
GAZİ ÜNİVERSİTESİ
Gazi
1926

KENDİ KENDİNE DENETİMLİ ÖĞRENME TABANLI TÜRKÇE KONUŞMA TANIMA SİSTEMİ

Alp Kaan TURAN

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

NİSAN 2024

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Alp Kaan TURAN
03/04/2024

KENDİ KENDİNE DENETİMLİ ÖĞRENME TABANLI TÜRKÇE KONUŞMA TANIMA SİSTEMİ

(Yüksek Lisans Tezi)

Alp Kaan TURAN

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Nisan 2024

ÖZET

Otomatik Konuşma Tanıma, kavramsal temelleri 1930'larda atılan ve o yıllardan bu yana üzerinde yoğun çalışmalar yürütülen bir konudur. İlk uygulamaları, 1950'li yılların başında donanım tabanlı, sınırlı çözümler şeklinde ortaya çıkmıştır. Bilgi işleme sistemlerindeki ilerlemelerle birlikte kapsamı genişlemiş, donanım tabanlı çözümler zamanla yerini istatistik temelli çözümlere bırakmıştır. Geleneksel Makine Öğrenmesi ve sonrasında Derin Öğrenme yöntemlerindeki gelişmeler, Otomatik Konuşma Tanıma alanında Yapay Zekânın kullanılmasını mümkün kılmıştır. Böylece, uçtan-uca, doğal konuşmayı tanıma yeteneğine sahip, çok dilli ve konuşmacılı sistemler geliştirilmiştir. Bu çalışmada, Derin Öğrenme yöntemlerinden biri olan Kendi Kendine Denetimli Öğrenme üzerinden Otomatik Konuşma Tanıma sistemleri incelenmiş ve *Whisper* mimarisini kullanan bir Otomatik Konuşma Tanıma sistemi uygulaması geliştirilmiştir. Temel kavramlar ve yöntemler açıklandıktan sonra geliştirilen uygulama üzerinde deney ve ölçümlemeler yapılmıştır. Ardından, yapılan eklemelerle, ince ayar işleminin uygulandığı modeller üzerindeki etkisi değerlendirilmiştir. Son olarak, *Whisper* temel mimarisinde bulunmayan eş zamanlı konuşma tanıma özelliği, kısa gecikmeli konuşma tanıma yeteneğine sahip *Whisper-Streaming* ve *WhisperLive* ek uygulamaları kullanılarak ölçümlenmiştir. Deneyler, Türkçe konuşma veri kümeleri üzerinde, *Whisper* mimarisine ait beş model tipi kullanılarak yürütülmüştür. Üst modellerle, ilgili veri kümeleri üzerinde yapılan ölçümlerde %4,3 ile %14,2 arasında kelime hata oranları elde edilmiştir. İnce ayar uygulanan modellerde, hata oranında %52,38'e varan iyileşmeler gözlemlenmiştir. *Whisper-Streaming* ve *WhisperLive* uygulamalarıyla, güncel Türkçe konuşmalar kullanılarak yapılan ölçümlerde sırasıyla %8,80 ve %16,1 kelime hata oranlarına ulaşılmıştır.

Bilim Kodu	:	92432
Anahtar Kelimeler	:	Otomatik konuşma tanıma, yapay zekâ, derin öğrenme, temsil öğrenme, kendi kendine denetimli öğrenme
Sayfa Adedi	:	91
Danışman	:	Doç. Dr. Hüseyin POLAT

SELF-SUPERVISED LEARNING BASED TURKISH SPEECH RECOGNITION SYSTEM

(M. Sc. Thesis)

Alp Kaan TURAN

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

April 2024

ABSTRACT

Automatic Speech Recognition is a topic whose conceptual foundations were laid in the 1930s and has been the subject of intensive research since then. Its first applications emerged in the early 1950s as hardware-based, limited solutions. With the advances in information processing systems, its scope has widened, and hardware-based solutions have gradually been replaced by statistics-based solutions. Advances in traditional Machine Learning and later Deep Learning methods have made it possible to use Artificial Intelligence in Automatic Speech Recognition. Thus, end-to-end, multi-lingual and multi-speaker systems capable of natural speech recognition have been developed. In this paper, we analyze Automatic Speech Recognition systems using Self-Supervised Learning, one of the Deep Learning methods, and develop an implementation of an Automatic Speech Recognition system using the Whisper architecture. After explaining the basic concepts and methods, experiments and measurements are performed on the developed application. Then, with the additions made, the impact of the fine-tuning process on the implemented models is evaluated. Finally, the simultaneous speech recognition feature, which is not available in the Whisper base architecture, is measured using the Whisper-Streaming and WhisperLive extensions, which are capable of short delay speech recognition. The experiments were conducted on Turkish speech datasets using five model types of the Whisper architecture. With the top models, word error rates between 4.3% and 14.2% were obtained on the relevant datasets. For the fine-tuned models, improvements of up to 52.38% were observed. With the Whisper-Streaming and WhisperLive applications, word error rates of 8.80% and 16.1%, respectively, were achieved using current Turkish speech.

Science Code	:	92432
Key Words	:	Automatic speech recognition, artificial intelligence, deep learning, representation learning, self-supervised learning
Page Number	:	91
Supervisor	:	Assoc. Prof. Dr. Hüseyin POLAT

TEŞEKKÜR

Öncelikle, konu seçiminden başlayarak, bu tezin gelişim sürecindeki yönlendirme ve yardımlarından dolayı saygıdeğer danışmanım Doç. Dr. Hüseyin POLAT'a içtenlikle teşekkür ederim. Her zaman bana destek olan annem başta olmak üzere, aileme ve çalışma arkadaşlarıma yardımları için minnettarım.

Çalışmada kullanılan nümerik hesaplamaların bir kısmı TÜBİTAK ULAKBİM Yüksek Başarım ve Grid Hesaplama Merkezi'nde (TRUBA kaynaklarında) yapılmıştır. Bu vesileyle, TÜBİTAK ULAKBİM'e sağladıkları imkânlar için teşekkürü bir borç bilirim.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
GHz	Gigahertz
Hz	Hertz, bir sinyal ya da dalganın frekansı
kHz	Kilohertz
ms	Milisaniye
MB	Megabayt

Kısaltmalar	**Açıklamalar**
ADD	Ayrık Dalgacık Dönüşümü
ADÖ	Algısal Doğrusal Öngörü
AM	Akustik Model
BDM	Büyük Dil Modeli
BZS	Bağlantıcı Zamansal Sınıflandırma
DARPA	Defense Advanced Research Projects Agency (Savunma İleri Araştırma Projeleri Ajansı)
DDA	Doğrusal Diskriminant Analizi
DDİ	Doğal Dil İşleme
DDU	Düşük Dereceli Uyumlama
DM	Dil Modeli
DÖ	Derin Öğrenme
DÖK	Doğrusal Öngörücü Kodlama
DVM	Destek Vektör Makinesi
ESA	Evrişimli Sinir Ağı
GDSKT	Geniş Dağarcıklı Sürekli Konuşma Tanıma
GİB	Grafik İşlem Birimi
GKM	Gauss Karışım Modeli

Kısaltmalar	Açıklamalar
GTB	Geçitli Tekrarlayan Birim
İBSA	İleri Beslemeli Sinir Ağı
KaHO	Karakter Hata Oranı
KHO	Kelime Hata Oranı
KKDÖ	Kendi Kendine Denetimli Öğrenme
KÖK	Karşılaştırmalı Öngörücü Kodlama
MFKK	Mel Frekans Kepstral Katsayıları
MİB	Merkezi İşlem Birimi
OKT	Otomatik Konuşma Tanıma
SMM	Saklı Markov Modeli
TBA	Temel Bileşen Analizi
TDK	Türk Dil Kurumu
TRUBA	Türkiye Ulusal Bilim E-Altyapısı
TSA	Tekrarlı Sinir Ağı
TÜBİTAK	Türkiye Bilimsel Ve Teknolojik Araştırma Kurumu
UKZB	Uzun Kısa Zamanlı Bellek
ULAKBİM	Ulusal Akademik Ağ Ve Bilgi Merkezi
YSA	Yapay Sinir Ağları
YZ	Yapay Zekâ
ZGSA	Zaman Gecikmeli Sinir Ağı

1. GİRİŞ

Canlılar arasındaki sesli iletişim, insanlık tarihinden bile eski bir olgudur ancak konuşmanın, yani kelimeler kullanılarak yapılan sesli iletişimin 100 000 - 200 000 yıl önce başladığı tahmin edilmektedir [1]. Konuşma kelimesi, "ko-" (bırakmak, koymak) ve eklenen "n" ekiyle "kon-" (kendini yerleştirmek, oturmak) haline alan kökten türetilmiştir. Buna, işteşlik eki "–uş" ve adıl eki "–ma" eklenmesiyle "birlikte oturma" anlamına gelen ve aynı ortamda bulunan birden fazla kişinin sesli iletişimini ifade eden "konuşma" ortaya çıkmıştır [2]. Kökeni itibariyle işteşlik içerse de, günümüzde, tek yönlü sesli iletişimi anlatan "söylev" yerine de kullanılmaktadır.

Konuşma, insanların duygu, düşünce ve deneyimlerini birbirlerine aktarmasını sağlayan en temel iletişim yöntemlerinden biridir. Sesli iletişim kapsamında yer alan konuşma, görsel (yazılı ve yazılı olmayan) iletişim gibi günlük yaşamda büyük bir öneme sahiptir. Konuşma, herhangi bir engeli olmayan insanlar için doğuştan gelen bir yetenektir ve anadil ek bir çaba harcamandan öğrenilir. Bu nedenle, insan-makine etkileşiminde; kullanımı kolay, doğal bir yetenek olan konuşmadan faydalanılmasına yönelik çalışmalar yürütülegelmiştir [3].

Bunun sonucunda, Otomatik Konuşma Tanıma (OKT) sistemleri ortaya çıkmıştır. OKT, basitçe, konuşma sinyallerinin dijital hale getirilerek yazılı metne dönüştürülmesidir. Günümüzde; iletişim, sağlık, eğitim, otomotiv, robotik, hukuk, istihbarat vb. birçok alandaki diyalog sistemleri, sanal asistanlar, sesli arama sistemleri vb. uygulamalarda OKT kullanılmaktadır [4-5].

Bu tez çalışmasında, öncelikle Otomatik Konuşma Tanıma (OKT) sistemlerinin temel özellikleri ele alınmıştır. Daha sonra, OKT sistemlerinin tarihsel gelişimi incelenmiş, OKT konusundaki belli başlı zorluklar bahsedilmiş; OKT sistemlerinin temel mimarisi ile türleri açıklanmıştır. Ardından, ilgili çalışmalara değinilmiş ve kullanılan yöntem ve algoritmalar açıklanmıştır. Devamında, Kendi Kendine Denetimli Öğrenme (KKDÖ) yöntemini içeren güncel *Whisper* [6] uygulamasıyla yapılan deneyler ve sonuçları paylaşılmıştır. Ayrıca, modeller üzerinde ince ayar (*fine-tuning*) işlemi yürütülmesinin başarı artışına katkısı değerlendirilmiş ve elde edilen sonuçlar sunulmuştur. Son olarak, alt bileşen olarak *Whisper*

mimarisini kullanan, kısa gecikmeli konuşma tanıma uygulamaları üzerinde deneyler yürütülmüş, sonuçlar değerlendirilmiştir.

Konuşmanın genel özellikleri

Konuşma, hava akciğerden dışarı atılırken ses telleri, gırtlak ve ağızdaki diğer öğelerin (diş, dudak, çene) durumuna göre çıkan farklı frekanslardaki titreşimlerin bir araya gelmesiyle oluşur. İnsan sesindeki titreşim, frekans dağılımındaki belirgin tepe noktaları veya rezonans bölgeleri *formant* olarak adlandırılır. Bastan tize doğru, *F1'*den başlayarak *F2, F3, ...Fn* şeklinde adlandırılan çeşitli formantlar bulunur. Temel frekans olan *F0*, sesin tınısını gösterir ve formant olarak kabul edilmez. Beden yapısının ses özelliklerine yansımasını inceleyen bir araştırmada, *F0* temel ses frekansının erkekler için 119,25±16,54 Hz, kadınlar içinse 209,74±19,53 Hz aralığında olduğu belirlenmiştir [7]. Ortalama *F0* frekansı, erkekler için yaklaşık 100 Hz, kadınlar içinse yaklaşık 225 Hz civarındadır [8]. OKT sistemlerinde, sesli harflerin tanınması için genellikle *F1* ve *F2 formantlarının* yeterli olduğu kabul edilir. Sessiz harflerin tanınmasında, bu iki *formant* ile birlikte *F3 formantı* da kullanılır [8]. Yetişkin bir erkek tarafından seslendirilen sesli harflere ait örnek *F1, F2, F3 formant* değerleri Çizelge 1.1'de verildiği gibidir.

Çizelge 1.1. Sesli harflere ait örnek formant değerleri [9]

	F1 (Hz)	***F2 (Hz)***	***F3 (Hz)***
a	628,9	1259,3	2706,2
e	485,6	1834,0	2614,1
ı	537,4	1577,5	2722,0
i	286,1	2177,9	2942,7
o	467,7	1064,5	2695,4
ö	543,9	1516,7	2549,3
u	309,9	908,8	2400,9
ü	372,1	1632,7	2369,3

OKT sistemlerinde yararlanılan temel disiplinler

OKT, konuşmanın makine tarafından anlaşılması ve doğru bir şekilde metne dönüştürülmesini sağlayan bir ses işleme teknolojisi olarak tanımlanabilir. OKT çalışmalarının amacı, akustik bir sinyali hesaplamalar yoluyla kelime dizisine dönüştüren sistemler oluşturmaktır [10]. Bu bağlamda, OKT probleminin çözümü için çok disiplinli bir yaklaşım gerekmektedir. *Rabiner ve Juan*, kitaplarında [11] bu problemin çözümünde gerek duyulan disiplinleri şu şekilde sıralarlar:

- Sinyal İşleme: Konuşmaya ait analog ses verisinin, zaman serileri şeklinde dijital veriye çevrilmesi ve sese ait karakteristik özelliklerin çıkarılmasıyla ilgilenir.
- Akustik ve Fizyoloji: Konuşmanın üretimi ve algılanması ile ilgili fizyolojik süreçleri inceler.
- Örüntü Tanıma: Ses desenlerine ait özellik ölçümlerine dayanarak, karşılaştırma ve kümeleme yoluyla desenleri eşleştirmeyi sağlar.
- İletişim ve Bilgi Teorisi: Çeşitli istatistik modeller ve modern kodlama/çözümleme algoritmalarıyla uzun ancak sınırlı bir ses dizisine en iyi uyan kelime dizisini ortaya çıkarır.
- Dil Bilimleri: konuşmaya ait seslerin, ilgili dildeki söz dizim (sentaks) ve anlambilimi (semantik) bağlamında ilişkilerini ortaya koyar.
- Bilgisayar Bilimleri: Hem yazılım hem de donanım bakımında konuşma tanıma için gerekli yöntemleri geliştirir.
- Psikoloji: Konuşmanın nasıl üretildiğinin anlaşılmasına ve akustik özelliklerin daha doğru bir şekilde analiz edilmesine yardımcı olur.

OKT sistemlerinin kısa tarihçesi

1939 yılından, *Bell* laboratuvarlarında çalışan *Dudley* ve diğerleri tarafından konuşma analiz ve sentezi için önerilen sistem modeli [12, 13] OKT sistemi çalışmalarının başlangıcı kabul edilir [14]. Bununla birlikte, deneysel çalışmalar dikkate alındığında, OKT sistemlerinin ortaya çıkışı Yapay Zekâ (YZ) kavramının bilimsel temellerinin atıldığı döneme denk gelir. *Russell ve Norvig* kitaplarında, 1950 yılında yayımlanan *Turing'e* ait "Bilgi İşlem Makineleri ve Zekâ" makalesini [15] YZ çalışmalarının başlangıcı olarak göstermişlerdir [16]. OKT sistemleriyle ilgili deneysel çalışmalar da bu döneme rastlar [11]. Bilinen ilk deneysel çalışma, 1952 yılında Bell Laboratuvarlarından Davis ve

diğerlerinin geliştirdiği, tek bir konuşmacı için izole rakam tanımayı sağlayan sistemdir [17]. 1950-1960 yılları arasında yapılan benzer çalışmalarda, genel olarak fonem, tek harf ya da heceyi ayırt etmeye yönelik örüntü tanıma sistemleri oluşturulması hedeflenmiştir [18-20].

