belirtilmiştir. Her yazılım geliştirme aşamasının güvenlik seviyesinin kademeli olarak arttırdığı bir güvenli yazılım geliştirme yaşam döngüsü önermektedir.

Richard Linger ve arkadaşları tarafından oluşturdukları bir model ile yüksek güvenilirliğe sahip yazılım sistemleri geliştirmek maksadıyla birbirinden farklı 14 süreç ve 20 iş ürünü kullanılarak güvenli yazılım geliştirme prensiplerinin uygulanması ile kalite ve üretkenlikte büyük başarılar elde edildiği ortaya konmuştur (Richard Linger & Carmen Trammell, 1996).

Bu çalışmalar toplu olarak güvenli yazılım geliştirme ilkelerinin ve yazılım geliştirme evresinin her aşamasında yazılım güvenlik araçlarının kullanımının önemini ve bunların geliştirme süreci boyunca tutarlı bir şekilde uygulanması gerektiğini vurgulamaktadır. Yazılım geliştirme aşamalarının tamamında güvenlik prensiplerini tespit etmek ve uygun yazılımları değerlendirmek amacıyla ilgili çalışmalar incelenmiştir. İlk bölümde yazılım geliştirme aşamalarındaki kontrollerden ve kullanılabilecek yazılım güvenlik araçlarından bahsedilecektir. İkinci bölümde ise temel güvenli yazılım geliştirme ilkeleri ve sonuç bölümü yer almaktadır.

1. Yazılım Geliştirme Aşamalarına İlişkin Güvenlik Araçları

Yazılım geliştirme yaşam döngüsü aşamalarına göre bilgi sistemlerinin denetimi, incelenmesi ve değerlendirilmesine yönelik çeşitli yazılım araçlarının sınıflandırılması, siber güvenlik uzmanlarının görevlerinde saldırılara karşı korunma yollarını öğrenmelerini daha iyi hale getirmelerine yardımcı olacaktır. Güvenlikle ilgilenen yazılım araçlarının yazılım geliştirme yaşam döngüsü ile nasıl ilişkili olduğu temel modellerden biri olan şelale modeli ("Waterfall Model", 2021) üzerinde incelenmiş olup temel aşamaları ve güvenlik araçları Tablo 1'de gösterilmektedir.

1.1. Gereksinim Analizi ve Tasarım Aşamasında Uygulanacak Güvenlik Araçları

Yazılım tasarım aşamasında kullanılacak yazılım araçlarıyla birlikte bir takım önlemler almak etkin bir şekilde güvenlik sağlama noktasında çok fayda sağlayacaktır. Bu önlemler arasında şu hususlar yer almaktadır:

Her safhanın girdileri, çıktıları, kontrol metotları, iş zaman planları ve uygulama planları belirlenmeli, benzersiz kullanıcı isimleri ve şifreler kullanarak güvenli erişim sağlanmalı, kullanıcılar işlevlerine göre gruplandırılmalıdır. Bilgi sistemlerinin bağlantısı için politikalar ve prosedürler geliştirilmeli, yüksek riskli uygulamalara bağlantı sınırlamaları getirilmeli ve ağlar güvenli bir şekilde yönetilmelidir. Kullanıcılar ve destek personeli, belirlenen erişim kontrol politikalarına uygun olarak sisteme erişim sağlamalıdır, bu da yetkisiz erişimleri önlemektedir. Bu önlemler, yazılım geliştirme sürecinin tasarım aşamasında alınacak güvenlik tedbirlerini kapsar ve sistemin bütünlüğünü koruma amacını taşımaktadır(T.C. Ulaştırma ve Altyapı Bakanlığı Siber Güvenlik Koordinatörlüğü Güvenli Yazılım Geliştirme Rehberi,2018).

Tablo 1. Yazılım geliştirme yaşam döngüsü açık kaynaklı yazılım güvenlik araçları

Gereksinim Analizi /Tasarım	Geliştirme /Kodlama	Test	Bakım	Eğitim
- Orcanos **- Visure Requirements** **- Security Pins** **- Pytm** **- Jama Software** **- Cornucopia**	- SonarQube - CRSF Guard - ESLint - Attack Surface - Detector - ESAPI - HTML Sanitizer - FindSecBugs - Application Gateway - Gitguardian	- Amass - Defectdojo - OWTF - ZAP - APICheck - Mobile Audit - Nettacker - Purpleteam	- AntiSamy - BLT - O-Saft	- Juice Shop - KnowBe4 - Pygoat - Secure Coding Dojo - Snakes and Ladders - Proofpoint Security Awareness Training - Honeypot - IoTGoat - SamuraiWTF

(OWASP t.y.; Dimov, Aleksandar & Vladimir Dimitrov, 2021)

Tasarım aşamasında kullanılacak yazılım araçları kapsamında ise;

Orcanos, ekiplerin ürünlerini belgelemesi, geliştirmesi ve teslim etmesine yönelik olarak kullanılan faydalı bir uygulamadır. Kullanıcı öykülerinden kullanım senaryolarına kadar gereksinim belgelerini oluşturmak, saklamak ve düzenlemek için bazı işlevlere sahiptir. Bunun yanı sıra, yazılım geliştirme ekiplerinin işbirliği yapmasını, meydana gelen değişiklikleri takip edebilmesini ve bu değişikliklerin nasıl bir etki yaratacağını analiz etmeyi gösterebilen bir uygulamadır(AppMaster, 2023).

Visure Requirements, gereksinim yönetimi ve izlenebilirlik için kulanılabilen bir platformdur. İşbirliği, izlenebilirlik ve etki analizi için özellikler sunar. Ürün geliştirme sürecinin planlama aşamasından ürün tanıtımına gereksinimlerin, tasarımın, testin ve geri bildirimin merkezi bir konumda takip edilip yönetilmesini sağlar. (AppMaster, 2023)

Security Pins, güvenlik özelliklerine yapılan mevcut yatırımın görselleştirilmesini sağlayan bir platformdur. Bu, yazılım sistemlerinin tasarım aşamasında güvenlik profesyonellerinin güvenlik hesaba katılmamış olası boşlukları belirlemelerine ve buraya ek çaba harcamalarına odaklanmalarına yardımcı olan bir platformdur. Özellikle kimlik doğrulama veya veri güvenliği gibi alanlarda, özel bir değer veya anahtarın (genellikle bir şifre, belirli bir doğrulama kodu veya benzeri) kullanıcı veya sistem tarafından doğrulanması için kullanılan bir yöntemi ifade eder. Pinler aracılığıyla yetkilendirme süreçlerinde veya veri erişim kontrollerinde kullanılabilir ve genellikle sistemin güvenliğini artırmak için kullanılır. Örneğin, bir API'ye erişirken güvenlik nedenleriyle kullanıcıya verilen özel bir anahtar, bir tür security pin olarak kabul edilebilir. Bu şekilde yalnızca doğru anahtara sahip

kullanıcıların yetkilendirilmesini sağlar ve böylece sistem güvenliği sağlanır.