1960-1970 arasındaki dönemde, Japonya'da geliştirilen donanım tabanlı 3 sistem dikkat çekmektedir [21-23]. Bunlarla birlikte, *IBM* firmasının *SHOEBOX* yazılımı [24], *RCA* Laboratuvarlarında *Martin'*in çalışmaları [25], dinamik programlama yöntemlerinin kullanıldığı *Vintsyuk'*un çalışması [26] bu dönemde öne çıkan çalışmalardır.

1970-1980 yılları arası, OKT sistemlerinde, örüntü tanıma ve dinamik programlamanın yanında *Viterbi* algoritması [27] gibi istatistik tabanlı yaklaşımların ortaya çıktığı zamanlar olmuştur. *Itakura*'nın Doğrusal Öngörücü Kodlama (DÖK) tabanlı çalışması [28], basit komutların algılanmasını sağlayan *VIP-100* yazılımı, *DARPA SUR* programı kapsamında geliştirilen *Hearsay* ve *HWIN* yazılımları, ayrıca *CMU* firmasının *HARPY* yazılımları ilgili dönemdeki önemli çalışmalar olarak kayda geçmiştir [29-30].

1980-1990'li yıllara arasında, istatistiksel modeller yaygınlaşırken, Yapay Sinir Ağları (YSA) ile ilgili çalışmalar da [31] ortaya çıkmaya başlamıştır. *Ferguson'un* [32] , *Wilpon* ve diğerlerinin [33], *Levinson* ve diğerlerinin [34] Saklı Markov Modeli (SMM) tabanlı çalışmaları dönem içinde göze çarpan çalışmalardır.

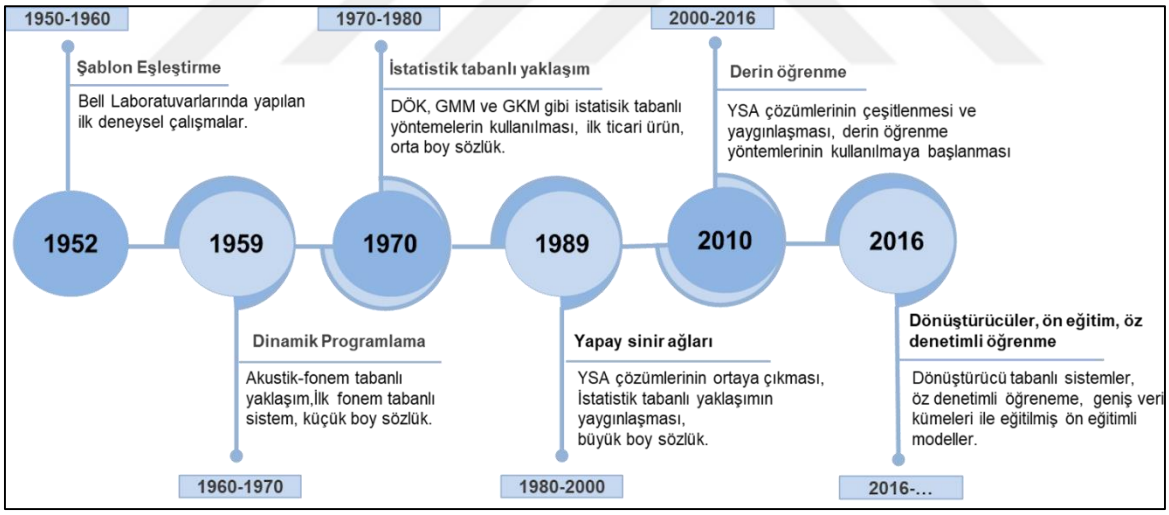
1990-2000 yılları arasında öne çıkan gelişmelerden bazıları: AT&T Ses Tanıma Çağrı İşleme (*VRCP, Voice Recognition Call Processing*) çözümü [35] ve Cambridge Üniversitesi tarafından geliştirilen Saklı Markov Modeli Araç Kitidir (*HMM Tool Kit*) [36]. Bu dönemde, Destek Vektör Makinesi (DVM) tabanlı çözümlerin de geliştirildiği görülmektedir [14].

2000 yılından sonraki dönemde, olgunlaşan istatistik tabanlı sistemlerin yanında YSA tabanlı sistemler gelişimini sürdürmüştür. 2010 yılında *Android* yazılımına "*Voice Search*" özelliği eklenmiş, *IOS* ile bütünleştirilmiş *Siri* 2011 yılında kullanıma açılmıştır. OKT sistemleri için *DARPA* programı kapsamında, *EARS* ve *GALE* veri kümeleri oluşturulmuştur [37].

2010 sonrasında, Derin Öğrenmenin (DÖ) ortaya çıkışıyla birlikte OKT sistemlerinde YZ kullanımı artmıştır. *Graves* ve diğerlerinin Bağlantıcı Zamansal Sınıflandırma (BZS) ile

birlikte Tekrarlı Sinir Ağı (TSA) ve Uzun Kısa Süreli Bellek (UKZB) temelli [38-39], *Chan* ve diğerlerinin *LAS* (*Listen, Attend, Speech*) adını verdikleri mekanizmayı kullanan İki Yönlü UKZB temelli [40], *Zhang* ve diğerlerinin *LAS*, evrişim mekanizması ve artık ağ temelli çalışmaları [41] dönem içindeki önemli gelişmelerdir. Bu dönemdeki gelişmelerden bir başkası, *Vaswani* ve diğerlerinin OKT sistemlerinde dönüştürücü mekanizması kullanmasıdır [42]. 2019 yılında, *Schneider* ve diğerleri *Wav2vec* modelini ortaya çıkarmışlardır [43]. *Baevski* ve diğerleri, *Wav2vec'*in geliştirilmiş hali olan ve öncülü gibi temsil öğrenme [44] temelinde çalışan *Wav2vec 2.0* adını verdikleri modeli geliştirmişlerdir [45]. 2020 yılında *Gulati* ve diğerleri Evrişimli Sinir Ağı (ESA) ve dönüştürücü mekanizmalarını içeren *Conformer* modelini kullanan OKT sistemini tanıtmışlardır [46]. 2022 yılında, *OpenAI* firmasından *Radford* ve diğerleri tarafından *Whisper* adı verilen OKT sistemi geliştirilmiştir. Dönem içinde *IBM, Microsoft, Google* ve *Amazon* gibi büyük firmalar da ürettikleri gelişmiş OKT sistemlerini piyasaya sürmüştür.

Bu bağlamda, OKT sistemlerinin tarihsel gelişim süreci Şekil 1.1'deki gibi özetlenebilir.



Şekil 1.1. OKT sistemlerinin gelişim süreci [14]

OKT sistemleriyle ilgili zorluklar

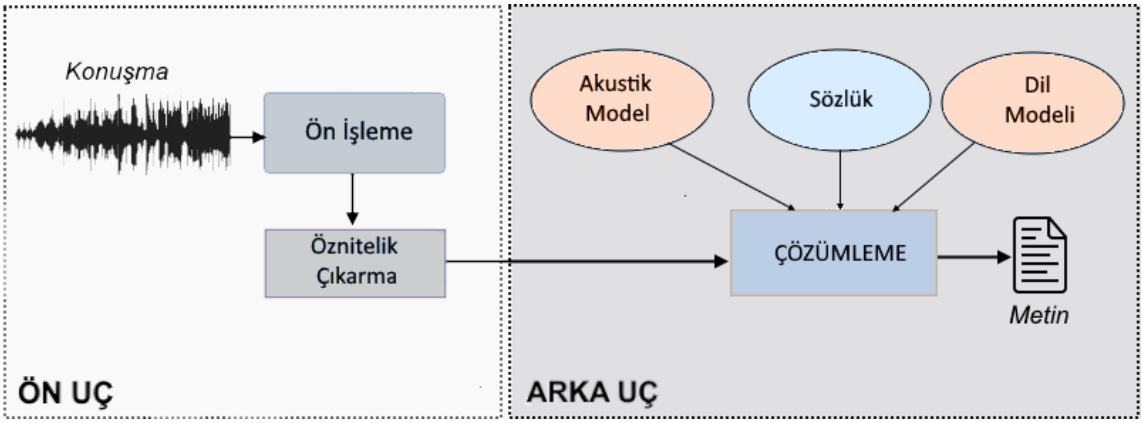
İnsanlar, konuşmayı metne dönüştürürken, geçmiş deneyim ve dilbilgilerinden yararlanırlar ancak OKT sistemleri için konuşma, sadece bir ses sinyalidir [47]. OKT sistemlerinde, ton ve beden dili gibi önemli iletişim unsurları, ses sinyalinin içeriğinde yer almadığı için kullanılamaz. Bu eksiklik istatistiksel modellerle giderilmeye çalışılır.

Konuşmacıya göre değişen lehçe, şive, aksan, telaffuz gibi unsurlar OKT sistemlerinin doğru çalışmasını etkileyen önemli zorluklar arasında yer alır. Konuşmacının cinsiyeti, konuşma tarzında farklılaşmaya neden olduğu için konuşmacı bağımsız OKT sistemleri açısından süreci güçleştirir. Kişiden kişiye ya da aynı kişi için duygu durumu, ortam, muhatap vb. unsurlara göre konuşma şekli değişebilir. Bir kelimenin akustik özellikleri, herhangi bir durum ya da ortam değişikliği olmadığı, seslendiren sabit kaldığı hallerde bile değişebilir. Konuşma temposunda, vurgularda meydana gelen değişimler, duraksama ve gereksiz tekrarlar başka bir zorluk olarak ortaya çıkar.

Bunların dışında, OKT sistemlerinin konuşma dili ve yazım dili farklılıklarıyla da başa çıkması gerekir. Çevresel faktörler ya da ses kayıt cihazlarından kaynaklanan konuşma dışı gürültü de OKT sistemlerinin başa çıkması gereken diğer bir sorundur.

OKT sistemlerinin temel mimarisi

OKT sistemlerinin temel mimarisi Şekil 1.2'de gösterildiği gibi ön uç (*front-end*) ve arka uç (*back-end*) olmak üzere iki ana bölümden oluşur. Ön uçta, konuşma sinyalinin ön işleme tabi tutulması ve öznitelik çıkarma işlemleri yürütülür. Ön işleme aşamasında, gürültü azaltma ve sinyal kalitesini iyileştirmek için ön vurgu filtreleri uygulanır [48]. Sonraki aşamada ön işlemeden geçirilen konuşma sinyaline ait perde, formant, spektral zarf, ton, genlik vb. özellik vektörleri elde edilir. Bu işlem için genellikle Mel Frekans Kepstral Katsayıları (MFKK), Doğrusal Öngörücü Kodlama Katsayıları (DÖK) gibi yöntemler kullanılır. Yaygın olmamakla beraber, Algısal Doğrusal Öngörü (ADÖ), Ayrık Dalgacık Dönüşümü (ADD), Doğrusal Diskriminant Analizi (DDA), Temel Bileşen Analizi (TBA) gibi yöntemlerden de faydalanılır.



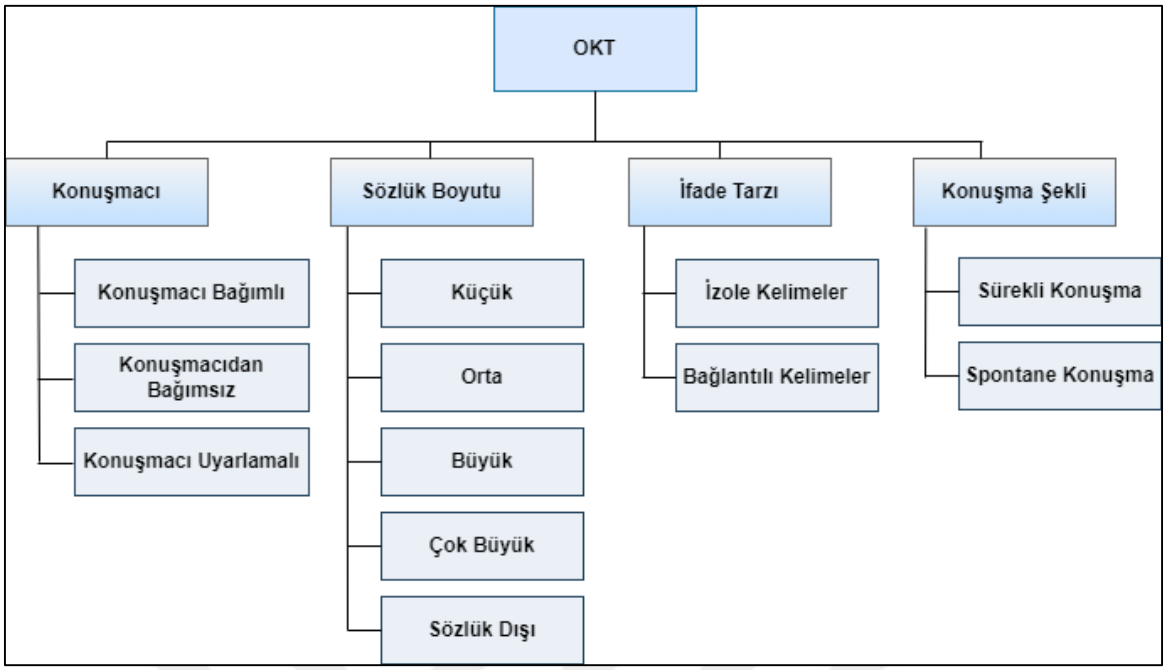
Şekil 1.2. OKT sistemlerinin temel mimarisi [49]

Öznitelik çıkarma işlemi sonrasında, konuşmanın metne dönüştürülmesi için çözümleme aşamasına geçilir. Akustik Model (AM) kullanılarak konuşmaya ait ses birimleri (*fonemler*) elde edilirken, Sözlük (Okunuş Sözlüğü) kullanılarak seslere karşılık gelen kelimeler elde edilir. Akustik Model için SMM, GKM, DVM ve YSA tabanlı yöntemler kullanılır. Dil Modelinde (DM) istatistik modeller, YSA vb. yöntemler kullanılarak, ilgili dil için en uygun olası kelime dizisi elde edilmeye çalışılır. DM çözümü olarak GKM, DVM, YSA gibi yöntemler uygulanır. *Viterbi* algoritmasına benzer şekilde çalışan ve *n* adet en olası dizilimi gösteren *n-best* ya da *n* adet kelimenin art arda bulunma ihtimali üzerine çalışan *n-gram* yaygın kullanılan DM çözümleridir [50]. OKT tasarımında, DM kullanımı genellikle sisteminin performansını artırır [10].

OKT sistemlerinin temel mimarisi Şekil 1.2'de gösterilen yapıya sahip olsa da, güncel çözümler, ayrıştırılmış öznitelik çıkarma işlemi, Akustik Model ve Dil Modeli olmadan çalışırlar. Günümüzde kullanılan uçtan uca mimarilerin pek çoğunda, bu işlemler bütünlük halinde tek aşamada gerçekleştirilir.

OKT sistemi türleri

OKT sistemleri konuşmacı, kullanılan sözlük boyutu, ifade tarzı ve konuşma şekli üzerinden Şekil 1.3'teki gibi sınıflandırılabilir [50].



Şekil 1.3. OKT sistemlerinin sınıflandırılması [50]

Konuşmacı bağımlı sistemler, tek bir konuşmacı üzerine kurulmuş sistemlerdir. Konuşmacıdan bağımsız sistemler ise farklı kişilere ait konuşma verileri kullanılarak eğitim ve test işlemleri yürütülür. Konuşmacı uyarlamalı sistemler, konuşmacı bağımlı ve konuşmacıdan bağımsız sistemlerin arasında yer alır ve konuşmacı eklendiğinde, yeni konuşma kalıplarını öğrenebilecek şekilde eğitilir [47-50].

Küçük boy sözlük 10-99 arası kelime içerir. 100-999 arası kelime içeren sözlükler orta boy sözlük olarak anılır. 1000-9999 arası kelime içeren sözlükler büyük, 10000'den fazla kelime haznesine sahip sözlükler çok büyük sözlük olarak adlandırılır. Sözlükte olmayan kelimeler, sözlük dışı olarak tanımlanır.

İzole kelime sistemlerinde, sistem her seferinde sadece bir kelime kabul eder. Bağlantılı kelime sistemlerinde, birden fazla kelime, aralarda durak verilerek seslendirilir [47-50].

Sürekli konuşma sistemleri, herhangi bir ek duraksama olmadan okunan metinlerin tanınmasına yöneliktir. Spontane konuşma sistemlerindeyse, metinden bağımsız yapılan doğal konuşma tanınmaya çalışılır [47-50].

2. İLGİLİ ÇALIŞMALAR

1939 yılında *Dudley* ve diğerleri tarafından yapılan bir dizi çalışma ve önerilen model [12, 13] teorik OKT çalışmalarının başlangıcı kabul edilir [14]. Öte yandan, OKT üzerine deneysel çalışmaların, 1952 yılında *Bell* Laboratuvarlarından *Davis* ve diğerlerinin geliştirdiği, tek konuşmacılı izole rakam tanıma sistemiyle başladığı düşünülür [17]. Sonrasında, 1956 yılında *RCA* Laboratuvarlarından *Olson ve Belar*, önceki çalışmaya benzer şekilde, sesli harflerdeki geçişlerin spektral ölçümüne dayanan, tek konuşmacı tarafından seslendirilen 10 tek heceli kelimenin tanınması üzerine bir çalışma yapmışlardır [18]. 1959 yılında *Fry ve Denes,* seslendirilen dört sesli ve dokuz sessiz harfi yakalamak için *fonem* tanıma sistemi geliştirmişlerdir [19]. Aynı yılda *Forgie ve Forgie* tarafından yapılan benzer bir çalışmada, "b" ve "t" harfleri arasına yerleştirilen 10 sesli harfin konuşmacıdan bağımsız olarak tahmin edilmesi amaçlanmıştır [20].

1960'lara gelindiğinde, Japonya'da geliştirilen donanım tabanlı üç sistem dikkat çekmektedir [21-23]. Bunun yanında, *IBM* firmasına ait, 0 ile 9 arası rakamları içeren 16 ayrık kelimeyi tanımayı sağlayan *SHOEBOX* yazılımı da dönemin önemli çalışmalarından biridir [24]. 1960 yılında *RCA* Laboratuvarlarında *Martin* tarafından yapılan bir çalışmada, konuşmanın başlangıcı ve bitişini belirleyen bir sistem tasarlanmıştır [25]. Bu tasarımda kullanılan zaman normalizasyonu tekniği, konuşmanın zaman düzlemindeki düzensizlikleriyle başa çıkabilmek için gerçekçi bir çözüm sunmuştur [11]. Aynı dönemde, *Vintsyuk*, konuşma ifadelerinde zaman uyumunu sağlamak için dinamik programlama yöntemlerinin kullanılmasını önermiştir [26].

1970'li yıllar, OKT sistemlerinde, örüntü tanıma ve dinamik programlamanın yanında istatistik tabanlı yaklaşımların ortaya çıktığı zamanlar olmuştur. Bu bağlamda *Itakura,* son teknoloji sistemlerin bazılarında da faydalanılan DÖK yönetimini kullanmıştır [28]. 1970'li yılların sonundan itibaren, dinamik programlama ve istatistik tabanlı *Viterbi* algoritması [27] gibi yöntemler OKT sistemlerinin vazgeçilmez bir parçası haline gelmiştir [14]. *Threshold Techonology* firması, basit komutların algılanmasını sağlayan ilk ticari ürün *VIP-100* yazılımını bu dönemde piyasaya sürülmüştür. 1970 yılında başlatılan *DARPA SUR* programı kapsamında *Hearsay* ve *HWIN* yazılımları geliştirilmiş ancak bu yazılımların performansı programda hedeflenenin altında kalmıştır [12-24]. Dönem içindeki önemli olaylardan bir

başkası, *CMU* firması tarafından 1000 kelimelik tanıma sözlüğüne sahip *HARPY* adı verilen yazılımın geliştirilmesidir [30].

1980'li yıllara gelindiğinde, OKT sistemlerinde istatistiksel modeller yaygınlaşmaya başlamıştır. Bilhassa SMM bu konuda önemli bir teknoloji olarak ortaya çıkmıştır. *Ferguson'un* çalışması [32], *Wilpon* ve diğerlerinin çalışması [33] ve *Levinson* ve diğerlerinin çalışması [34] öne çıkan SMM tabanlı OKT sistemleri olmuştur. İlgili dönemde, YSA tabanlı çözümlerin de geliştirilmeye başlandığı görülmektedir [31]. Bunun yanında, 1980'liler *n-gram* DM'nin ortaya çıktığı ve OKT sistemlerinin bir parçası haline geldiği zamanlardır.

1990'lı yıllarda, özellikle "çağrı merkezi" alanındaki gelişmeler OKT sistemlerine olan ihtiyacın artmasına sebep olmuştur [11]. 1992 yılında devreye alınan AT&T Ses Tanıma Çağrı İşleme (*VRCP, Voice Recognition Call Processing*) çözümü, 1,2 milyar sesli işlemin OKT teknolojisiyle yönlendirilmesi sağlanmıştır [35]. Bu dönemde, minimum sınıflandırma veya ampirik hata kavramı daha sonra bir dizi tekniğin ortaya çıkmasını sağlamıştır. Bunların arasında yer alan ayırt edici eğitim ve çekirdek tabanlı DVM gibi yöntemler popüler çalışma konuları haline gelmiştir [14]. 1990'lı yıllarda yaşanan başka bir önemli gelişme, sonrasında pek çok yazılıma adapte edilen Saklı Markov Modeli Araç Kitinin (*HMM Tool Kit*) Cambridge Üniversitesi tarafından geliştirilip kullanıma açılmasıdır [36].

2000'li yıllara gelindiğinde, istatistik tabanlı sistemler olgunlaşırken, YSA tabanlı sistemler gelişimini sürdürmüştür. Akıllı telefonlar, 2000'li yıllarda OKT sistemlerinin gelişiminde önemli bir tetikleyici olmuştur. *Google* firması 2010 yılında *Android* yazılımına "*Voice Search*" özelliğini eklemiştir. *Apple* firması tarafından *IOS* sistemlere eklenen OKT sistemi *Siri* 2011 yılında kullanıma açılmıştır. OKT modelleri geliştirilirken, bir yandan da ihtiyaç duyulan veri kümelerinin oluşturulmasıyla ilgili çalışmalar yürütülmüştür. *DARPA'*nın desteklediği programlardan *EARS* ve *GALE* bu dönemde yürütülen iki önemli çalışmadır. *EARS* kapsamında, 500 konuşmacılı 260 saatlik telefon ve *GALE* kapsamında Arapça ile Mandarin (Standart Çince) dillerinde haber bülteni konuşma veri kümeleri oluşturulmuştur [37].

DÖ'nün ortaya çıkmasıyla birlikte, OKT sistemlerinde istatistiksel modeller yerine gelişmiş YZ araçlarının kullanımı artmıştır. *Graves* ve diğerleri, BZS'nin temel ilkelerini ve TSA

tabanlı otomatik konuşma tanıma alanında nasıl kullanılabileceğini açıklamışlardır [38]. 2013 yılında yayımlanan bir çalışmada, BZS ve TSA türevi UKZB ve İki Yönlü UKZB kullanılarak konuşma tanıma uygulaması yapılmıştır [39]. 2016 yılına gelindiğinde, *Chan* ve diğerleri *LAS* (*Listen, Attend, Speech*) adını verdikleri yöntemi kullanan, İki Yönlü UKZB ve dikkat mekanizmasını içeren çalışmalarını yayımlamışlardır [40]. 2017 yılında *Zhang* ve diğerleri çalışmalarında, evrişim mekanizmasını ve artık ağı kullanarak *LAS* tabanlı uçtan uca bir model geliştirmişlerdir [41]. Aynı yıl yayımlanan *Vaswani* ve diğerlerinin çalışmasında, OKT sistemlerinde dönüştürücü mekanizması kullanımı örneklenmiştir [42].

2019 yılında *Facebook* Yapay Zekâ Araştırmaları birimi bünyesinde *Schneider* ve diğerleri tarafından *Wav2vec* adı verilen mimari geliştirilmiştir [43]. Bu çalışmada, *Oord* ve diğerleri tarafından genel hatları açıklanan Temsil Öğrenme [44] kullanılarak başarılı sonuçlar elde edilmiştir. Bu modelin başarısının ardından, bu sefer *Baevski* liderliğindeki ekip tarafından *Wav2vec 2.0* adı verilen uçtan uca konuşma tanıma sistemi tasarlanmıştır [45]. 2020 yılında *Gulati* ve diğerleri ESA ve dönüştürücü mekanizmalarının bir arada kullanılmasıyla başarılı sonuçlar elde ettikleri *Conformer* adı verilen sistemi tanıtmışlardır [46]. 2022 yılında, *OpenAI* firması bünyesinde çalışan *Radford* ve diğerleri tarafından, *Vaswani* ve diğerlerinin çalışmasını [42] temel alan, çok dilli, çok görevli *Whisper* mimarisi geliştirilmiştir. Son olarak, 2023 yılında *Google* firmasından *Wang* ve diğerleri tarafından *USM (Universal Speech Model*) [51] adı verilen, *Conformer* ve dönüştürücü mekanizmalarını içeren, ön eğitim, yarı denetimli öğrenme ile kendi kendine denetimli öğrenme tekniklerini kullanan yeni bir model ortaya konulmuştur.

Bu dönem içindeki önemli gelişmelerden bir başkası, *IBM, Microsoft, Google ve Amazon* gibi büyük firmaların geliştirdikleri OKT sistemlerini piyasaya sürmeleridir.

- IBM Watson Speech to Text: 2017 yılında piyasaya sürülmüştür. Bulut tabanlı olarak konuşmayı metne dönüştürme ve çeviri hizmetlerini yerine getiren bir üründür.
- Microsoft Azure Speech Services: 2016 yılında piyasaya sürülmüştür. *Microsoft Azure* platformunda barındırılan *Microsoft Azure Speech Services*, konuşmayı metne dönüştürme ve çeviri hizmetlerini sunar.
- Google Cloud Speech-to-Text: 2016 yılında piyasaya sürülmüştür. *Google Cloud Speech-to-Text*, konuşmayı gerçek zamanlı metne dönüştürmeyi sağlayan aynı zamanda toplu işlemeye izin veren bir üründür. *Google Cloud* Platform platformunda barındırılmaktadır.

- Amazon Transcribe: 2017 yılında piyasaya sürülmüştür. Diğer ürünlerle benzer özelliklere sahiptir ve *Amazon Web Services (AWS)* platformu üzerinde hizmet vermektedir.

OKT bağlamında, Türkçe ve benzeri sondan eklemeli diller, diğer dillere nazaran ek zorluklar içerir [5]. Buna rağmen, OKT konusunda az sayıda da olsa Türkçe çalışmalar mevcuttur. *Artuner*'e ait, kendi kendine örgütlenen öznitelik haritalarının (*self-organized feature maps*) kullanıldığı, Türkçe özgün ses birimi kod defterine sahip, konuşmacı bağımlı sistem öncü çalışmalardan biridir. Bunun yanında, diğer erken dönem Türkçe çalışmalarda genel olarak akustik modelin iyileştirilmesi konusuna odaklanılmıştır [52-55]. Sonraki yıllarda, SMM tabanlı ve DM kullanılan OKT çözümlerindeki artış dikkat çekmektedir [56-61]. Takip eden dönemde, özellikle YSA tabanlı OKT çalışmaları yaygınlaşırken, Geniş Dağarcıklı Sürekli Konuşma Tanıma (GDSKT) çözümlerine odaklanılmıştır [62-64]. Gelişen teknolojiye koşut olarak 2010 yılı sonrasında DÖ kavramının ortaya çıkışıyla birlikte, OKT sistemlerinde DÖ kullanımı yaygınlaşmaya, uçtan uça modeller geliştirilmeye başlanmıştır [5, 8, 65-69].

Bunların dışında, *Haznedaroğlu* ve diğerleri Sphinx III kullanarak %22,5 KHO'ya ulaşan bir model geliştirmişlerdir [70]. *Çetinkaya* ve diğerleri GKM'yi ve Zaman Gecikmeli Sinir Ağını (ZGSA) kullanarak Türkçe Haber Bülteni veri kümesi üzerinde karşılaştırmalı bir çalışma yürütmüşlerdir [71]. *Tombaloğlu* ve *Erdem* DVM ve MFKK tabanlı bir model geliştirerek deneysel çalışmalar yapmışlardır [72]. *Arslan* ve *Barışçı*, BZS kullanılarak UKZB tabanlı OKT modeli üzerinden çeşitli iyileştirme algoritmalarının etkinliğini ölçmüşlerdir. *Oyucu* ve *Polat* istatistik ve İleri Beslemeli Sinir Ağı (İBSA) temelli DM'lerin model KHO üzerindeki etkinliği üzerinde çalışmışlardır [73]. *Yalman* ve *Tüfekçi*; TSA, GTB, UKZB ile birlikte BZS kullanılarak geliştirilen modellere üzerinde karşılaştırmalı bir çalışma yürütmüşlerdir [74]. Güncel sayılabilecek çalışmalardan biri, *Safaya* ve *Erzin* tarafından yapılmıştır. Bu çalışmada HUBERT [75] mimarisi 6.500 saatlik konuşma verisi ile ön eğitime tabi tutulduktan sonra çeşitli veri kümeleri üzerinde testler yürütülmüş ve karşılaştırmalı sonuçlar paylaşılmıştır. *Mercan* ve diğerleri yaptıkları çalışmada *Wav2vec 2.0* [45] ve *Whisper* [6] OKT sistemlerinin karşılaştırmışlardır [76].

3. MATERYAL VE METOTLAR

Bu bölümde konuşma sinyalinin makine öğrenmesi yöntemleri ile işlenebilmesi için uygulanan ön işleme ve öznitelik çıkarma tekniklerinden bahsedilmiştir. Ardından, kullanılan konuşma veri kümeleri tanıtılmıştır. OKT işleminde kullanılan makine öğrenmesi yöntemlerinden bahsedildikten sonra OKT araçları tanıtılmış ve OKT sistemlerinin başarı ölçümünde kullanılan ölçütler açıklanmıştır.

3.1. Konuşma Sinyalinin İşlenmesi/Ön işleme

Konuşma sinyalinin, OKT sistemlerindeki çeşitli algoritmalarla işlenebilmesi için ön işleme tabi tutularak dijital hale dönüştürülmesi gerekir. Daha sonra, bu işlem sonucunda elde edilen veriden konuşmaya ait özellikler çıkarılır. Bu bölümde, ses verisi üzerinde yapılan ön işlemler ve konuşma özelliklerinin çıkarılmasında kullanılan teknikler anlatılmıştır.

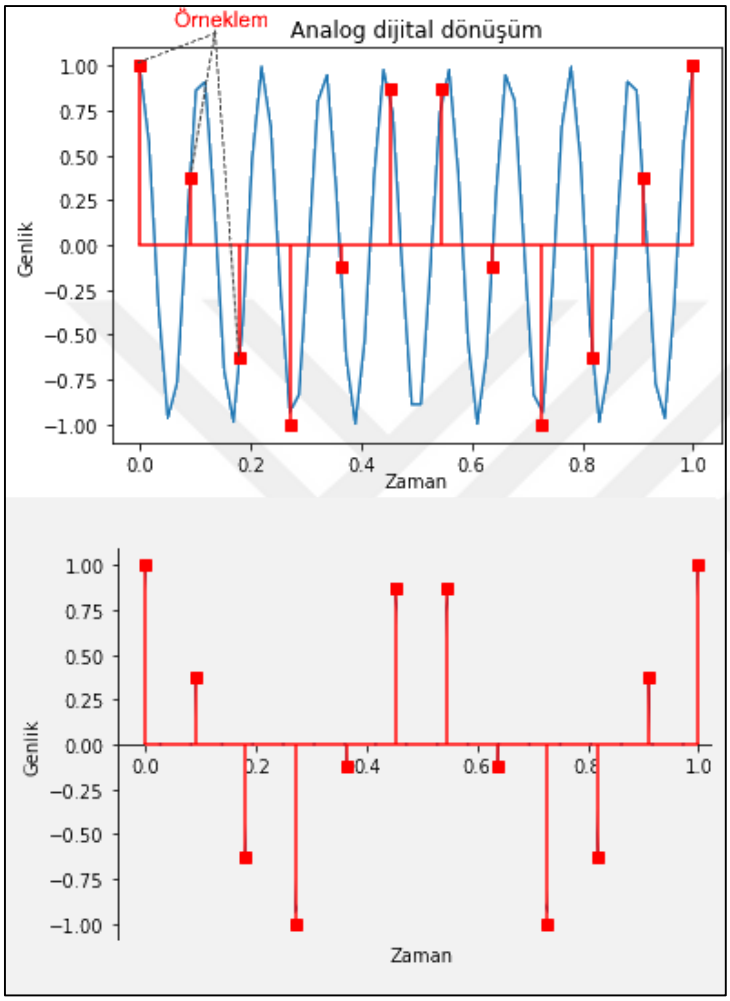
3.1.1. Analog dijital dönüşüm

Konuşmanın, makine öğrenmesi yöntemleriyle işlenebilmesi için, öncelikle analog ses sinyallerinin sayısal hale dönüştürülmesi gerekir. Bu dönüştürme işlemi, *Hertz* cinsinden belirli bir örneklem frekansına (*sampling frequency*) göre gerçekleştirilir.