Pytm, ise önceden tanımlanmış öğeler aracılığıyla bir sistem tasarımını tanımlamak için kullanılan bir çerçevedir. Bu temelde, belirli UML diyagramları oluşturabilir ve bu şekilde sisteme potansiyel tehditleri gösterebilir. Bu araç, Python sözdizimini kullanır ve bu nedenle geliştirme aşamasında da kullanılabilir (OWASP t.y.; Dimov, Aleksandar & Vladimir Dimitrov, 2021).

Jama Software, gereksinim analizi yönetimi ve ürün geliştirmede kullanılan bir uygulamadır. Kullanıcılar, gereksinim belgeleri gibi kullanıcı öyküleri ve kullanım senaryolarını oluşturabilir, saklayabilir ve yönetebilirler. Aynı zamanda farklı endüstrilerdeki ürün geliştirme ekiplerinin iş akışlarını optimize etmelerine ve ürünlerini etkili bir şekilde yönetmelerine olanak sağlar (AppMaster, 2023).

Cornucopia, yazılım mühendislerine güvenlik gereksinimlerini daha iyi tanımlamalarına yardımcı olmayı amaçlayan bir kart oyunudur. Cornucopia'nın kullanımı, bir dizi senaryo ve durum üzerinden gerçekleştirilir. Her senaryo, potansiyel bir güvenlik açığı veya riski temsil eder ve geliştirme ekibi veya güvenlik uzmanları, bu senaryoları kullanarak sistemdeki zayıflıkları belirlerler. Bu yöntem, tehdit modellemesi ve güvenlik risk değerlendirmesi gibi güvenlik testlerinde kullanılan diğer tekniklerle birlikte kullanılır. Belirli bir yazılım geliştirme sürecine bağlı değildir ve bu nedenle yazılım geliştirme yaşam döngüsündeki gereksinim tanımlama aşamasında kullanılabilir.

1.2. Geliştirme Aşamasında Uygulanacak Güvenlik Araçları

Yazılım geliştirme aşamasında kullanılacak yazılım araçlarıyla birlikte fayda sağlayacak bir takım önlem arasında şu hususlar yer almaktadır:

Yazılım kodlama aşamasında, modüler yapı, yapısal bütünlük, parametrik hazırlık ve veri doğruluğu gibi güvenli yazılım kodlama teknikleri kullanılmalı, sistem bütünlüğü korunmalı, veri girişi anında kontrol edilmeli, çıkış verisi bilgi sızıntısına izin vermemeli ve elektronik iletişimde kriptografi teknikleri uygulanmalıdır. Kötü niyetli kodlara karşı saptama, önleme ve kurtarma kontrolleri ile dış kaynaklardan alınan yazılım hizmetleri titizlikle denetlenmelidir (T.C. Ulaştırma ve Altyapı Bakanlığı Siber Güvenlik Koordinatörlüğü Güvenli Yazılım Geliştirme Rehberi, 2018). Geliştirme aşamasında kullanılacak yazılım araçları kapsamında;

SonarQube, SonarQube uygulaması, kod kalitesini artırmak ve güvenliği sağlamak amacıyla kullanılan bir araçtır. Projelerin kodlarını analiz eder, hataları, kod tekrarlarını, karmaşıklığı ve güvenlik açıklarını tespit eder. Bu sayede, geliştiricilere daha temiz kod yazma konusunda geri bildirim sağlar, güvenlik açıklarını tespit eder ve maliyet tasarrufu sağlar. Ayrıca, kod kalite standartlarının sürdürülmesine yardımcı olur, kodun anlaşılabilirliğini arttırır ve kod revizyonlarını kolaylaştırır. Yazılım projelerinin sürdürülebilirliğini ve uyumluluğunu artırmak için kullanılır.

CRSF Guard, web uygulamalarında kullanılan bir güvenlik önlemidir ve Cross-Site Request Forgery (CRSF) saldırılarına karşı koruma sağlar. Bu mekanizma, kullanıcıların bilgisi olmadan veya

istemeden, oturum açıkken veya kimlik doğrulaması yaparken başka bir web sitesi üzerinden istekler göndermesini önlemek için tasarlanmıştır. CSRF Guard, genellikle web uygulamalarının güvenliğini artırmak ve kullanıcıların hassas işlemler yaparken güvende olmalarını sağlamak amacıyla kullanılır. Bu mekanizma oturum kimlik doğrulaması kullanarak çalışır ve böylece gerçek isteklerin doğrulanması sağlanır, saldırganların istekleri gerçekleştirmesi engellenir.

ESLint, ESLint, JavaScript ve ECMAScript tabanlı kodları analiz etmek ve hataları, stil ihlallerini bulmak için kullanılır. ESLint, JavaScript kodlarının değerlendirilmesi ve hataların veya kod kalitesiyle ilgili sorunların tespiti için kullanılan bir açık kaynaklı JavaScript aracıdır. Statik kod analizi yoluyla, belirli bir stil kılavuzuna uyulmasını sağlar ve genellikle geliştiricilere yaygın hataları veya kötü uygulamaları belirleme imkanı sunar. Bu şekilde, daha tutarlı, okunabilir ve güvenilir JavaScript kodları oluşturulmasına yardımcı olur. ESLint, genellikle JavaScript geliştirme süreçlerinde kullanılan ve proje bazında özelleştirilebilen yapılandırma dosyalarıyla (örneğin, .eslintrc dosyası) entegre edilir. Bu araç, özellikle büyük ve karmaşık projelerde, kod temizliğini ve kalitesini artırmak için önemli bir rol oynar.

Attack Surface Detector, web uygulamalarının saldırı yüzeyini belirlemek için kullanılan bir araçtır. Saldırı yüzeyini analiz etmek için ağ haritalama, port taraması, açık servislerin tespiti, güvenlik açıkları taraması gibi yöntemler kullanabilir. Bu analizler sonucunda elde edilen bilgileri kullanarak uygulamanın güvenlik durumunu değerlendirebilir ve potansiyel riskleri belirleyebilir. Saldırganların hedeflerine ulaşmak için kullanabilecekleri açıkları

ve zayıf noktaları belirleyerek, organizasyonların güvenliklerini artırmalarına yardımcı olabilir.

Enterprise Security API (ESAPI), yazılım geliştiricilere, mevcut ve yeni uygulamaların güvenlik uygulamalarında yardımcı olan bir program kütüphanesidir. Güvenli yazılım geliştirme uygulamaları için bir dizi araç ve yönerge sağlar. Bu kütüphane, yazılım uygulamalarının güvenlik açıklarını azaltmak ve karşılaştığı güvenlik tehditlerini en aza indirmek için kullanılır. Yazılım sistemlerinin uygulanma veya bakım aşamasında kullanılabilir.

HTML Sanitizer, kullanıcı girdileri gibi dış kaynaklardan alınan HTML veya diğer metin içeriğini güvenli bir biçimde işlemek için kullanılır. Temel amacı, potansiyel olarak zararlı olan veya güvenlik açıklarına neden olabilecek HTML veya diğer metin içeriğini temizlemek ve güvenli bir şekle dönüştürmektir. Web uygulamalarında Cross-Site Scripting (XSS) gibi saldırıları önlemeye yardımcı olur.