Nyquist hızı (*Nyquist Rate*), bir analog sinyalin doğru bir şekilde dijital formata dönüştürülebilmesi için gerekli olan en düşük örneklem frekansını ifade eder. *Nyquist* hızı, sinyalin en yüksek frekans bileşeniyle ilişkilidir. Frekans örneklemede, *Shannon-Nyquist* teoremi olarak bilinen, "bir sinyalin örneklenebilmesi için, sinyalin en yüksek frekansının iki katı veya daha fazla bir örneklem frekansına ihtiyaç duyulur" kuralına uyulur. Aksi halde, örtüşme (*alising*) denilen frekansların karışması durumu oluşur [11].

Konuşma verisinin işlenmesi için kullanılacak en düşük ve uygun örneklem frekansı, insan konuşmasının en yüksek frekans bileşenlerini doğru bir şekilde temsil edecek kadar yüksek olmalıdır. İnsan konuşmasının en yüksek frekansı yaklaşık 4 kHz civarındadır [11]. Dolayısıyla, *Nyquist-Shannon* örneklem teoremi bağlamında değerlendirildiğinde, insan konuşmasını doğru bir şekilde temsil etmek için kullanılacak en düşük örneklem frekansı 8

kHz'dir. Öte yandan, daha iyi ses kalitesi ve konuşmanın daha doğru temsili için genellikle 16 kHz, 44,1 kHz, 48 kHz gibi daha yüksek örneklem frekansları tercih edilir. Şekil 3.1'de analog bir sinyalin 12 Hz örneklem frekansında dijital sinyale dönüştürülmesi örneklenmiştir.



Şekil 3.1. Analog sinyalin dijitale dönüştürülmesi

3.1.2. Önvurgu filtreleme (Pre-emphasize filtering)

Önvurgu filtreleme, ses sinyallerinin özelliklerini daha iyi çıkarabilmek amacıyla uygulanan bir ön işleme tekniğidir. Önvurgu filtresi, düşük frekanslı bileşenleri vurgulamak için tasarlanmıştır. Bu filtre, ses sinyalinin düşük frekans bileşenlerini belirginleştirirken yüksek frekans bileşenlerini bastırmayı amaçlar. Bu şekilde, özellikle konuşma işleme uygulamalarında kullanılan önvurgu filtresi, düşük frekanslı ses özelliklerini daha belirgin

hale getirerek konuşmanın analiz edilmesini kolaylaştırır [10]. Önvurgu işlemi sonucunda oluşan sinyal Eş. 3.1'deki gibi hesaplanır.

$$y(n) = x(t) - \alpha x(t-1) \tag{3.1}$$

Eşitlikte $x(t)$, t anındaki sinyali; $x(t-1)$, önceki sinyali; α ise filtreleme katsayısını gösterir. α için genellikle 0,90 ile 0,95 arasında sabit bir değer kullanılır [77].

3.1.3. Çerçevelere bölme (*Frame blocking*)

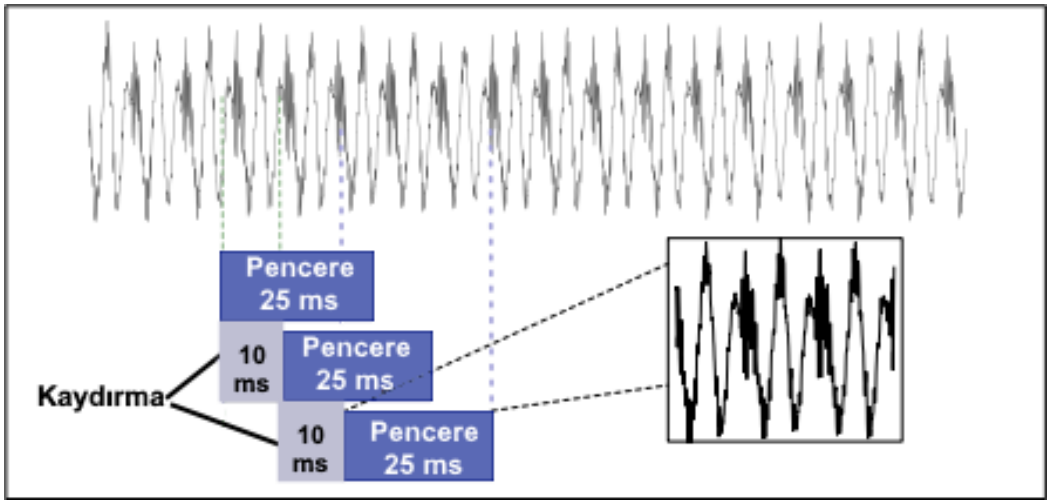
Konuşma işleme alanında yaygın olarak kullanılan bu yöntemde, ses sinyali küçük zaman dilimlerine bölünür. Çerçevelere bölmeyle, sürekli bir ses verisi küçük parçalara ayrılır. Böylece, her parça, üzerinde işlem yapılabilir hale gelir [11].

Art arda çerçevelerin belirli kısımlara örtüşecek şekilde çerçeveleme işlemi yapılabilir. Çerçeve Örtüşmesi (*frame overlap*) diye de adlandırılan bu işlemle ardışık çerçevelerin kısmen aynı veriyi paylaşması sağlanır.

Örneğin, 200 ms'lik bir çerçeve için ardışık iki çerçeve arasında 100 ms'lik bir örtüşme olduğunda, art arda çerçevelerden ilkinin son ve ikincinin ilk yarısı aynı olacaktır. Bu işlemle, çerçeveler arasında sürekli bir geçiş sağlanır. Örtüşme, keskin geçişleri yumuşatarak daha iyi frekans analizi yapabilmeyi sağlar.

3.1.4. Pencereleme (*Windowing*)

Sayısallaştırılmış ses verisinin niceliksel temsilinden spektral özelliklerin çıkarılması gerekir [10]. Bu bağlamda, gürültüyü azaltmak ve öznitelik çıkarma fonksiyonlarının performansını artırmak için ses verisi daha küçük, örtüşen pencerelere ayrılır. Pencereleme işleminde genellikle 20-30 ms'lik pencere boyu 10 ms civarında örtüşme/kaydırma kullanılır. Şekil 3.2'de pencereleme işlemi örneklenmiştir [10].



Şekil 3.2. Ses verisinin pencerelere bölünmesi [10]

Pencereleme fonksiyonu $w[n]$, ses verisi $s[n]$ olmak üzere $n.$ pencereyi temsil eden $y(n)$ Eş. 3.2'de gösterildiği gibi bulunur [10].

$$y(n) = w[n]s[n] \tag{3.2}$$

L uzunluğundaki bir pencere için dikdörtgen pencereleme (*rectangular windowing*) fonksiyonu $w[n]$ Eş. 3.3'te gösterildiği gibi elde edilir.

$$w[n] = \begin{cases} 1 & ,0 \leq n \leq L-1 \; ise \\ 0 & , \quad değilse \end{cases} \tag{3.3}$$

Bu yöntem dışında *Hanning, Hamming, Bartlett* ve *Blackman* vb. pencereleme yöntemleri de mevcuttur. Dikdörtgen pencereleme, kenarlardaki sinyalde ani kesilmelerden dolayı, öznitelik çıkarımında kullanılan *Fourier* işleminde sorun yaratır [10]. Bu nedenle, OKT sistemlerinde genellikle *Hamming* yöntemi kullanılır. *Hamming* yöntemin ait pencereleme fonksiyonu Eş. 3.4'teki gibi hesaplanır.

$$w[n] = \begin{cases} 0{,}54 - 0{,}46\cos\left(\frac{2\pi n}{L}\right) & ,0 \leq n \leq L-1 \; ise \\ 0 & ,değilse \end{cases} \tag{3.4}$$

3.2. Konuşmadan Öznitelik Çıkarma

Wav2vec gibi ayrıca özelliklerin/özniteliklerin çıkarılmasına ihtiyaç duyulmayan mimariler dışında, pek çok geleneksel OKT sisteminde ses verisi doğrudan kullanılmaz. Özniteliklerin elde edilmesinde MFKK, DÖK başta olmak üzere ADÖ, ADD, Doğrusal Diskriminant Analizi (DDA), TBA vb. yöntemlerden faydalanılır.

3.2.1. Mel frekans kepstral katsayıları (MFKK)

MFKK ses sinyallerinden öznitelik çıkarmak için kullanılan bir yöntemdir. Yöntemde, insan işitme sisteminin sesleri algılama şekli taklit edilir. İnsan kulağı tarafından, daha düşük frekanstaki sesler doğrusal olarak algılanırken, daha yüksek frekanstaki sesler logaritmik olarak algılanır [78].

MFKK işlem adımları aşağıdaki gibidir:

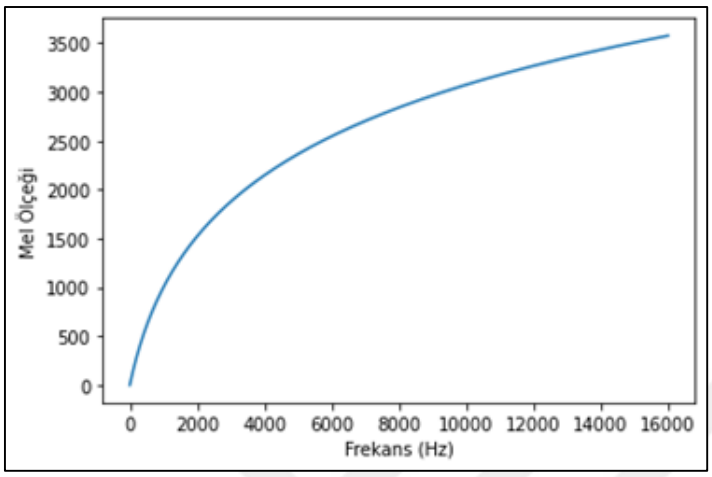
- Analog-dijital dönüşüm (ön işleme)
- Ön vurgulama (ön işleme)
- Çerçevelere bölme (ön işleme)
- Hamming pencereleme
- Hızlı Fourier Dönüşümü (HFD)
- Mel Filtre Bankasının Uygulanması
- Logaritmik Dönüşüm
- Kesikli Kosinüs Dönüşümü (KKD)

Ön işleme sonrası pencerelenmiş ses verisini, zaman alanından frekans alanına geçirmek için HFD kullanılır. Frekans bileşeni sayısı N olmak üzere k. frekans bileşeni Eş. 3.5 doğrultusunda hesaplanır [10].

$$X_k = \sum_{n=0}^{N-1} x_m e^{-i2\pi kn/N}, 0 \leq k \leq N-1 \quad (3.5)$$

Daha sonra elde edilen frekanslardan, Eş. 3.6 doğrultusunda, Şekil 3.3'te örneklenen Mel Ölçekleri elde edilir [10].

$$M(f) = 2595 \cdot log_{10}\,(1 + f/700) \tag{3.6}$$



Şekil 3.3. Frekanslara denk gelen Mel Ölçekleri grafiği

Logaritmik dönüşüm ve normalizasyon işlemi uygulanan veri kesikli kosinüs dönüşümüne tabi tutulur ve *Mel* frekans katsayıları elde edilir. Birden fazla KKD hesaplama formülü mevcuttur. Eş. 3.7'de 2. tipteki KKD hesaplaması örneklenmiştir [79].

$$Ci = \sum_{j=1}^{N-1} m_j \cos(\frac{\pi i}{N}(j + 0{,}5)) \tag{3.7}$$

Eşitlikte, N toplam frekans bileşeni sayısını; $m_j,\ j.$ Mel frekans ölçeğini göstermektedir.

3.2.2. Doğrusal öngörücü kodlama katsayıları (DÖKK)

Doğrusal Öngörücü Kodlama; konuşma sentezleme, konuşmacı tanıma, OKT vb. alanlarda kullanılan bir yöntemdir ve ses verisinden öznitelik çıkarmaya yarar. Seri halindeki ses verisine ait bir örneğin, kendisinden önceki örneklerden öngörülebileceği ilkesine göre çalışır [10]. Eş. 3.8'de yapılan işlemlerle α_k katsayıları bulunmaya çalışılır [80].

$$\dot{s}\,(n) \;=\; \sum_{k=1}^{p} \alpha_k \cdot s\,(n \;-\; k) \tag{3.8}$$

Eşitlikte, n. ses örneği, kendinden önceki p adet örneğin her birinin ayrı ayrı α_k katsayıları ile çarpılmasıyla elde edilir. Bunun için, Eş. 3.9'daki hesaplama doğrultusunda, gerçek değer $s\,(n)$ ile tahmin edilen değer $\dot{s}\,(n)$ arasındaki hata $e(n)$ minimize edilmeye çalışılır [80].

$$e(n) = s\,(n) - \dot{s}\,(n) = s\,(n) - \sum_{k=1}^{p} \alpha_k \cdot s\,(n - k) \tag{3.9}$$

Bu amaçla, otomatik korelasyon analizi (*autocorrelation analysis*) ve *Levinson-Durbin* algoritması yöntemlerden faydalanılır [80] .

3.3. OKT Veri Kümeleri

Bu bölümde, deneysel çalışmalarda kullanılan beş Türkçe konuşma veri kümesi; boyut, ses dosyası türü, üst veri alanları vb. özellikleri sıralanarak tanıtılmıştır.

3.3.1. ODTÜ Mikrofon Konuşması veri kümesi

ODTÜ Mikrofon Konuşması (ODTÜ MK, *METU Microphone Speech 1.0*) veri kümesi [81], çeşitli yaş gruplarından (19-50 yaş) ve cinsiyetlerden (60 erkek, 60 kadın) 120 konuşmacıya ait yaklaşık 5,6 saatlik ses ve bunlara karşılık gelen metin dosyalarından oluşur. Her konuşmacı, toplam 2462 cümle içinden rastgele seçilen 40 adet cümleyi seslendirmiştir. Ses verisi, örneklem frekansı 16 kHz olan *WAV*; metin, düz yazı şeklinde *TXT* formatındaki dosyalarda tutulur.

3.3.2. Türkçe Haber Konuşma ve Metin veri kümesi

Türkçe Haber Konuşma ve Metin (THKM) veri kümesi [82], Türkçe yayın yapan çeşitli haber kanallarındaki bültenlerden derlenen konuşmaları içeren bir veri kümesidir. Yaklaşık 13000000 kelimeden oluşan 194 saatlik ses ve metin verisini içerir. Ses verisi, 16 kHz örneklem frekansına sahip *WAV;* metin, düz yazı şeklinde *TXT* formatındaki dosyalardan oluşmaktadır.

3.3.3. Mozilla Common Voice veri kümesi

Mozilla Common Voice (Mozilla CV), Mozilla şirketi tarafından oluşturulan ve açık kaynak sunulan bir ses veri kümesidir [83]. Bu veri kümesi, dünya çapındaki gönüllülerin bağışladığı, 500'den fazla dile ait konuşma kayıtlarından oluşur. Kayıtların yer aldığı veri kümesinin sürekli güncellenen sürümleri yayımlanmaktadır. Bu metin yazıldığı sırada yayımlanmış en güncel sürümü, 14.09.2023 tarihli *Common Voice Corpus 15.0;* 1511 kişiye ait 111 saati doğrulanmış 115 saatlik ses dosyalarını içermektedir. Ses dosyaları, 48 kHz örneklem frekansında ve *MP3* formatındadır. Konuşma metni, aksan, cinsiyet, yaş, bölge vb. özniteliklerle birlikte *TSV* formatındaki dosyalarda tutulmaktadır.

3.3.4. FLEURS veri kümesi

FLEURS [84], 102 farklı dilde, her dil için ayrı ayrı yaklaşık 12 saatlik konuşma içeren bir veri kümesidir. Ses verisi 16 kHz örneklem frekansına sahip *WAV* uzantılı dosyalarda tutulmaktadır. Konuşma metni, ayrık metin, konuşmacı cinsiyeti vb. özniteliklerle beraber *TSV* formatındaki dosyalarda tutulur. Kayıtlar *dev, train, test* olmak üzere üç bölüme ayrılmıştır. Türkçe veri kümesi içeriğindeki *train, test, dev* bölümleri sırasıyla 743, 2526, 338 olmak üzere toplam 3607 kayıttan oluşmaktadır.

3.3.5. Türkçe otomatik konuşma tanıma test veri kümesi (TOKTT)

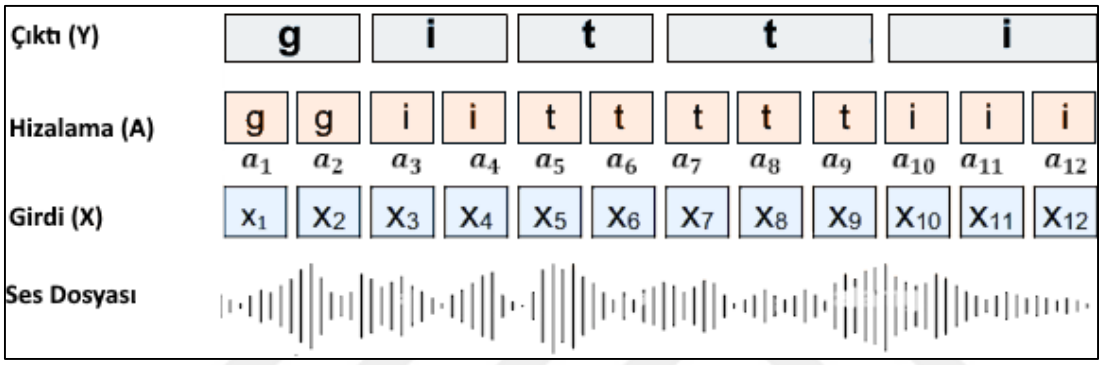
Oyucu tarafından derlenen TOKTT veri kümesi [85], 20 farklı kategoride 143 kadın, 143 erkek toplam 286 konuşmacıya ait yaklaşık 186 dakikalık konuşma verisi içerir. Ses verisi 16 kHz örneklem frekansına sahip *WAV* uzantılı dosyalarda tutulur. Veri kümesinde, her bir ses dosyasına karşılık gelen ve adında kategori, konuşmacı cinsiyeti ile sıra verilerinin yer aldığı *TXT* uzantılı konuşma metni dosyaları bulunmaktadır.

3.4. Makine Öğrenmesi Yöntemleri

Bu bölümde, çalışma kapsamında bahsedilen veya uygulamalarda kullanılan makine öğrenmesi yöntemleri açıklanmıştır.

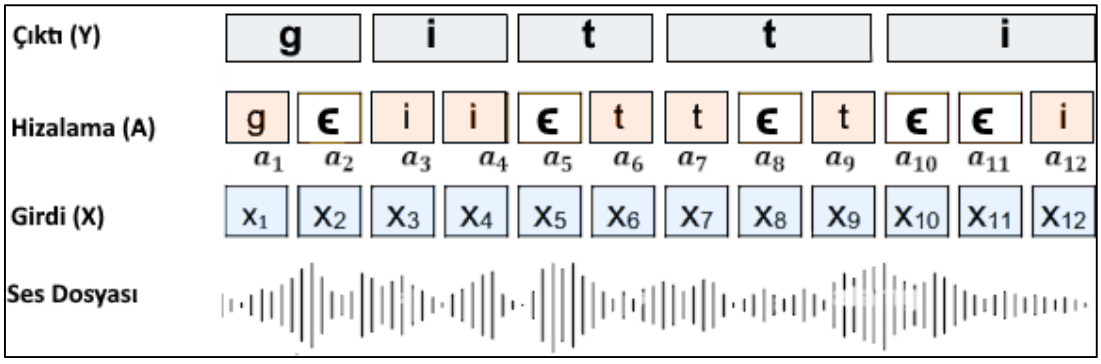
3.4.1. Bağlantıcı zamansal sınıflandırma (BZS)

BZS, herhangi bir hizalanmanın olmadığı, dizi halindeki verileri doğru bir şekilde işleyebilmek için kullanılan bir puanlama fonksiyonudur [86]. Şekil 3.4'te gösterilen konuşma verisi için $X = [x_1, x_2, x_{3,\dots} x_n]$ çerçevelerinden oluşan girdi kümesini, $A = [a_1, a_2, a_{3,\dots} a_n]$ girdiye ait hizalanmayı, $Y = [y_1, y_{2,\dots} y_n]$ sonuç metnini temsil etmektedir. X ile A hizalama kümesi arasında bire-bir, A hizalama kümesi ile Y kümesi arasında çoka-bir ilişki vardır.



Şekil 3.4. BZS'nin işleyişi, hizalama [49]

Örtüşen çerçevelerden oluşan ses girdisinde, karşılığı olmayan (boş) girdiler olabilir. Benzer şekilde, aynı ses birimi art arda çerçevelerde yer alabilir. Bu durumda, ilgili çerçevenin önceki ses biriminin devamı mı yoksa ikinci bir ses birimi mi olduğunu ayırt etmek gerekir. Bunu sağlamak için, Şekil 3.5'te gösterildiği gibi hizalama alfabesine boşluk (ε) karakteri de eklenir [49].



Şekil 3.5. BZS'nin işleyişi, hizalanma dizisine boşluk karakterinin eklenmesi [49]

Girdi verisi ve hedef etiketlerinin hizalanmasının belirli olmadığı durumlarda BZS, sonuç olasılıklarından en uygun olan çıktıyı elde etmeyi sağlar [87]. İlk olarak hizalanma olasılıkları bulunur. C, ε'yi de içeren karakter kümesini ifade eder ve $a_t \in C$'dir. Hizalanma için bir karakter dizisinin olasılığı Eş. 3.10'da verildiği gibi bu diziyi oluşturan karakterlerin olasılıkları çarpımıyla bulunur [10].

$$P_{BZS}(A \mid X) = \prod_{t=1}^{T} p(a_t \mid X) \tag{3.10}$$

Eş. 3.11'de gösterilen hesaba göre, toplam olasılıklar hesaplanır. Eşitlikte B, hizalanmadan sonuca dönüşüm fonksiyonunu ifade eder. Bu durumda $Y = B(A)$ ise fonksiyonun tersi $A = B^{-1}(Y)$'dir. Son olarak, Eş. 3.12 doğrultusunda, toplam olasılığı en yüksek çıktı seçilir [10].

$$P_{CTC}(Y \mid X) = \sum_{A \in B^{-1}(Y)} P(A \mid X) \sum_{A \in B^{-1}(Y)} \prod_{t=1}^{T} p(a_t \mid X) \tag{3.11}$$

$$\hat{Y} = \underset{Y}{\operatorname{argmax}}\, P_{CTC}(Y \mid X) \tag{3.12}$$

Örneğin, $X = [x_1, x_2, x_3, x_4]$ girdi kümesi için en büyük olasılığa sahip ilk dört hizalanma Çizelge 3.1.'deki gibi olduğunu ve diğer olasılıkların çok küçük olduğunu varsayalım. Görüldüğü gibi, 0,25 değeri ile "aa" en yüksek olasılıklı hizalanmaya sahip çıktıdır. Öte yandan, tek tek olasılıklar daha düşük olsa da, diğer üç hizalanmada ortak olan "ab" çıktısının toplam olasılığı 0,33 (0,13 + 0,11 + 0,09) olarak hesaplanır. Bu nedenle, "ab" en yüksek toplam olasılığa sahip çıktıdır.

Çizelge 3.1. Hizalama karakter olasılıkları

SIRA	HİZALAMA(A)	ÇIKTI(Y)	OLASILIK(P_{CTC})
1	**aεaa**	aa	0,25
2	**aεbb**	ab	0,13
3	**aεbε**	ab	0,11
4	**aaεb**	ab	0,09

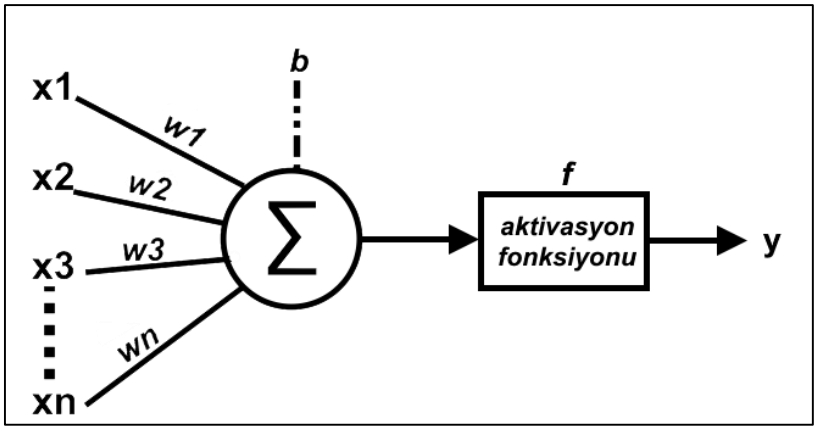
Anlaşılacağı üzere, alfabedeki tüm karakterler için olasılıkların hesaplanması, işlem yükü açısından oldukça maliyetlidir. Bu maliyeti azaltmak adına *Viterbi* Işın Araması (*Viterbi Beam Search*) [88] gibi dinamik programlama yöntemlerine başvurulur [10].

OKT sistemlerinde bu algoritmadan faydalanmak için BZS maliyet fonksiyonu kullanılır. Eş. 3.13 doğrultusunda, D veri kümesi için her bir girdi ve çıktının negatif logaritmik olasılıkları toplanır [10].

$$L_{\text{BZS}}(Y \mid X) = \sum_{(X,Y)\in \text{D}} - LogP_{BZS}(Y \mid X) \tag{3.13}$$

3.4.2. Yapay sinir ağları (YSA)

Biyolojik nöronların taklit edilmesiyle ortaya çıkan makine öğrenmesi yöntemidir [89]. Hata toleransının yüksek olması, doğrusal olmayan problemlerle daha iyi başa çıkması, uyarlanabilir olması gibi özelliklerinden dolayı yaygın olarak kullanılır. Sinir ağı katmanları, yan yana gelen algılayıcılardan (*perceptron*) oluşur. Sinir ağlarına ait algılayıcılar Şekil 3.6'daki gibi çalışır [90].

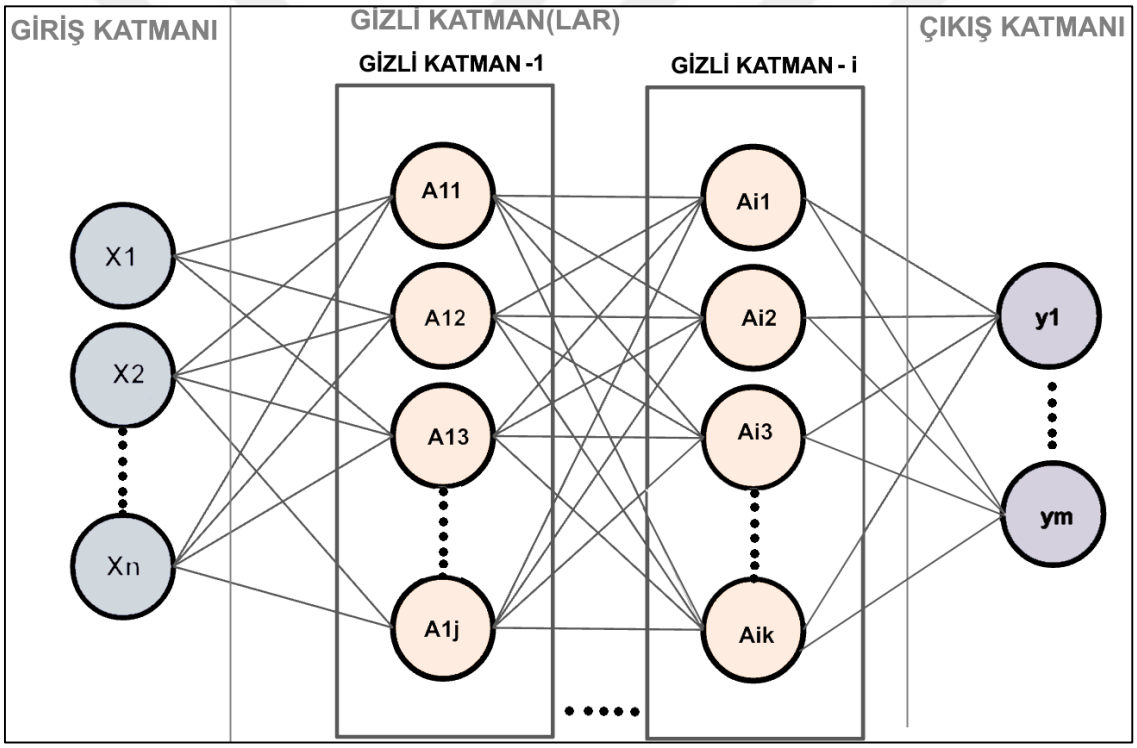


Şekil 3.6. McCulloch–Pitt yapay nöron modeli [90]

Eş. 3.14'te verildiği gibi, algılayıcı girdileriyle $(x1, x2, x3, \ldots xn)$ eğitim aşamasında hesaplanan ağırlık değerleri $(w1, w2, w3, \ldots wn)$ çarpılıp b taban değeriyle toplandıktan sonra aktivasyon fonksiyonu f kullanılarak hedef değeri elde edilir [91].

$$f(b + \sum_{i=1}^{n} x_i w_i) \tag{3.14}$$

Problemin niteliğine göre *sigmoid, Tanh, Relu, LeakRelu, Softmax, Swing* vb. aktivasyon fonksiyonları kullanılır. Birbirinden bağımsız birden fazla algılayıcının yan yana gelmesiyle katmanlar oluşur. Şekil 3.7'de gösterildiği gibi, bir girdi katmanı (*input layer*), bir ya da birden fazla gizli katman (*hidden layer*) ve çıktı katmanının (*output layer*) bir araya gelmesiyle de Çok Katmanlı Yapay Sinir Ağı meydana gelir. Başka bir deyişle, katmandaki algılayıcıların çıktılarının, diğer katmanlardaki algılayıcıların girdisi olacak şekilde bir araya geldiği yapıya Çok Katmanlı Yapay Sinir Ağı denir. Sinir ağlarında, gerçek ve tahmin edilen hedef özniteliği değeri karşılaştırılarak geri yayılım (*back propagation*) algoritması ile ağırlık değerleri güncellenir. İleri yayılım (*forward propagation*) ile elde edilen ağırlık değerleri ile yeniden hesaplama yapılır ve bu döngüler halinde tekrarlanır [91].

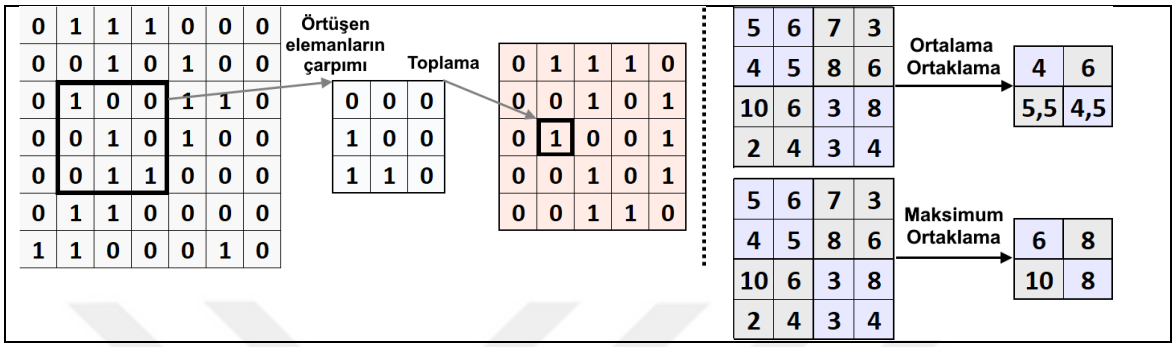


Şekil 3.7. Çok katmanlı yapay sinir ağı mimarisi

3.4.3. Evrişimli sinir ağları (ESA)

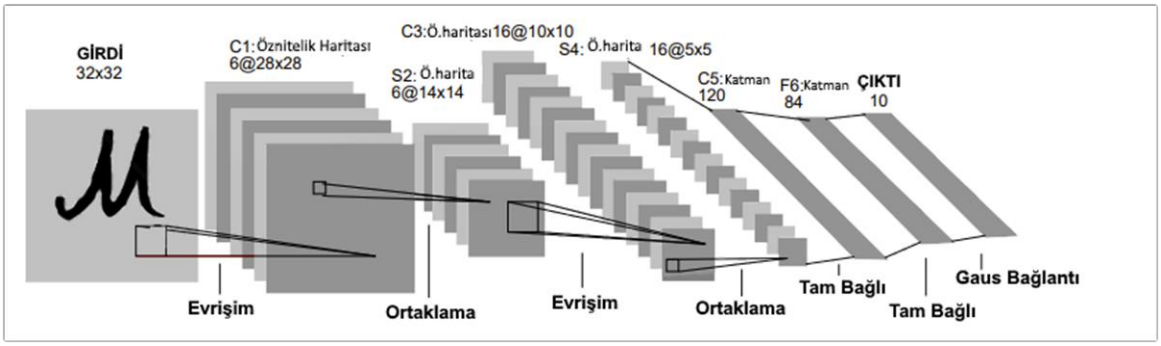
Derin Öğrenme kapsamında yer alan ESA, genellikle görüntü işleme vb. alanlarda kullanılan bir Yapay Sinir Ağı mimarisidir. ESA, art arda dizilen evrişim (*convolution*), ortaklama (*pooling*) ve aktivasyon fonksiyonu (*activation function*) katmanlarından meydana gelen çok katmanlı bir yapıdır. Evrişim katmanında belirli filtreler uygulanarak girdi özellikleri

çıkarılır. Ortaklama katmanında, hesaplama yükünü azaltmak adına, maksimum ortaklama (*maximum pooling*) ya da ortalama ortaklama (*average pooling*) yapılarak katman girdisinin boyutu daraltılır. Şekil 3.8'de, 3x3 boyutunda filtre kullanılarak yapılan evrişim ve 2x2 boyutunda maksimum, ortalama ortaklama işlemleri örneklenmiştir.