FindSecBugs (OWASP Java Find Security Bugs), java dilinde yazılmış uygulamaların kod denetimleri için kullanılan bir araçtır. Özellikle Java projelerinde kullanılan bu araç, statik kod analizi yaparak uygulama kodunda bulunan SQL enjeksiyonu, Cross-Site Scripting(XSS), kimlik doğrulama problemleri gibi potansiyel güvenlik zafiyetlerini ve zayıf noktaları belirler. Ayrıca Eclipse, IntelliJ, Android Studio gibi popüler yazılım geliştirme ortamları (IDE'ler) ile entegre edilebilir.

Application Gateway, web uygulamalarının yüksek ölçekte yönetilmesi ve güvenliğinin sağlanması için tasarlanmış bir uygulama dağıtım denetleyicisidir. Bu hizmet, web uygulamalarının

yük dengeleme, güvenlik duvarı koruması, SSL yönetimi ve trafik yönlendirme gibi özelliklerle yönetilmesini sağlar. Bu sayede, Azure'da barındırılan web uygulamalarının performansını artırır ve güvenliklerini sağlar. Yazılım geliştirme yaşam döngüsünün tasarım aşamasında da kullanılabilir.

GitGuardian, yazılım geliştirme süreçlerinde kullanılan kod paylaşım platformları ve hizmetlerinde (GitHub, GitLab vb.) bulunan hassas verilerin (API anahtarları, kimlik bilgileri, parolalar) yanlışlıkla paylaşılması veya sızdırılması durumunda bu verileri tespit etmek ve korumak için tasarlanmıştır. Kullanıcıların kod deposundaki değişiklikleri ve depolanan verileri otomatik olarak izler. Sızıntıları ve potansiyel güvenlik risklerini önlemek amacıyla kullanıcıları uyarır.

1.3. Test Aşamasında Uygulanacak Güvenlik Araçları

Yazılım geliştirme süreci test aşamasında kullanılacak yazılım araçlarıyla birlikte bir dizi önlem alınması gerekmektedir. Bu önlemler arasında, yazılım uygulamasının test aşamasında, modül kalite ve miktar testleri uygulanmalı, kodlama, test ve işletim imkanları ayrılarak yetkisiz kullanıcı erişimi ve müdahale riskleri azaltılmalıdır. Sistem performansı, veritabanı büyüklüğü ile listelenen ve sorgulanan kayıt sayısı arasında kontrol edilmelidir. Test verileri özenle seçilmeli, korunmalı ve kontrol edilmelidir. Yazılım ürünleri, gerçek kullanıcı donanımı ve işletim ortamında tüm gereksinimleri karşılayıp karşılamadığının kontrolü sağlanmalıdır (T.C. Ulaştırma ve Altyapı Bakanlığı Siber Güvenlik Koordinatörlüğü Güvenli Yazılım Geliştirme Rehberi, 2018). Bu süreçte kullanılacak yazılım araçları kapsamında;

Amass, kurumsal bir yazılım sisteminin mevcut ve eski altyapısını güvence altına almaya yöneliktir. Bilgi toplama süreçlerinde ve siber güvenlik testlerinde geniş kapsamlı bir keşif sağlamak amacıyla tasarlanmıştır. DNS aktif keşfi, alt alan keşfi, port taraması, SSL sertifikası keşfi, hedef varlıkların haritalanması ve ilişkisel analiz gibi işlevleri içerir. Bu sayede, bir hedef hakkında mümkün olan en geniş bilgi yelpazesini elde etmek için kullanılır. Potansiyel saldırılara karşı duyarlı olan organizasyonel varlıkları tanımlayabilir ve bu şekilde özellikle yazılım güvenliği profesyonellerine yardımcı olur.

Defectdojo, zafiyet yönetimini hedefleyen ve farklı yazılım geliştirme yaşam döngüsü aşamalarında uygulanabilen bir araçtır. Yazılım geliştirme sürecindeki güvenlik kusurlarını izlemek, yönetmek ve raporlamak için kullanılır. Yazılım güvenlik testlerinden gelen bulguları toplar, sınıflandırır, önceliklendirir ve yönetir. Ayrıca, güvenlik ekibine bulguların çözüm sürecini izlemeleri ve güvenlik açıklarını kapatma ilerlemesini raporlamaları için bir arayüz sağlar.

Offensive Web Testing Framework (OWTF), web uygulamalarının güvenlik açıklarını belirlemek maksadıyla zafiyet taraması, sızma testi, güvenlik açıklarının analizi ve raporlama gibi çeşitli güvenlik testlerini destekleyen birçok güvenlik aracının bir araya getirilmesiyle oluşturulmuş bir platformdur. Kullanıcı arayüzü, güvenlik testlerini yönetmeyi kolaylaştırmak için geliştirilmiş olup kullanıcıların test sonuçlarını anlamalarına ve analiz etmelerine yardımcı olacak şekilde tasarlanmıştır. Özellikle, penetrasyon testini daha etkili hale getirmeyi amaçlar.

Zed Attack Proxy (ZAP), web uygulamalarındaki güvenlik açıklarını belirlemek, sızma testleri yapmak ve güvenlik açıklarını gidermek için tasarlanmıştır. Aktif ve pasif zafiyet taraması, güvenlik açıklarının otomatik algılanması, otomatik sızma testleri, tarayıcı entegrasyonu, API erişimi, otomasyon yetenekleri ve kapsamlı raporlama gibi imkan ve kabiliyete sahiptir. Kullanıcıların web tarayıcıları üzerinden trafiği yönlendirerek web uygulamalarıyla etkileşimde bulunmalarını sağlar. Aynı zamanda test aşamasında trafiği izleyebilme, değiştirebilme ve manipüle edebilme gibi kabiliyetleri bulunmaktadır.

APICheck, API'lerin güvenlik açıklarını ve performans sorunlarını tespit etmek için kullanılan bir dizi farklı araç ve bileşenin bir araya getirilmesinden oluşmaktadır. bu araç ve bileşenleri birleştirerek API'lerin güvenlik zafiyetlerini tespit etmek ve çeşitli güvenlik testlerini gerçekleştirmek için kullanıcı dostu bir arabirim sunar. Güvenlik açıkları taraması, otomatik güvenlik testleri, API performans testleri, API metrikleri izleme, otomatik testlerin entegrasyonu ve kapsamlı raporlama gibi kabiliyetlere sahiptir.

Mobile Audit, mobil uygulamaların güvenlik açıklarını tespit etmek, analiz etmek ve raporlamak maksadıyla kimlik doğrulama ve yetkilendirme mekanizmalarının kontrol edilmesi, hassas verilerin korunması, veri şifreleme, ağ trafiğinin kontrolü gibi yetenekleri içeren denetim ve test aracıdır. Uygulamanın performansının değerlendirilmesi, yanıt süreleri, yük testleri, bellek kullanımı ve diğer performans metrikleri üzerindeki incelemeleri içermektedir. Mobil uygulamaların güvenliğini, performansını ve

kullanılabilirliğini artırmak, kullanıcı memnuniyetini ve güvenliğini sağlamak için önemlidir.