Şekil 3.8. Evrişim ve ortaklama işleminin işleyişi

Bir katmandaki tüm algılayıcıların, sonraki katmandaki bütün algılayıcılarla bağlı olduğu yapıya tam bağlı (*fully connected*) yapı denir. En temel ESA mimarilerinden *LeNet*'e ait örnek mimari Şekil 3.9'da gösterildiği gibidir.

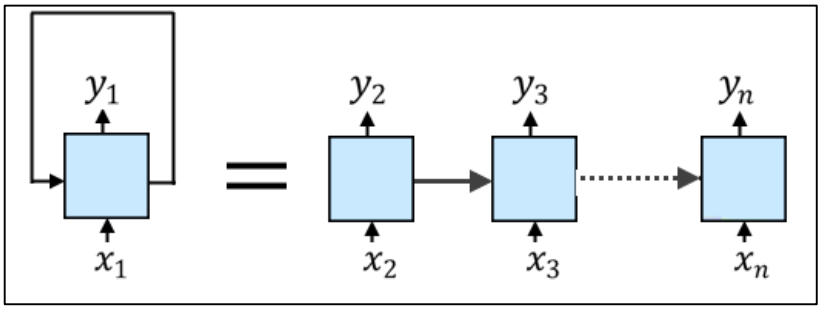


Şekil 3.9. Örnek ESA mimarisi (LeNet) [92]

3.4.4. Tekrarlayan sinir ağları (TSA)

TSA, sıralı dizi ya da zaman serisi şeklindeki verileri işlemek için kullanılan bir YSA türüdür. Şekil 3.10'da gösterildiği gibi TSA hücresi açıldığında, bir hücre çıktısının sonraki hücreye beslendiği görülür. Girdi sayısıyla ilgili herhangi bir sınırlama yoktur ve boyutu girdiden bağımsızdır. Önceki $(t-1)$ hücrenin durumu, sonraki hücrenin (t) hesabında kullanılır. Bununla birlikte, hesaplamaların yavaş olması, çok gerideki hücrelerinin

durumlarının unutulması ve sonraki hücre durumlarının önceki hücrelerde hesaba katılmaması gibi dezavantajları vardır.

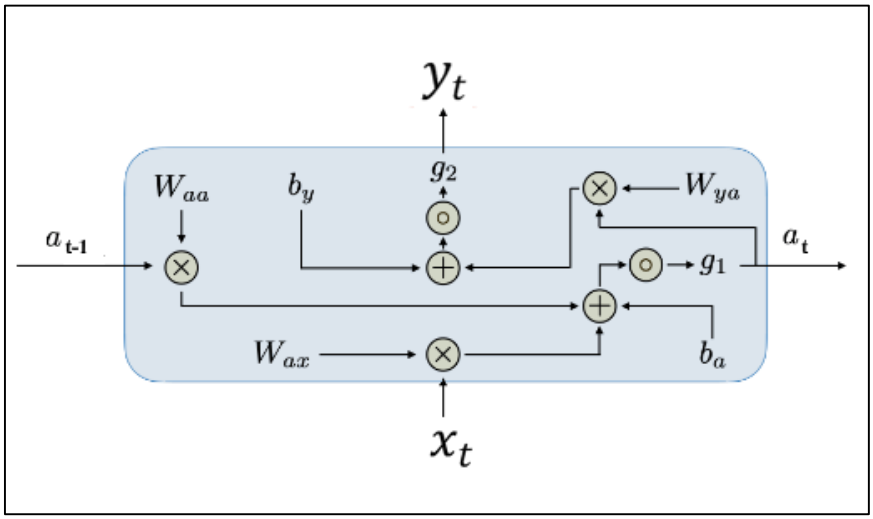


Şekil 3.10. TSA düğümü [93]

Şekil 3.11'de TSA hücresinin iç işleyişi gösterilmiştir. Sonraki hücreye aktarılan durum bilgisi a_t , çıktı y_t ve g_1, g_{12} aktivasyon fonksiyonları olmak üzere hesaplamalar Eş. 3.14 ve Eş. 3.15'teki gibi yapılır [94].

$$a_t = g_1(W_{aa}a_{t-1} + W_{ax}x_t + b_a) \quad (3.14)$$

$$y_t = g_2(W_{ya}a_t + b_y) \quad (3.15)$$

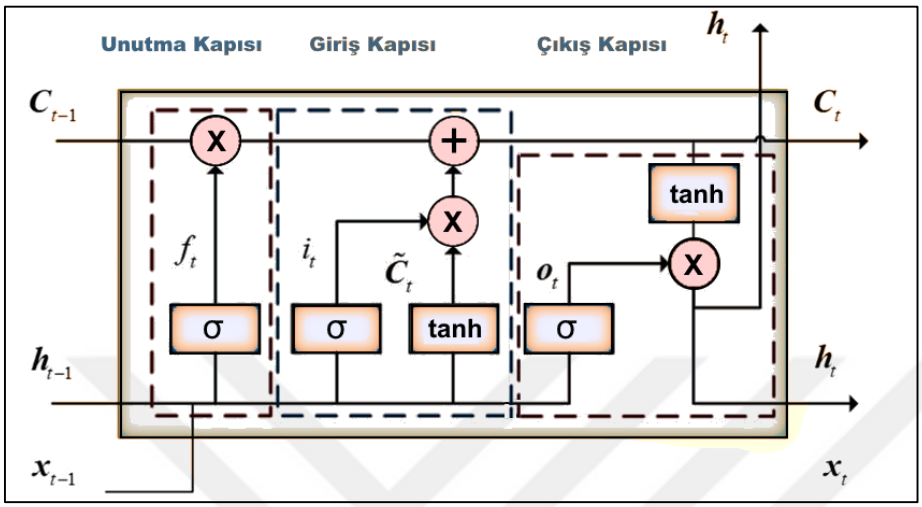


Şekil 3.11. TSA hücresinin iç işleyişi [94]

3.4.5. Uzun kısa zamanlı bellek (UKZB)

UKZB, bir TSA türevidir ve TSA ile benzer kullanım alanlarına sahiptir. UKZB, uzun dönemli bağımlılıkların işlenmesi, verideki önemli kısımların tutulmasını ya da

unutulmasını sağlayan hafıza ünitesi, unutma kapısı gibi özellikleriyle TSA'dan farklılaşır. Bu özellikler sayesinde UKZB, TSA'da var olan kaybolan eğim (*vanishing gradient*) sorunuyla daha kolay başa çıkar. Şekil 3.12'de örnek UKZB hücresi yer almaktadır [95].



Şekil 3.12. UKZB hücresinin iç işleyişi [95]

$\boldsymbol{h_{t-1}}$ ifadesi bir önceki hücrenin gizli durumunu, σ simgesi sigmoid fonksiyonunu temsil eder. Eş. 3.16'da verilen f_t işlevi, 0 ile 1 arasında bir değer alır. 1 değerini alması durumunda bir önceki hücrenin durumu ($\boldsymbol{C_{t-1}}$) tutulur, aksi halde unutulur [95].

$$f_t = \sigma\left(\boldsymbol{W}_f \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_f\right) \tag{3.16}$$

Bunların dışında hücrenin gizli durumu ($\boldsymbol{h_t}$), hücre durumu ($\boldsymbol{C_t}$) Eş. 3.17, Eş. 3.18, Eş 3.19 ve Eş. 3.20 doğrultusunda hesaplanır [95].

$$i_t = \sigma(\boldsymbol{W}_i \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + b_t),\ \overline{\boldsymbol{C}}_t = \tanh(\boldsymbol{W}_C \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_C) \tag{3.17}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_0 \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_0) \tag{3.18}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \cdot \tanh(\boldsymbol{C}_t) \tag{3.19}$$

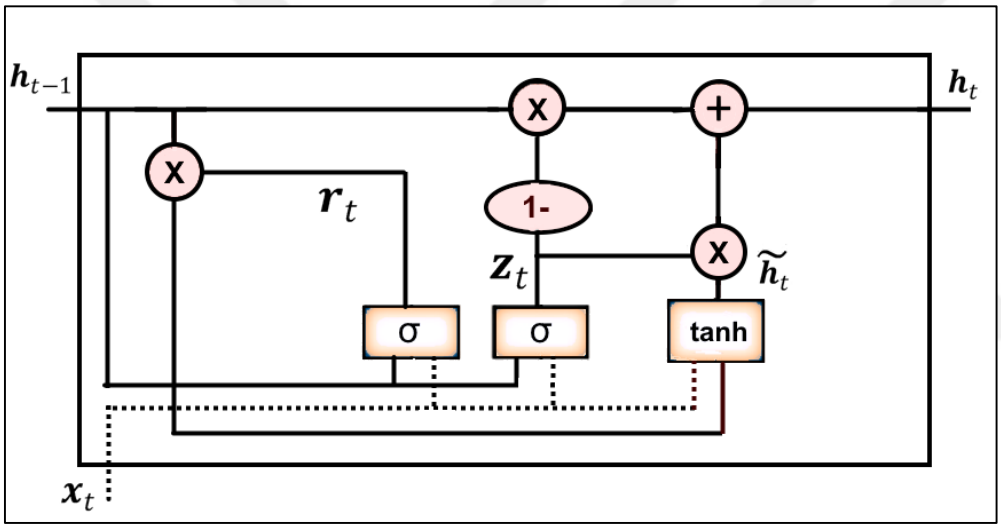
$$\boldsymbol{C}_t = C_{t-1} \cdot f_t + \overline{\boldsymbol{C}}_t \cdot i_t \tag{3.20}$$

Çift Yönlü UKZB, UKZB'nin geliştirilmiş bir türevidir. Bu yapıda, ilk hücre dışındaki hücreler hem önceki hem de sonraki hücrenin durumunu girdi olarak alır. Çift Yönlü UKZB,

zaman serileri analizi gibi alanlarda kullanıldığında genellikle UKZB'ye nazaran daha iyi performans gösterir [96].

3.4.6. Geçitli tekrarlayan birim (GTB)

GTB, UKZB'ye benzer bir şekilde uzun vadeli bağımlılıkları öğrenebilen bir TSA türüdür. UKZB'ye göre daha basit bir yapıya sahiptir [97]. Şekil 3.13'te gösterildiği gibi GTB'de, UKZB'deki giriş, çıkış ve unutma kapıları birleştirilmiştir. UKZB'den farklı olarak sıfırlama (*reset*) kapısına sahiptir. Daha az parametre içerir ve daha hızlı eğitim süreçlerine olanak tanır.



Şekil 3.13. GTB hücresinin iç işleyişi [97]

GTB'de hesaplamalar Eş. 3.21, Eş. 3.22, Eş. 3.23 ve Eş 3.24'te verildiği gibi yapılır. Verilen eşitliklerde, σ simgesi sigmoid fonksiyonunu gösterir. $\boldsymbol{h_t}$ ifadesi hücrenin gizli durumunu ve $\boldsymbol{r_t}$ ifadesi sıfırlama kapısının değerini temsil eder.

$$z_t = \sigma(W_z h_{t-1} + U_z x_t) \tag{3.21}$$

$$r_t = \sigma(W_r h_{t-1} + U_r x_t) \tag{3.22}$$

$$\tilde{h}_l = \tanh(W(r_l^* h_{l-1}) + U x_l) \tag{3.23}$$

$$h_t = (1 - z_t)^* h_{t-1} + z_t{}^* \tilde{h}_t \tag{3.24}$$

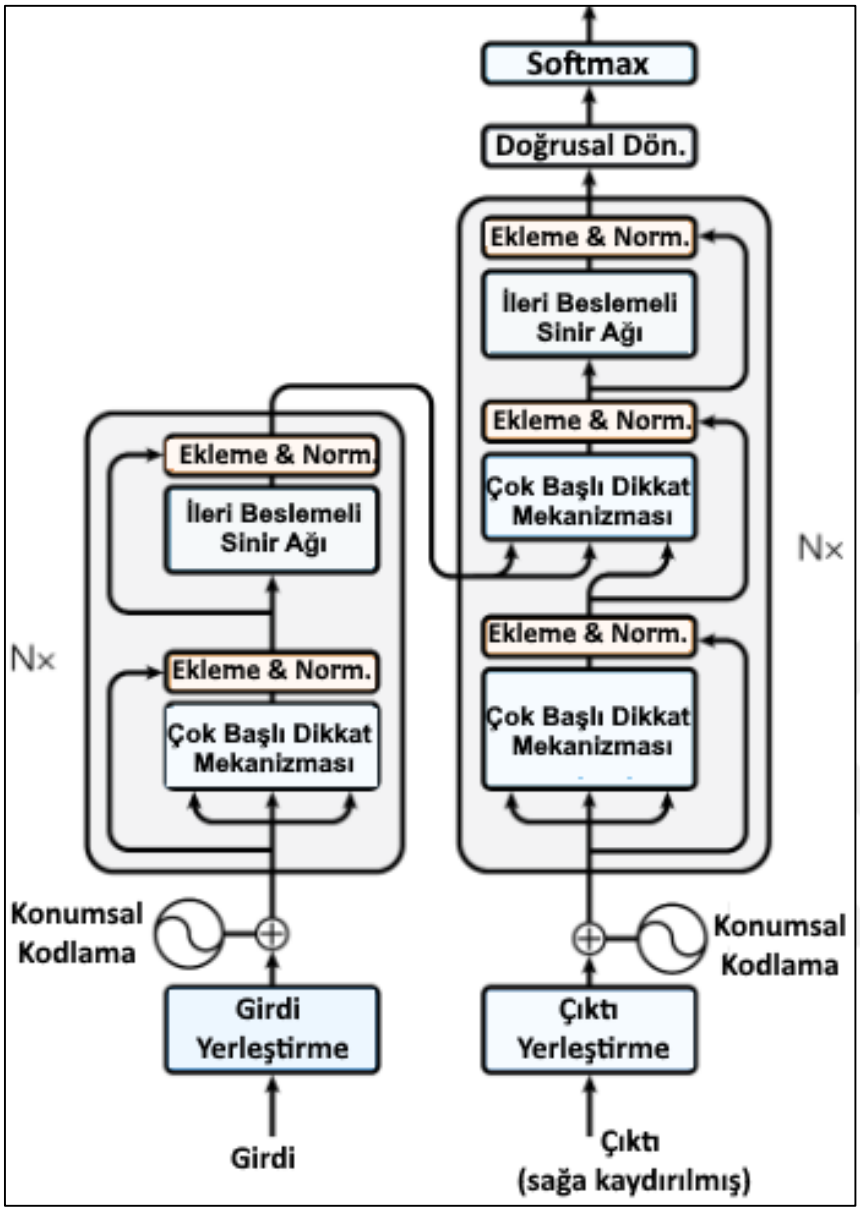
3.4.7. Dönüştürücüler

TSA ve türevleri GTB, UKZB gibi çözümler makine çevirisi, dil modelleme, dizi aktarım ve modelleme görevlerinde bir noktaya kadar başarılı olmuştur. Öte yandan, sıralı yapısı nedeniyle paralel işleme izin vermemesi, uzun dizilerdeki unutma sorunu ve karmaşıklığı nedeniyle yeni çözümlere ihtiyaç duyulmuştur. Bu sorunu aşmak adına geliştirilen çözümlerden biri, *seq2seq* [98] mimarisidir. *Seq2seq* mimarisi, TSA birimlerinden oluşan kodlayıcı (*encoder*) ve çözümleyiciden (*decoder*) meydana gelen bir yapıya sahiptir ancak öncüleri gibi uzun dizilerle başa çıkmakta yetersiz kalır.

Geliştirilen çözümlerden bir başkası, veri içindeki belirli noktaların öne çıkarılmasını, önemli noktalara odaklanılmasını sağlayan Dikkat (*attention*) mekanizmasının modele eklenmesi olmuştur [99]. Bununla birlikte, önerilen mimarilerdeki; uzun mesafe bağımlılıkların korunamaması, genelleme yeteneğinin zayıf kalması, hesaplama maliyetinin yüksekliği, paralel işlemle ilgili kısıtlar gibi sorunlar nedeniyle başka çözüm arayışlarına gidilmiştir.

Bu noktada, bahsedilen sorunların aşılması adına DÖ kapsamında yer alan dönüştürücü (*transformser*) mimarisi önerilmiştir. 2017 yılında *Vaswani* ve diğerleri tarafından yazılan makalede [42] genel çerçevesi ortaya konan dönüştürücü, kodlayıcı ve çözümleyici olmak üzere ikili bir yapıdan oluşur.

Kodlayıcı, dikkat mekanizması ve İBSA olmak üzere iki ana bileşene sahiptir. Çözümleyici, kodlayıcının bağlı olduğu bir öz dikkat mekanizması ve İleri Beslemeli Sinir Ağı (İBSA) ana bileşenlerinden meydana gelir. Dönüştürücü mimarisinin bileşenlerine ait temel yapı Şekil 3.14'te gösterildiği gibidir.



Şekil 3.14. Temel dönüştürücü mimarisi [42]

Konumsal kodlama

Dizi boyu uzadıkça, girdiye ait indis çok büyük değerler alır. İndisler, bu halleriyle, dönüştürücüde kullanıma uygun değildir. Normalizasyon işlemi bir noktaya kadar çözüm sağlasa da, dizi boyutundaki farklılıklar yine sorun yaşanmasına neden olur.

Konumsal kodlama ile girdi dizisi, sinüs ve kosinüs fonksiyonları kullanılarak konum matrisine dönüştürülür. Konumsal kodlama (*positional encoding*), girdinin sırası pos,

modelin çıktı boyutu d_{model} $(0 \leq i \leq d_{\text{model}}/2)$, ve kullanıcı tanımlı ölçüt n olmak üzere Eş. 3.25 ve Eş 3.26'da gösterildiği gibi hesaplanır.

$$PE_{(pos,2i)} = \sin\left(pos/\mathrm{n}^{2i/d_{\text{model}}}\right) \tag{3.25}$$

$$PE_{(pos,2i+1)} = \cos\left(pos/\mathrm{n}^{2i/d_{\text{model}}}\right) \tag{3.26}$$

Örneğin, $d = 4$ ve n=10 000 olmak üzere bir metin girdisindeki ilk 6 kelimeye ait konumsal kodlama matrisi Çizelge 3.2'deki gibi oluşur.

Çizelge 3.2. Örnek konumsal kodlama matrisi

i =	0	0	1	1
	$\boldsymbol{\sin\left(pos/\mathrm{n}^{2i/\mathrm{d}}\right)}$	$\boldsymbol{\cos\left(pos/\mathrm{n}^{2i/\mathrm{d}}\right)}$	$\boldsymbol{\sin\left(pos/\mathrm{n}^{2i/\mathrm{d}}\right)}$	$\boldsymbol{\cos\left(pos/\mathrm{n}^{2i/\mathrm{d}}\right)}$
x_0	0	1	0	1
x_1	0.841471	0.540302	0.009999	0.999950
x_2	0.909297	-0.416147	0.019999	0.999800
x_3	0.141120	-0.989992	0.029996	0.999550
x_4	-0.756802	-0.653644	0.039989	0.999200
x_5	-0.958924	0.283662	0.049979	0.998750

Çok başlı dikkat mekanizması (ÇBDM)

Şekil 3.15'te görüleceği üzere, ÇBDM, paralel h adet Ölçeklendirilmiş Nokta Çarpımı Dikkat Mekanizmasının (*Scaled Dot-Product Attention*) bir araya gelmesiyle oluşur. Dönüştürülmüş girdinin (X) ayrı ayrı ağırlık matrisleri ile çarpılmasıyla, her dikkat birimine beslenen ve Eş. 3.27, Eş. 3.28 ve 3.19'da verilen Sorgu (***Q**uery*), Anahtar (***K**ey*) ile Değer (***V**alue*) elde edilir.

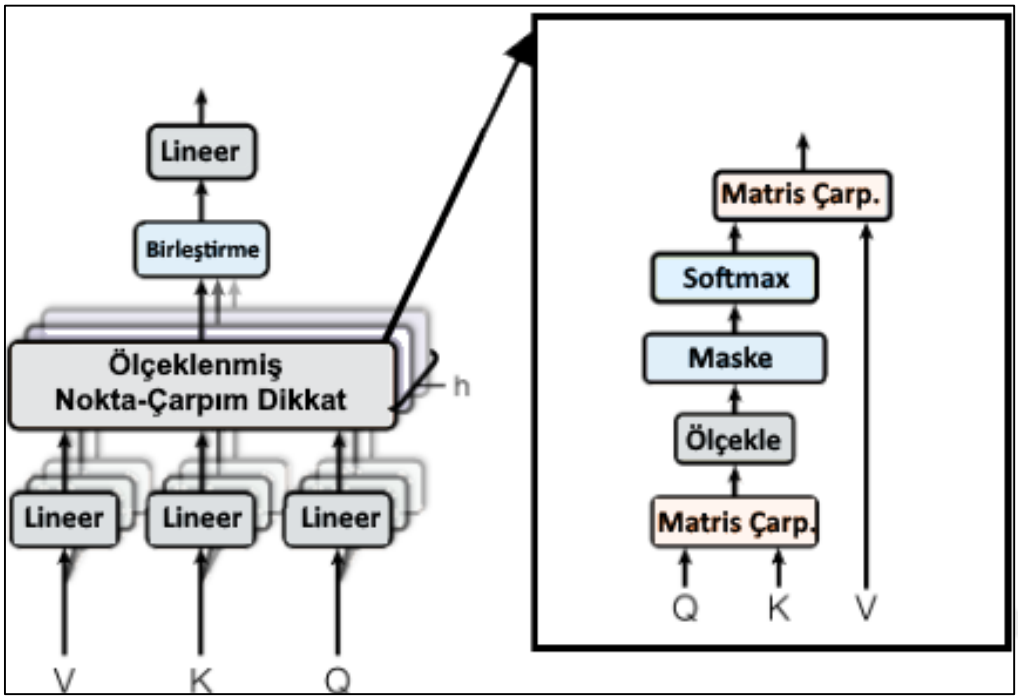
$$Q = XW_Q^T \tag{3.27}$$

$$K = XW_K^T \tag{3.28}$$

$$V = XW_V^T \tag{3.29}$$

Eş. 3.30'da gösterildiği gibi, sorgu (Q) ve anahtar (K) matris çarpımı, boyutun kareköküyle ölçeklendirilir ve *softmax* fonksiyonu ile elde edilen ağırlık, değer (V) matrisi ile çarpılır.

$$\text{Dikkat}\ (Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3.30}$$



Şekil 3.15. Çok başlı dikkat mekanizması [42]

Eş. 3.31'de yer aldığı şekilde, her bir dikkat biriminden elde edilen sonuç Eş. 3.32'de gösterilen şekilde birleştirildikten sonra bir üst katmana iletilir.

$$Dikkat_i = \text{Dikkat}\ (\ QW_i^Q,\ KW_i^K, VW_i^V) \tag{3.31}$$

$$\text{ÇokBaşlıDikkat}\ (Q, K, V) = \text{Birleştir}(Dikkat_1, Dikkat_2, \dots) \tag{3.32}$$

<u>İleri beslemeli sinir ağı (İBSA)</u>

İBSA, bir önceki başlıkta bahsedilen Çok Başlık Dikkat Mekanizmasının çıktısıyla beslenir. İBSA'ya ait hesaplama, koyu kısım *ReLU* aktivasyon fonksiyonunu göstermek üzere, Eş. 3.33'teki gibi yapılır. Bir önceki katmanın çıktısı ağırlıklandırılarak *ReLU* aktivasyonu

fonksiyonuna beslenir. Daha sonra, ikinci bir ağırlık matrisi ile çarpılıp bir sabit eklendikten sonra çözümleyici girdisi olarak kullanılır.

$$\text{İ}BSA(x) = \mathbf{max}(\mathbf{0}, \mathbf{x}\boldsymbol{W_1} + \mathbf{b_1})W_2 + b_2 \quad (3.33)$$

Kodlayıcı yer alan mekanizmaların aynısı çözümleyicide de yer alır. Farklı olarak:

- Kodlayıcıdaki konumsal kodlama işleminde girdi olarak $x_1, x_2, x_3, \ldots x_n$ alınırken çözümleyicide y_0 $(başlangıç), y_1, y_2, y_3, \ldots y_{n-1}$ girdi olarak kullanılır.
- Çözümleyicide yer alan ilk ÇBDM'de aşırı öğrenmeyi önlemek adına maskeleme yapılır.
- Çözümleyicide son çıktı *softmax* fonksiyonu kullanılarak sonuç dizisindeki her bir elemana 0-1 arasında olasılık değeri atanır ve en yüksek olasılığa sahip çıktılar seçilir.

3.4.8. Denetimli, denetimsiz ve yarı denetimli öğrenme

Makine öğrenmesi temelde, denetimli öğrenme (*supervised learning*) ve denetimsiz öğrenme (*unsupervised learning*) olmak üzere iki temel türe ayrılır. Denetimli öğrenme için kullanılan veri kümelerinde, her bir örnek için girdi öznitelikleri ($x_1, x_2, x_3, \ldots x_n$) ve bir ya da birden fazla hedef özniteliği (y) bulunur. Bu tür veri kümeleri, "etiketli veri kümesi" olarak adlandırılır. Veri kümeleriyle eğitilen model kullanılarak yeni girdi örnekleri için hedef özniteliğinin değeri tahmin edilmeye çalışılır. Denetimli öğrenme, genellikle regresyon ve sınıflandırma görevlerinde kullanılır [91].

Denetimsiz öğrenme yönteminde kullanılan veri kümeleriyse, sadece girdi özniteliklerinden ($x_1, x_2, x_3, \ldots x_n$) oluşur ve herhangi bir hedef özniteliği içermez. Bu nedenle, denetimsiz öğrenmede kullanılan veri kümeleri "etiketsiz veri kümesi" olarak adlandırılır. Denetimsiz öğrenmede, veri örneklerinin benzerliklerine göre gruplandırılması amaçlanır.

Bu iki türün dışında, yarı denetimli öğrenme (YDÖ, *semi-supervised learning*) olarak adlandırılan başka bir öğrenme türü vardır. YDÖ, kavramsal olarak denetimli ve denetimsiz öğrenme arasında konumlanır. YDÖ üzerine yapılan çalışmalarda genellikle sınıflandırmaya odaklanılmıştır. Performansı arttırmak için, sınırlı miktarda etiketli veri ile birlikte etiketlenmemiş veriden de yararlanılmaya çalışılır [100]. YDÖ, etiketlenmiş verinin sınırlı olduğu durumlarda performanslı bir sınıflandırma yapabilmek için kullanılan etkili bir

stratejidir [101]. YDÖ'de, model etiketli ve etiketsiz veri birlikte kullanılarak eğitilir ve yeni veri örnekleri için hedef özniteliğinin değeri tahmin edilir [102].

Sınıflandırma bağlamında düşünürsek, YDÖ'deki temel varsayım, aynı yüksek yoğunluk bölgesinde bulunan iki veri örneğinin büyük olasılıkla aynı sınıf etiketine sahip olması gerektiğidir. Bu kabul, düzgünlük varsayımı *(smoothness assumption)* olarak adlandırılır [103]. Düzgünlük varsayımına göre veri kümesindeki iki veri örneğinin girdi öznitelikleri (x, x') birbirine yakınsa, bu örneklere ait hedef özniteliklerinin sınıf etiketleri de (y, y') aynı olmalıdır. Düzgünlük varsayımı, aynı zamanda, geçişli olarak da uygulanabilir. X_1, sınıf etiketine sahip bir veri örneği olsun. Ayrıca sınıf etiketine sahip olmayan X_2 ve X_3 veri örnekleri mevcut olsun. Veri uzayında X_2 veri örneği, X_1 veri örneğine ve X_3 veri örneği, X_2 veri örneğine yakınsa; geçişli yayılımdan dolayı, X_1'e yakın olmasa bile X_3, X_1 veri örneği ile aynı sınıf etiketine sahip olacaktır [101].

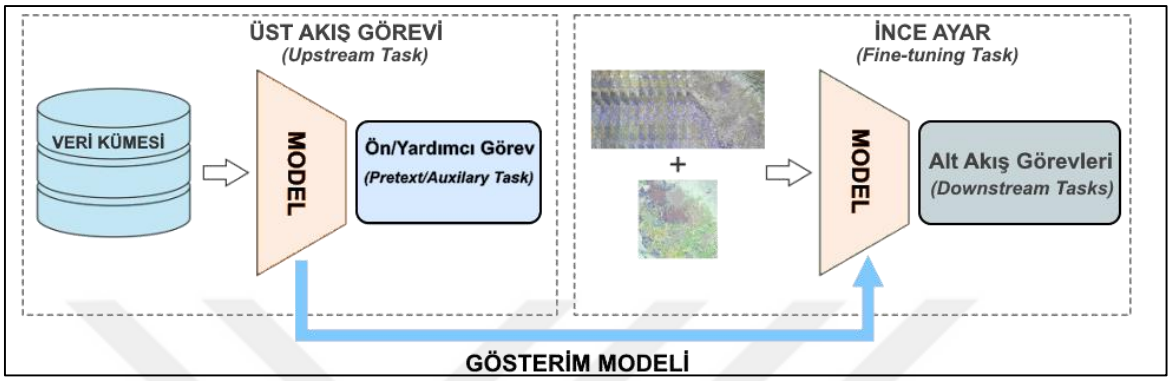
3.4.9. Kendi kendine denetimli öğrenme (KKDÖ)

Kendi Kendini Denetleyen Öğrenme ve Öz Denetimli Öğrenme olarak da adlandırılan KKDÖ oldukça yeni bir yöntemdir. OKT, Doğal Dil İşleme ve bilgisayarlı görü görevlerinde, zaman serisi verilerin işlenmesinde kullanılır. Çalışma prensibi, büyük miktarda etiketsiz veriyle gösterim modeli oluşturulması ve sonrasında kullanım alanına göre küçük miktardaki ikincil eğitim verisiyle modelin ince ayar (*fine-tunning)* yapılmasıdır. Bu özelliği, sıklıkla denetimsiz öğrenme ile karıştırılmasına neden olur. Denetimli ve denetimsiz öğrenme adımlarına sahip olduğu için, yarı denetimli öğrenme gibi bu iki öğrenme türünün arasında konumlandırılır [104]. Konuşma tanıma alanında *Wav2vec* ve *Wav2vec 2.0* ve Doğal Dil İşleme alanında, birçok popüler örneğin öncüsü sayılabilecek, *BERT* (Transformatörlerden Çift Yönlü Kodlayıcı Temsilleri, *Bidirectional Encoder Representations from Transformers*) [105] gibi başarılı örnekleri mevcuttur.

KKDÖ, iki aşamalı bir işleyişe sahiptir. Birinci aşamada, etiketsiz veri kullanılarak yapılan denetimsiz öğrenmeyle temsillerin oluşturulduğu üst akış görevi (*upstream*/*pretext/auxiliary task*) gerçekleştirilir [104]. Daha sonra, asıl görev olan sınıflandırma, kümeleme vb. alt akış görevi (*downstream task*) gerçekleştirilir. Bu nedenle, KKDÖ bir transfer öğrenme yönetimi olarak da kabul edilir. Bazı durumlarda, ara aşama olarak, amaç doğrultusunda belirli seviyede ön katmanlar dondurularak, küçük miktarda etiketli veriyle ince ayar (*fine-tuning)* yapılabilir. İnce ayar, üst akış görevinde elde edilen

modelin performansını iyileştirmek için kullanılan bir tekniktir. Bu aşamada, modelin ağırlıkları, alt akış görevinin daha iyi performans göstermesi için güncellenir.

Görüntü sınıflandırma bağlamında KKDÖ, Şekil 3.16'da gösterilen işleyişe sahiptir.