Nettacker, birçok farklı güvenlik testi modülüne sahiptir ve bu modüller, port taraması, servis tespiti, zafiyet taraması, brute-force saldırıları, oturum yönetimi analizi gibi çeşitli güvenlik testlerini gerçekleştirmek için kullanılan açık kaynaklı bir sızma testi aracıdır ve ağ güvenliği testlerini gerçekleştirmek için kullanılır. Ağ güvenliği testlerini otomatikleştirmek ve ağ güvenliğini artırmak isteyen güvenlik uzmanları tarafından yaygın olarak kullanılmaktadır.

Purpleteam, çalışan bir web uygulamasındaki veya arayüzündeki güvenlik hatalarını bulma yeteneğine sahip bir platformdur. Güvenlik sorunu bulduğunda nerede olduğu ve ne olduğu hakkında bildirimler gönderir. Bir dizi güvenlik testini otomatikleştirir ve bu testleri birleştirerek uygulamanın güvenlik durumunu değerlendirir. Özellikle DevOps süreçlerine entegre edilmek üzere tasarlanmıştır. Purpleteam, bir dinamik uygulama güvenliği testi aracıdır ve bu nedenle yazılım geliştirme test aşamasına aittir(OWASP, t.y.; Dimov, Aleksandar & Vladimir Dimitrov, 2021)

1.4. Yazılımların Kullanımı Sırasında ve Bakım Aşamasında Kullanılan Araçlar

Geliştirilen yazılımların kullanımı esnasında yazılım araçlarıyla birlikte bir dizi önlem alınması gerekmektedir. Bu önlemler arasında yazılımlar modüler, yapısal bütünlüğe sahip ve programcı müdahalesi en aza indirilmiş olmalı, modül eklemeleri veya değişiklikler sistem bütünlüğünü etkilememeli, v eri girişleri

kontrol edilmeli, kayıt bilgileri güvenli tutulmalıdır. Uygulama çıktıları ve veri işleme hataları için doğrulama kontrolleri uygulanmalıdır. Veri güvenliği için kriptografi teknikleri kullanılmalı ve dış kaynaklı yazılım hizmetleri titizlikle izlenmelidir (T.C. Ulaştırma ve Altyapı Bakanlığı Siber Güvenlik Koordinatörlüğü Güvenli Yazılım Geliştirme Rehberi, 2018). Bakım aşamasında kullanılacak yazılım araçları kapsamında;

AntiSamy, kullanıcı tarafından yüklenen HTML ve Cross-Site Scripting(CSS) dosyalarının kötü amaçlı sorunlar içermemesini sağlayan bir arayüzdür. Kullanıcıların girdiği HTML veya CSS kodlarını temizleyerek, güvenlik açıklarını kapatır ve potansiyel Cross-Site Scripting saldırılarını engeller. Bu şekilde, yazılım geliştirme yaşam döngüsünün yazılım kullanımı veya bakım aşaması için güvenlik mekanizmaları sağlar.

Bug Logging Tool (BLT), yazılım geliştirme sürecinde hata takibi ve hata raporlama işlemlerini yönetmek için kullanılan bir araçtır. Sorun izleme ve yönetimi için araçlar sağlayarak bakımı kolaylaştırır. Yazılım geliştirme sürecinde hata takibi ve raporlama için kullanılabilir. Hataların bildirilmesi, takibi, çözümü ve raporlanması gibi işlevleri sağlamaktadır. Böylelikle yazılım geliştirme sürecinde hata yönetimini kolaylaştırır ve yazılım kalitesini artırır.

O-Saft, SSL (Güvenli Yuva Katmanı) protokolünü analiz etmek için geliştirilmiştir. SSL/TLS bağlantıları üzerinde derinlemesine analiz yapabilen ve SSL/TLS trafiğini izleyen, çözen ve analiz eden bir araçtır. Ağ trafiğindeki güvenlikle ilgili sorunları tespit etmek ve geliştirme sürecinde SSL/TLS konfigürasyonlarını

optimize etmeyi mümkün kılar. Penetrasyon testçileri, güvenlik denetçileri veya sunucu yöneticileri tarafından kullanılmak üzere tasarlanmıştır (OWASP t.y.; Dimov, Aleksandar & Vladimir Dimitrov, 2021).

1.5. Eğitim İçin Kullanılacak Yazılımlar

Güvenlik eğitim araçları, güvenlik kavramları, uygulamaları veya becerileri hakkında bilgi edinmeyi sağlayan kaynaklar, kurslar veya zorluklar sunarak yardımcı olan yazılım araçlarıdır. Bu araçlar, güvenlik bilincini artırmaya, bilgi seviyesini geliştirmeye ve güvenlik trendleri veya en iyi uygulamalar konusunda güncel kalmaya yardımcı olabilir. Güvenlik eğitim araçlarının örnekleri arasında şunlar yer alır:

Juice Shop, siber güvenlik alanında eğitim ve farkındalık oluşturmak amacıyla oluşturulan ve bilinçli olarak birçok zafiyeti içeren bir web uygulamasıdır. Bu açıkları tespit etmek, anlamak ve düzeltmek için çeşitli görevler gerçekleştirilir. Öncelikle kullanıcılar uygulamada bulunan zafiyetleri sıralar ve müteakiben DoS (Denial of Service), XSS (Cross-Site Scripting), CSRF (Cross-Site Request Forgery), SQL Injection, ve diğer yaygın güvenlik açıklarını bulurlar. Aynı zamanda, yazılım geliştirme yaşam döngüsünün tüm aşamalarında uygulanabilir olarak sınıflandırılabilir, ancak başlıca faydası proje maliyetlerinin en aza indirilebileceği erken aşamalarda ortaya çıkar.

KnowBe4, kuruluşların çalışanlarını siber güvenlik konularında eğitmek ve farkındalıklarını artırmak için kullanılan bir platformdur. Bu platform, bilinçli bir şekilde yapılan siber saldırılara karşı korunma amacıyla geliştirilmiştir. Kullanıcılarına sahtekarlık

e-postaları, fidye yazılımı, kimlik avı gibi siber tehditlere karşı korunma konusunda eğitim ve farkındalık kazandırmayı amaçlar.

Pygoat, kullanıcıların Python programlama dilini kullanarak siber güvenlik becerilerini geliştirmelerine olanak tanır. Platform, güvenlik uzmanlarının ve siber güvenlik öğrencilerinin pratik yapmasını sağlayarak, güvenlik zafiyetlerini tespit etme ve düzeltme konularında deneyim kazanmalarını sağlar. Kullanıcılar, gerçek hayattaki senaryolara dayalı olarak Python kodlama becerilerini uygulayarak ve geliştirerek çeşitli güvenlik testlerini gerçekleştirirler.

Secure Coding Dojo, yazılım geliştirme ekibinin güvenli kodlama becerilerini geliştirmek için kullanılan bir eğitim ve pratik platformudur. Yazılım geliştiricilerini yazılım kodundaki güvenlik hatalarını bulmaya ve önlemeye yönelik bir güvenlik eğitimi sağlar. Bu araç, geliştirme aşamasında uygundur ve saldırı/savunma çiftleri şeklindeki zorluklar üzerine dersler sunar.

Snakes and Ladders yazılım geliştirme süreçlerinde kullanılan bir görselleştirme tekniğidir. Bu teknik, genellikle Agile yazılım geliştirme yöntemlerinde kullanılarak, projenin ilerlemesini takip etmek ve süreci daha anlaşılır hale getirmek için kullanılır. Yazılım geliştirme süreçlerinde "Snakes and Ladders" tekniği, projenin belirli aşamalarda bulunduğu noktayı temsil eden bir tahta kullanır. Tahtanın kareleri, projenin farklı görevlerini veya aşamalarını temsil eder. Yazılım ekibi, her bir görevin tamamlanmasıyla birlikte tahtada ilerler. Ancak, belirli durumlar veya engellerle karşılaştıklarında geriye gidebilirler veya belirli başarılar elde ettiklerinde daha hızlı ilerleyebilirler. Bu teknik,

projenin ilerlemesini görsel ve interaktif bir şekilde takip etmeyi kolaylaştırır.

Proofpoint Security Awareness Training, organizasyonların çalışanlarını siber güvenlik tehditlerine karşı eğitmek ve bilinçlendirmek amacıyla kullanılan bir güvenlik eğitim platformudur. İnteraktif eğitim modülleri, senaryolar, simülasyonlar, testler ve diğer kaynaklar aracılığıyla çalışanlara siber güvenlik konularında eğitim sağlar. Bu eğitimler, oltalama (phishing), fidye yazılımı, kimlik avı (spear phishing), güvenli parola kullanımı, veri koruma yöntemleri ve diğer güvenlik konularını içerebilir.

Honeypot, siber saldırıları tespit etmek ve saldırganların faaliyetlerini izlemek için kullanılan bir güvenlik aracıdır. Bu araç, gerçek bir sistem gibi davranan ancak aslında gerçek veri içermeyen sahte bir sistem veya ağdır. Saldırganlar, bu sahte sistemlere veya ağlara saldırdıklarında, güvenlik uzmanları bu saldırıları tespit edebilir, analiz edebilir ve saldırganların faaliyetlerini takip edebilirler. Sınıflandırmamızdaki başka bir belirgin kategori olmamasına rağmen, bu araç aslında yatay bir araçtır ve tüm yazılım geliştirme yaşam döngüsü aşamalarında uygulanabilir. Bu nedenle, eğitim kategorisine en uygun olduğu düşünülmektedir.

IoTGoat, Internet of Things (IoT) cihazları üzerinde güvenlik açıklarını tespit etmek ve anlamak için kullanılan bir eğitim platformudur. Çeşitli IoT cihazlarının güvenlik zafiyetlerini simüle eden bir sanal laboratuvar ortamı sunar. Bu platform, kullanıcıların gerçek IoT cihazları üzerinde güvenlik testleri yapmadan önce pratik yapmalarını sağlar. Yazılım geliştiricilerin IoT uygulamaları için en

yaygın IoT cihazı zafiyetlerini bulmalarını ve önlemelerini sağlamayı amaçlar.

SamuraiWTF (Web Testing Framework), VMWare VirtualBox üzerinde çalışabilen bir sanal makinedir ve web penetrasyon testi motoru olarak yapılandırılmıştır. Ancak, yaratıcılarına göre, başlıca amacı bir eğitim platformu olarak hizmet etmektir. Bu amaçla, çeşitli güvenlik risklerini değerlendirmeyi öğrenmek için gerekli olan zafiyet tarama, saldırı simülasyonu, veritabanı güvenliği, ağ analizi gibi bir dizi uygulamayı ve aracı içerir (OWASP, t.y.; Dimov, Aleksandar & Vladimir Dimitrov, 2021).

Bu araçlar, yazılım geliştirme sürecinde güvenliği artırmak ve potansiyel zafiyetleri belirlemek için kullanılabilir. Ancak, her projenin ihtiyaçları farklı olduğundan doğru araçları seçmek için projenin gereksinimlerini değerlendirmek önemlidir.

Yazılımları daha güvenli hale getirmek ve potansiyel güvenlik sorunlarından daha az etkilenmek amacıyla yazılım geliştirme yaşam döngüsünün tüm aşamalarında bahsedilen yazılım araçları kullanılabilmektedir. Ayrıca yazılım araçlarının yanı sıra bazı temel prensip ve yöntemlerin kullanılması siber güvenlik ortamının istenilen seviyeye ulaşmasına büyük katkı sağlayacaktır.

2. Yazılım Güvenliği İlkeleri

Yazılım güvenliği, geliştirilen yazılımların dışardan gelecek ya da içerde oluşabilecek tehditlere karşı önlem alınmasıdır. Kullanılan yazılımlarda oluşacak bir sorun gündelik hayatta aksamalara yol açabilmektedir. Hatta öyle ki bankacılık, ulaşım, haberleşme gibi kritik sektörlerde kullanılan uygulamalarda

oluşacak sorunlar hayatı durma noktasına getirebilmektedir. Yazılımların bu denli kullanılması ve insan hayatının vazgeçilmez bir parçası olması onları kötü niyetli kişilerin hedefi haline getirmiştir. Yazımların üzerinde barındırdığı zafiyetlerden yararlanıp birçok siber saldırı yapılmaktadır. Bu saldırılar sonucu insanların kişisel bilgileri çalınmakta, kişi ve kurumların kullanmış olduğu bilgisayar, telefon gibi elektronik cihazlar ele geçirilmektedir. Bu siber saldırılardan korunmak için yazılım sahipleri ciddi yatırımlar yapmaktadır. Yazılımlarını güvenlik yönünde sıkılaştırmak için çeşitli güvenlik ürünlerini yazımlarını kendi yazılımlarına entegre etmekte ve siber güvenlik uzmanları çalıştırmaktadırlar. Bir yazılımın güvenliği yazılımın işlevine göre çeşitli katmanlarda sağlanır. Geliştirilen uygulamanın etkileşim boyutuna göre bu katmanlar azalır ya da artar. Geliştirilen uygulamada ağ haberleşmesi düzeyinde bir etkileşim söz konusu ise, TCP/IP modeli üzerinden düşünürsek, Uygulama, Taşıma, Ağ ve Fiziksel katman olmak üzere dört farklı katmanda güvenliğinin sağlanması gerekmektedir (İzzet Gökhan ÖZBİLGİN & Mustafa ÖZLÜ, 2010).

Yazılımın hatalarını minimuma indirip güvenliği ve kalitesini artırmak amacıyla çeşitli güvenli yazılım geliştirme prensipleri bulunmaktadır. Aşağıda en temel güvenli yazılım geliştirme ilkeleri açıklanmıştır.

2.1. En Az Yetki (Least Privilege)

Yazılım geliştirmesi, nesneler, sınıflar, fonksiyonlar ve kullanıcılar gibi çeşitli bileşenlerden oluşan bir yapıyı içerir ve her bir bileşenin kendine özgü işlevleri vardır. Bu temel ilkenin amacı, her bir bileşene, işlevini gerçekleştirebilmek için gerekli minimum

yetkiyi tanımaktır. Bu yaklaşım, bileşenlerin amaç dışı kullanımını engelleyerek güvenliği artırır.

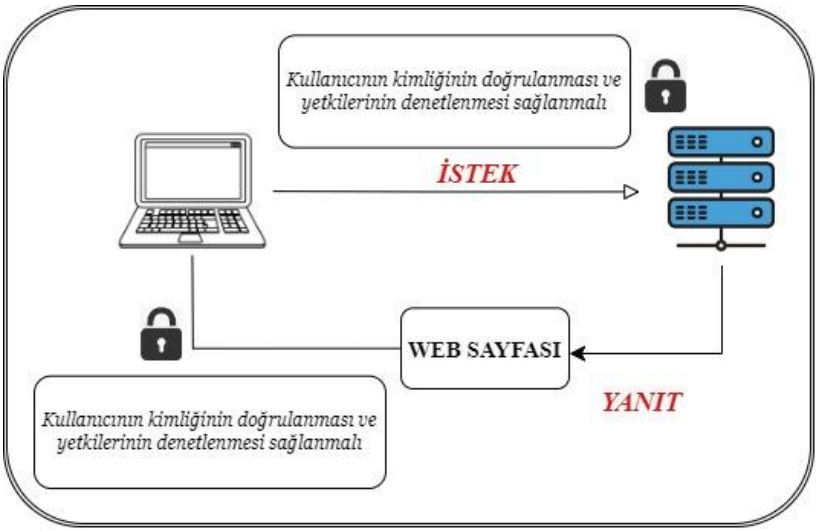
Bu prensibi anlamak için şu örnek de ele alınabilir: Varsayalım ki geliştirilen bir uygulama, sunucu üzerinde belirli bir dizine dosya yazma yeteneğine sahiptir. Bu durumda, uygulamanın sunucuda çalıştırıldığı kullanıcıya, yalnızca ilgili dizindeki dosyaları düzenleme yetkisi tanınmalıdır. Aksi takdirde, uygulamanın işleyişinde anormal durumlar ortaya çıkabilir veya bir saldırgan, sunucu üzerindeki diğer dosyalara ve uygulamalara müdahale ederek uygulamayı ele geçirebilir (TÜBİTAK BİLGEM, 2018; Nigel Douglas, 2022).

Bu senaryo, en az yetki ilkesinin pratik bir örneğini sunmaktadır. İlgili kullanıcıya sadece gerekli olan yetkilerin verilmesi, uygulamanın işlevselliğini sürdürürken güvenliği artırabilir. Bu prensip, bir uygulama veya kullanıcıya sadece belirli görevleri yerine getirebilme yeteneği tanıyarak, potansiyel güvenlik risklerini minimize etmeyi amaçlar (S. A. Ebad, 2022; N. Davis, & ark., 2004).

2.2. Tüm Erişimlerin Denetlenmesi (Audit Access)

Erişim denetimi ilkesi, uygulamanın birleşenlerine yönelik erişimleri kontrol altında tutarak yetkisiz erişimleri önlemeyi hedeflemektedir. Örneğin, bir uygulamada normal kullanıcı ve yetkili kullanıcı gibi farklı kullanıcı rolleri bulunabilir. Erişim gerçekleştiğinde, kurulan denetim mekanizmasının normal ve yetkili kullanıcıları doğru bir şekilde tanımlayabilmesi gerekmektedir. Bu mekanizma, kullanıcı rollerine göre bileşenlere erişimi ayırt etmeli ve düzenlemelidir.

Her bir öğeye yönelik her erişim durumunda yetki denetimi uygulanmalıdır. Bu prensibe uyum, sistem genelinde kapsamlı bir erişim kontrolünü gerektirir. "Tüm erişimleri denetle" ilkesi, normal işleyiş durumlarından bağımsız olarak, başlatma, geri kazanım, kapatma ve bakım aşamalarında da erişim kontrolünün sağlanmasını içerir.



Şekil 1. Sunucu ve istemci arasındaki yetki kontrolü

Şekil 1'de görüldüğü gibi sunucu ve istemci arasında yetki kontrolü çift taraflı olup, aynı kontrol yanıtlanırken uygulanmalıdır (TÜBİTAK BİLGEM, 2018; N. Davis & ark., 2004).

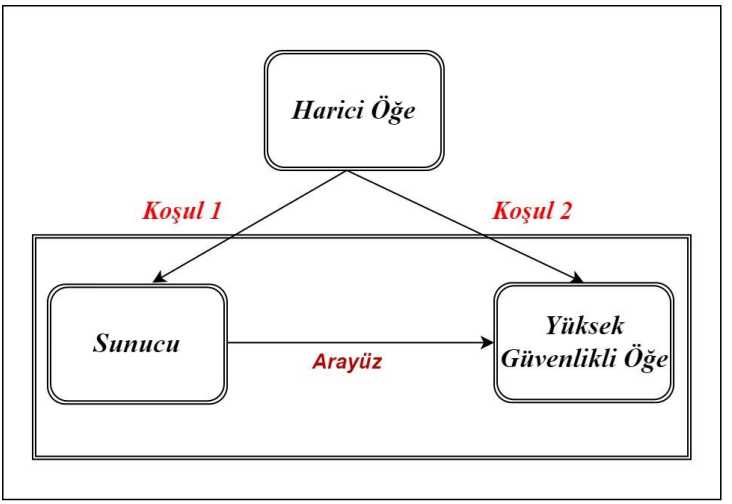
Bu ilkeye bir örnek olarak bir kuruluşun bilgi sistemlerine yetkisiz erişimi önlemek ve güvenliği artırmak amacıyla bir kimlik doğrulama ve yetkilendirme sistemi uygulaması verilebilir. Bu sistem, kullanıcıların sisteme giriş yaparken kimliklerini doğrulamak için kullanıcı adı ve parola gibi güvenlik bilgilerini girmelerini gerektirir. Ayrıca, belirli kullanıcı rollerine göre erişim

yetkileri atanır ve her kullanıcının yaptığı işlemler kaydedilir. Böylece, tüm erişimler denetlenir ve izlenir, bu da potansiyel güvenlik ihlallerini tespit etmeye ve engellemeye yardımcı olur.

2.3. Görevler Ayrılığı (Separation of Duties)

Görev ayrılığı ilkesinde bir role birden fazla veya yüksek yetki verilmemesi durumudur. Bu ilke en az yetki ilkesi karıştırılabilir ama farklı bir ilkedir. Bu ilkede ana amaç kritik ve büyük boyutlu işlemlerin tek bir role bağlanmaması durumudur. Bu kritik ve büyük boyutlu işlemler parçalara bölünmeli ve her bir parçanın kontrolü bir role atanmalıdır. Bir başka deyişle bir sistem, yüksek güvenlik düzeyine sahip bir işlemi yalnızca tek bir koşulu yerine getirerek izin vermemelidir. Yüksek güvenlik gerektiren bir işlemi başlatmak için, birden fazla koşulun doğrulanması gerekmelidir. Yetkilerin ayrımını gösteren bir şema Şekil 2'de belirtilmiştir.

Sıradan bir işlem için ilk olarak sunucu öğede oturumun açılması sağlanmalıdır(1.Koşul).



Şekil 2. Yetkilerin ayrımı için örnek koşullar

Yüksek güvenlik seviyesine sahip bir işlem gerçekleştirmek için, öncelikle yüksek güvenlik düzeyine sahip olan bir hesaba oturum açılmalıdır (2. Koşul). Yüksek güvenlik gerektiren işlem, her iki koşulun da sağlandığı durumda gerçekleştirilmelidir (TÜBİTAK BİLGEM, 2018)

2.4. Savunma Derinliği Oluşturma (Defense In Depth)

Savunma derinliği ilkesinin amacı, katmanlı bir savunma stratejisi oluşturmaktır. Tek bir savunma mekanizması yerine çeşitli savunma mekanizmalarının kullanılması esas alınır. Bu yaklaşım, bir savunma mekanizmasının aşıldığı durumlarda diğer savunma katmanlarının sistemi korumasını sağlamayı hedefler. Bu bağlamda, farklı güvenlik ürünlerini uygulamaya entegre etmek mümkündür.

Bazı durumlarda, bir güvenlik ürününün algılayamadığı zararlı aktivitelerin, başka bir ürün tarafından tespit edilebilme potansiyeli bulunmaktadır. Bu durumda, çeşitli güvenlik ürünlerinin

entegrasyonu, geniş bir tehdit yelpazesine karşı etkili bir savunma sağlamak adına önemli bir stratejidir (S. A. Ebad, 2022).

Örneğin, bir şirketin siber güvenlik stratejisi, sadece dışarıdan gelen siber saldırıları önlemekle kalmamalı, aynı zamanda içeride çalışanların yanlışlıkla veya kötü niyetle şirket bilgilerine zarar vermesini engellemelidir. Bu bağlamda, ağ içinde çok katmanlı güvenlik önlemleri uygulanabilir. Örneğin, güvenlik duvarları, ağ izleme sistemleri, kullanıcı yetkilendirme ve erişim kontrolü gibi önlemlerle içerideki bir kullanıcının yetkileri sınırlanabilir ve anormal aktiviteler tespit edilebilir. Kısacası sistemdeki güvenlik önlemleri sadece dış saldırılara karşı değil, aynı zamanda içeriden kaynaklanan tehditlere karşı da etkili olmalıdır.

2.5. Saldırı Yüzey Alanını Azaltma (Minimizing Attack Surface Area)

Saldırı yüzey alanı, yazılımın yetkisi olmayan bir kullanıcı tarafından, yazılımın bulunduğu ortamdan veri elde etmek, veriyi değiştirmek ve farklı yetkisiz işlemler gerçekleştirmek amacıyla erişebileceği tüm noktaları ifade eder. Saldırı yüzeyini azaltma ise, geliştirilen bir uygulamadaki etkileşim noktalarını sınırlayarak gerçekleştirilir. Bu bağlamda, uygulamada kullanılan her özelliğin gerçekten gerekli olup olmadığı titizlikle değerlendirilir. Çünkü uygulamada kullanılan özelliklerin artması, daha fazla etkileşim noktasının ortaya çıkmasına neden olur. Bu noktaların sayısındaki artış, güvenlik önlemlerini karmaşıklaştırabilir ve maliyeti artırabilir. Bu nedenle, uygulamada gereksiz olan özelliklerin kaldırılması, saldırı yüzeyinin azaltılması ve genel güvenlik durumunun iyileştirilmesi açısından kritik bir öneme sahiptir (S. A. Ebad, 2022 ; N. Davis & ark., 2004).

Örneğin, gereksiz servislerin kapatılması veya devre dışı bırakılmasıyla, sunucu tarafında saldırı yüzeyi azaltılabilir. Kullanılmayan veya gereksiz sayıdaki ağ bağlantı noktalarının kapatılması, saldırganların ağa sızma olasılığını azaltabilir. Ayrıca, uygulamanın dışa açıkta bulunan ve gereksiz olan iletişim portlarının kapatılması da saldırı yüzeyini azaltan bir önlem olabilir. Bu ilkeye göre, gereksiz veya kullanılmayan özelliklerin kapatılması, güvenlik açısından gereksiz riskleri ortadan kaldırabilir ve saldırı yüzey alanını minimal seviyeye çekebilir.

2.6. Açık Tasarım (Open Design)

Açık tasarım prensibi, Claude Shannon'ın "Assume the enemy knows the system" (Düşmanın sistemi bildiğini varsay) ilkesine dayanır ve aynı zamanda Kerckhoffs prensipleri kapsamında geçerlidir. Bu prensibe göre, sistemler, düşmanın hemen tam aşinalık kazanacağı varsayımıyla tasarlanmalıdır. Bu ilkede amaç, geliştirilen uygulamanın kaynak kodları ele geçirilse veya uygulamanın bulunduğu sunucuya sızılsa bile kritik bilgilerin ele geçirilememesini sağlamaktır.

Bu hedefe ulaşmak için kriptografik fonksiyonlar kullanılarak veriler şifrelenir ve şifreleme anahtarları özenle korunur. Temel amaç, sistemden ziyade şifreleme anahtarlarını korumaktır. Bu sayede, saldırganlar sistemin iç yapısını bile bilseler dahi kritik bilgilere erişmeleri engellenmiş olur (S. A. Ebad, 2022; TÜBİTAK BİLGEM, 2018)

Açık kaynaklı yazılım projelerinde, projenin kaynak kodları ve tasarım belgeleri genellikle kamuya açıktır ve herkes tarafından erişilebilir. Bu durumda, yazılımın tasarımı, algoritmaları ve iç

işleyişi topluluğun katılımına açıktır. Herhangi bir geliştirici veya kullanıcı, projenin gelecekteki sürümlerine katkıda bulunabilir, hataları düzeltebilir veya yazılımı geliştirebilir. Bu, yazılımın tasarımının topluluk tarafından denetlenebildiği ve iyileştirilebildiği anlamına gelir.

2.7. Güvenli Kapatma (Failing Securely)

Bu prensibin temel amacı, sistemin en kötü senaryoda nasıl tepki vereceğini belirlemektir. Sisteme izinsiz bir erişim veya kritik bir sorunun ortaya çıkması durumunda, sistem kendini güvene alacak şekilde tasarlanmalıdır. Bu, özellikle kritik bir kurumda düşünülebilir; örneğin, izinsiz bir giriş veya alarm durumunun oluşması durumunda, kurumdaki tüm kapılar kilitlenir.

Benzer bir yaklaşım, geliştirilen uygulamalar için de geçerlidir. Saldırı veya arıza durumunda, uygulama kendisini güvenli bir şekilde koruyabilen mekanizmalar içermelidir. Bu, uygulamanın bütünlüğünü ve işlevselliğini sürdürmek amacıyla tasarlanmış güvenlik önlemlerini içerir (S. A. Ebad, 2022; TÜBİTAK BİLGEM, 2018)

Örneğin, "Kullanıcı adı veya şifre hatalı" gibi bir hata mesajı, potansiyel saldırganlara hangi kısmın yanlış olduğu konusunda bilgi verebilir ve saldırıya yönelik bir açık bırakabilir. Bunun yerine, uygulama daha genel bir mesajla kullanıcının hatalı giriş yaptığını belirtmeli ve ayrıntılı bilgileri içermemelidir. Bu şekilde, kullanıcıya hata durumu bildirilirken güvenlik açısından bilgiler korunmuş olur.

2.8. İşlem Kaydı (Logging)

İşlem kaydı (logging), özellikle Linux işletim sistemi üzerinde, işletim sistemi denetiminin kritik bir unsurudur. Birçok

güvenlik olayı genellikle önceki olaylarla birlikte gerçekleşir. Önemli öncül olayları belirleme sürecinde, sistem faaliyetinin hayati işaretleri ve kaydedilmesi gereken denetim olayları önemli bir rol oynar. İşlem kaydı ile dosya erişimleri izlenir, sistem çağrıları takip edilir, kullanıcılar tarafından yürütülen komutlar kaydedilir ve kimlik doğrulama, yetkilendirme ve ayrıcalık yükseltme gibi güvenlik olayları belirlenerek günlüğe kaydedilir. Bir olayın özellikleri genellikle tarih, saat, tür, konu kimliği, sonuç ve hassasiyet etiketlerini içerir.

Örneğin, bir günlük kayıt sistemi kullanıcı giriş ve çıkış saatleri, hangi kullanıcının hangi dosyaları veya sistem kaynaklarını eriştiği, potansiyel saldırı girişimleri veya başarılı saldırılar ve ilgili ağ trafiği verilerini kaydedebilir. Bu günlük kayıtları, güvenlik olaylarını izlemek, anlamak ve bu olaylara müdahale etmek için kullanılabilir. Ayrıca, günlük kayıtları yasal gereksinimlere uygunluk, güvenlik denetimleri ve saldırılara karşı koruma amacıyla saklanabilir. Bu, bir güvenlik ekibine sistemdeki potansiyel sorunları tespit etme ve müdahale etme yeteneği sağlar (S. A. Ebad, 2022; Zeng, L., Xiao, Y. & Chen, H., 2015).

Sonuç

Siber tehdit ortamının büyümesi ve çeşitlenmesi ile birçok siber saldırı meydana gelmektedir. Yazılımlar üzerinde bulunan güvenlik açıkları gün geçtikçe daha tehlikeli bir hal almaya başlamıştır. Bu gerçek, yazılım geliştirme profesyonellerinin uygulamalarının güvenliğini sağlama çabalarını kolaylaştırmalarına yardımcı olurken aynı zamanda geliştirmelerini de karmaşıklaştırabilir. Bu makalede yazılımları daha güvenli hale getirmek ve potansiyel güvenlik sorunlarından en az seviyede zarar

görmek amacıyla yazılım geliştirme yaşam döngüsünün tüm aşamalarında kullanılabilecek yazılım araçları ve güvenli yazılım geliştirme için uygulanacak temel prensipler açıklanmıştır. Yazılım geliştirmenin ana aşamalarına karşı araçların bir sınıflandırması yapılmış ve bu araçların kısa açıklamaları verilmiştir. Sadece açık kaynak ve ticari olmayan araçları dikkate alınmıştır. Böylece siber güvenlik uzmanlarının güvenlik gereksinimlerine uygun olarak güvenlik altyapılarını etkili bir şekilde kurmalarına yardımcı olacaktır.

Kaynaklar

NIST (2023) NATIONAL VULNERABILITY DATABASE. Erişim adresi: https://nvd.nist.gov/general/nvd-dashboard

S. A. Ebad, "Exploring How to Apply Secure Software Design Principles," in IEEE Access, vol. 10, pp. 128983-128993, (2022), doi: 10.1109/ACCESS.2022.3227434.

M. Ruggieri, T. T. Hsu ve M. L. Ali, "Security Considerations for the Development of Secure Software Systems," (2019) IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York, NY, USA, 2019, pp. 1187-1193, doi: 10.1109/UEMCON47517.2019.89930

M. Almousa, M. Keshavarz ve M. Anwar, (2020). Awareness and Working Knowledge of Secure Design Principles: A User Study. In: Moallem, A. (eds) HCI for Cybersecurity, Privacy and Trust. HCII 2020.

R. Khan, S. Khan, H. Khan, ve M. Ilyas, (2022). Systematic Literature Review on Security Risks and its Practices in Secure Software Development. IEEE Access, 10, 5456-5481.

R. Trifonov, O. Nakov, G. Pavlova, S. Manolov, G. Tsochev ve P. Nakov, "Analysis of the Principles and Criteria for Secure Software Development," 2020 28th National Conference with International Participation (TELECOM), Sofia, Bulgaria, (2020), pp. 125-128, doi: 10.1109/TELECOM50385.2020.9299567

R. E. Smith, ``A contemporary look at Saltzer and Schroeder's (1975) design principles," IEEE Secur. Privacy, vol. 10, no. 6, pp. 20_25,

R. Fujdiak, "Managing the Secure Software Development," (2019) 10th IFIP International Conf. on New Technologies, Mobility and Security (NTMS), pp. 1-4,

Richard Linger ve Carmen Trammell. Cleanroom Software Engineering Reference (CMU/SEI-96-TR-022). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, (1996)

İzzet Gökhan ÖZBİLGİN ve Mustafa ÖZLÜ, Yazılım Geliştirme Süreçleri ve ISO 27001 Bilgi Güvenliği Yönetim Sistemi, (2010)

Waterfall model. (2021). Erişim adresi: http://en.wikipedia.org/wiki/Waterfall_model

T.C. Ulaştırma ve Altyapı Bakanlığı Sivil Havacılık Genel Müdürlüğü Havacılık Güvenliği Daire Başkanlığı-Siber Güvenlik Koordinatörlüğü Güvenli Yazılım Geliştirme Rehberi, (2020). Erişimadresihttps://web.shgm.gov.tr/documents/sivilhavacilik/files/pdf/duyuru/2021/HGD/HGD-GYGR.pdf

OWASP Projects, (t.y.). Erişim adresi: https://owasp.org/projects/

Dimov, Aleksandar, ve Vladimir Dimitrov. "Classification of software security tools." Information Systems and Grid Technologies (2021).

TÜBİTAK BİLGEM Bilişim ve Bilgi Güvenliği İleri Teknolojiler Araştırma Merkezi, (2018)

Nigel Douglas, CSPM – Least privilege principle in practice, (2022, 22 Kasım). Erişim adresi: https://sysdig.com/blog/cspm-least-privilege-principle/

N. Davis, W. Humphrey, S. T. Redwine, G. Zibulski ve G. McGraw, "Processes for producing secure software," in IEEE Security & Privacy, vol. 2, no. 3, pp. 18-25, May-June (2004), doi: 10.1109/MSP.2004.21.

AppMaster, Yazılım Gereksinim Analizi (2023). Erişim adresi: https://appmaster.io/tr/blog/yazilim-gereksinimleri-analizi

Zeng, L., Xiao, Y., ve Chen, H. (2015). Auditing overhead, auditing adaptation, and benchmark evaluation in Linux. Security and Communication Networks, 8(18), 3523-3534