Şekil 3.16. Kendi kendine denetimli öğrenmenin işleyişi [106]

Şekilden de anlaşılacağı üzere, ilk aşamada ön bilgi çıkarma görevi (*pretext task*) icra edilir. Yüksek miktarda veri örneği ile denetimsiz öğrenme yapılarak temsil/bağlam bilgisi elde edilir. Ara aşamada etiketli veri ile üst akış görevinde oluşan model üzerinde ince ayar görevi (*fine-tuning task*) yürütülerek iyileştirme yapılır. Birinci aşamadaki temsil oluşturma işleminde kayıp fonksiyonu olarak, KÖK gibi pozitif ve negatif örneklerin birlikte kullanıldığı ya da KÖK-olmayan, sadece pozitif örneklerin kullanıldığı yöntemlerden faydalanılır. Alt akış görevinde denetimli öğrenme yapılabileceği gibi denetimsiz öğrenme de gerçekleştirilebilir [107].

Yukarıda da ifade edildiği gibi KKDÖ, görüntü işleme ve sıralı dizi halinde ifade edilen ses gibi verilerin işlenmesine oldukça yeni bir yaklaşımdır. Yöntemin amacı, veriye ait bağlamın da öğrenme sürecine dâhil edilmesidir. Görüntü işleme üzerinden basit bir şekilde örneklersek, geleneksel MÖ yöntemlerinde yeşil renkli bir piksel, bulunduğu bağlamdan bağımsız bir veridir. Oysaki KKDÖ'de, bu piksel bulunduğu örneğe göre farklı temsillere sahiptir. Örneğin, ağaç yaprağındaki bir yeşil nokta ile çimene ait bir yeşil noktanın KKDÖ bağlamında temsilleri farklı olacaktır.

Motor becerilerin modellenmesinde, insan öğrenmesini taklit eden takviyeli öğrenme (*reinforcement learning*) yöntemlerinden faydalanılır. Öte yandan, konuşma ve yazma gibi becerilerin modellenmesinde KKDÖ gibi farklı yöntemlere ihtiyaç duyulur. İşitilen

kelimelerin insanlar tarafından metne dönüştürülmesi üzerinden örnek verirsek, bu işlem kabaca üç adımda gerçekleşir:

1. Yazma öğrenilinceye kadar, nasıl yazıldığını bilinmeyen birçok kelimeyi temsil eden seslerin duyarak öğrenilmesi. Bu, KKDÖ'deki üst akış, denetimsiz öğrenme görevinin icra edildiği temsil oluşturma aşamasına karşılık gelir.
2. Belirli kelimelerin yazılışlarının öğrenilmesi. Bu adım, KKDÖ bağlamında ince ayar işlemine karşılık gelir.
3. KKDÖ'deki karşılığı alt akış görevi olan, nasıl yazıldığı bilinmeyen bir kelime ya da kelime dizisinin metine dönüştürülmesi.

Verilen örnekten de anlaşılacağı üzere, KKDÖ, konuşmanın yazıya dökülmesi gibi becerilerin modellenmesinde insan öğrenmesini taklit eder.

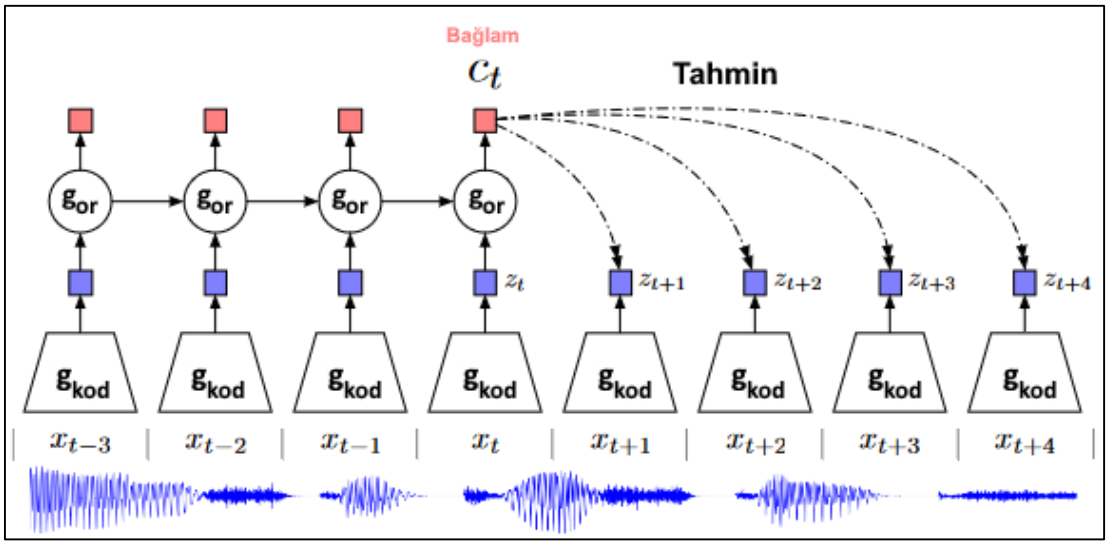
Millet ve diğerleri tarafından yapılan çalışma [108] konuyla ilgili çarpıcı örneklerden biridir. Bu çalışmada, bir konuşmanın KKDÖ tabanlı *Wav2vec 2.0* modelindeki karşılığı ile insan beynindeki oluşturduğu etki karşılaştırılmıştır. Konuşmanın beyinde oluşturduğu aktivite sinyalleri ile *Wav2vec 2.0* modelindeki temsili arasında yapılan karşılaştırılma sonucunda, aralarında yüksek bir ilgileşim olduğu ortaya konulmuştur.

Özetle, Kendi Kendine Denetimli Öğrenme, yüksek miktarda etiketsiz veri ile temsil modelleri oluşturulmasını ve oluşturulan modeller üzerinde ince ayar yapılarak farklı görevlerin yerine getirilmesini sağlayan bir yöntemdir.

3.4.10. Karşılaştırmalı öngörücü kodlama (KÖK)

KÖK, zaman serisi verilerin işlenmesinde, OKT ve DDİ gibi alanlarda kullanılan bir Temsil Öğrenme yöntemidir. Bu yöntem, veri üzerinde yapılan tahminlerin karşılaştırmayla öğrenilmesine dayanır. Sıralı dizi halindeki veri için gelecekteki örneklerin öncekilerden tahmin edilmesi ve veri kümesindeki gizli örüntülerin ortaya çıkarılması için kullanılır.

Konuyla ilgili temel çalışma, 2018 yılında *Oord* ve diğerleri tarafından yazılan Temsil Öğrenme makalesidir [44]. KÖK'ün, ses verisi işleme bağlamındaki işleyişi Şekil 3.17'de gösterildiği gibidir.



Şekil 3.17. Temsil öğrenme bağlamında KÖK'ün kullanımı [44]

g_{kod} ile temsil edilen kodlayıcı, x_{t+n} ile temsil edilen pencerelere bölünmüş ses verisini işleyerek, z_{t+n} ile temsil edilen özellikleri çıkarır. Sonrasında TSA tabanlı otoregresör (g_{or}) da kullanılarak c_{t+n} ile temsil edilen bağlam matrisleri elde edilir. Buradaki amaç, t anından sonraki z_{t+n}'leri en iyi tahmin eden bağlam c_t'nin bulunmasıdır.

KÖK'de, tahminleme işleminde x (gelecek), c (şimdi) için karşılıklılık bilgisi en üst düzeyde korunacak şekilde Eş. 3.34'teki birleşik vektör temsili kullanılır.

$$I(x;c) = \sum_{x,c} p(x,c) log \frac{p(x|c)}{p(x)} \tag{3.34}$$

Üretici model ile c_t, x_{t+k} verisinden doğrudan tahmin edilemez. İlk olarak, $z_t = g_{enc}(x_t)$ kodlayıcı ile diziye ait temsillerin elde edilmesi gerekir. Bağlam bilgisi, otoregresör ile t zamanından önceki gizli temsillerden $c_t = g_{enc}(z_{\leq t})$ eşitliğiyle çıkarılır. Bunun yerine Eş. 3.35'teki gibi ardışıklık bilgisinin korunduğu bir yapı kullanılır ve bu *log-bilineer* model ile gösterilir [44].

$$f(x_{t+k}, c_t) \propto \frac{p(x_{t+k}|\, c_t)}{p(x_{t+k})}, f(x_{t+k}, c_t) = \exp(z_{t+k}^T W_k c_t) \tag{3.35}$$

Eşitlikte, W_k ağırlıklandırmayı temsil ederken, $\propto$ orantılı anlamında kullanılmıştır. Öğrenmenin gerçekleşmesi için kodlayıcı ve otoregresif modelin birlikte eğitildiği, *InfoNCE*

olarak adlandırılan Gürültü-Karşılaştırmalı Öngörücü (*Noise-Contrastive Estimation*) tabanlı kayıp fonksiyonunun kullanıldığı bir model oluşturulur [44]. Pay kısmı pozitif örnek sayısını, payda kısmı negatif örnek sayısını göstermek üzere *InfoNCE* kayıp değeri Eş. 3.36'daki gibi hesaplanır [44].

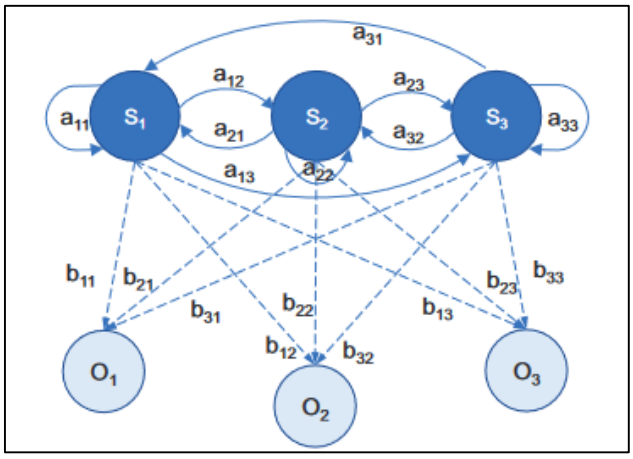
$$L_N = -E_X\left[\log\frac{f_k(x_{t+k},c_t)}{\sum_{x_j\epsilon X} f_k(x_j,c_t)}\right] \tag{3.36}$$

3.4.11. Saklı markov modeli (SMM)

Saklı Markov Modeli (*Hidden Markov Model*) istatistiksel bir modelleme tekniğidir ve genellikle sıralı ya da zaman serisi şeklindeki veriler için kullanılır. OKT, DDİ, bilgisayarlı görü, genetik dizilerin analizi vb. gibi kullanım alanlarına sahiptir. SMM'deki temel prensip, gözlemlenebilir bilgiden, gözlemlenemeyen gizli ilişkilerin çıkarılmasıdır. Şekil 3.18'deki modeli ele alırsak. $S = \{S_1, S_2, S_3\}$ gizli durumları, $O = \{O_1, O_2, O_3\}$ gözlemlenen durumları temsil eder. t zamanındaki gizli durum X_t ile, gözlemlenen durum Y_t ile temsil edilir. $a_{ij} = P[X_t = j|X_{t-1} = i]$ elemanlarından oluşan A matrisi, gizli durumlar arasındaki geçiş olasılık dağılımını (*state transition probability*) gösterir. $b_{ij} = P[Y_t = j|X_t = i]$ elemanlarından oluşan B matrisi, gizli durumlar ile gözlemlenebilir durumlar arasındaki çıkarım olasılıklarının (*emission probability*) dağılımını içerir. Gizli durumların başlangıçtaki dağılımı π ile temsil edilir.

Buna göre gizli durumlar ve gözlemlenen durumlar arasındaki bağlantı ve olasılık vektörleri Eş. 3.37'deki gibi bulunur.

$$S_t^T = S_{t-1}^T A, \quad O_t^T = S_t^T B \tag{3.37}$$



Şekil 3.18. Üçer gizli durum ve gözlemlenen durumu içeren saklı markov modeli [109]

İşleyişten de anlaşılacağı üzere, SMM mevcut haliyle kullanıldığında yüksek hesaplama maliyeti yaratır. SMM, OKT gibi işlerde kullanılırken, hesaplama maliyetini azaltmak adına *Viterbi, Baum-Welch* algoritması gibi dinamik programlama tekniklerinden faydalanılır [110].

3.4.12. Gauss karışım modeli (GKM)

Gauss Karışım Modeli (*Gauss Mixture Model*), bir veri kümesindeki gizli yapıları keşfetmek ve bunu bir dizi Gauss dağılımıyla modellemek için kullanılan bir olasılık modelidir. GKM, aslında bir denetimsiz öğrenme yöntemidir ve veri kümesini farklı alt gruplara ayırmak, karmaşık yapıları modellemek için kullanılır. GKM'de, veri noktalarının Gauss normal dağılımlarını temsil ettiğini varsayılır ve bundan faydalanılarak veri anlamlandırılmaya çalışılır. GKM'deki ana prensip, veri setindeki karmaşık dağılımları daha basit Gauss dağılımlarıyla modelleyerek veriyi daha iyi anlamaktır. Ancak bu yöntem, veri setindeki alt gruplar birbirinden farklı Gauss dağılımları ile temsil ediliyorsa kullanışlıdır. Standart Gauss dağılım fonksiyonu, ortalama (μ), standart sapma (σ) ve varyans (σ^2) kullanılarak, Eş. 3.38'deki gibi ifade edilir.

$$G(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3.38}$$

M adet küme içeren bir GKM için ortalamalar vektörü (μ), kovaryans matrisini (Σ), karışım dağılım ağırlıkları (w_i) kullanılarak bileşen yoğunlukları/olasılıkları Eş. 3.39'daki [111] gibi hesaplanır.

$$p(\mathbf{x}|\lambda) = \sum_{i=1}^{M} w_i G(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\ \lambda = \{w_i, \mu_i, \Sigma_i\}, \sum_{i=1}^{M} w_i = 1 \quad (3.39)$$

$\boldsymbol{x}$ ile temsil edilen, D boyutlu öznitelikleri de içerecek şekilde dağılım fonksiyonu açıldığında, Eş. 3.40'daki yapıya dönüşür [111].

$$G(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\ \boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right\} \quad (3.40)$$

GKM'deki ilgili parametreler eğitim verilerinden tahmin edilirken, Beklenti-maksimizasyon (*expectation-maximization*) veya En Büyük Sonsal (*maximum a posterior*) algoritmaları tekrarlı şekilde kullanılır [111].

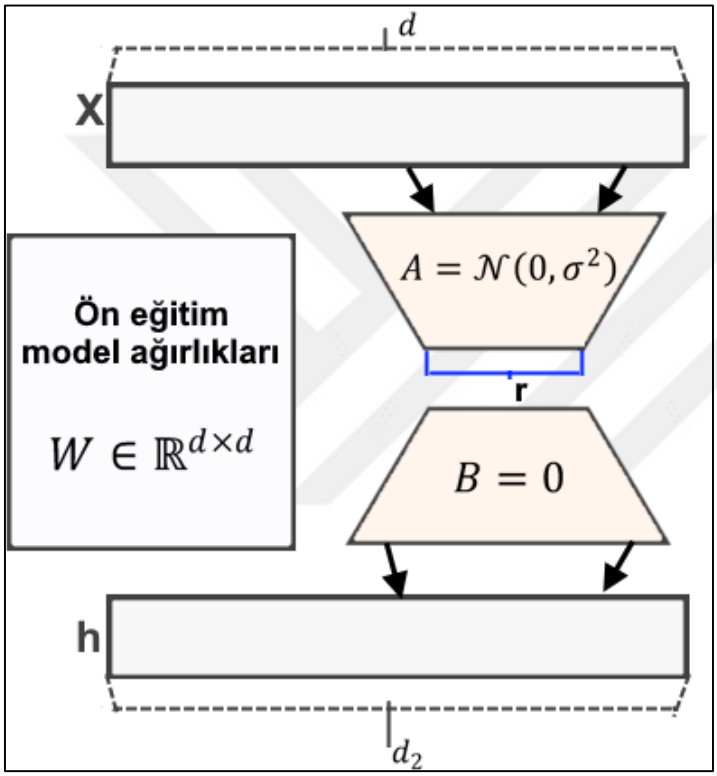
OKT, konuşmacı tanıma, görüntü işleme, yazı tanıma vb. alanlardaki veri setleri genellikle heterojen ve karmaşık olduğundan, SMM gibi esnek olasılık modelleri kullanılarak veri analizi yapılabilir.

3.4.13. Düşük dereceli uyumlama (DDU)

Büyük Dil Modeli (BDM), DDİ, OKT sistemlerinde kullanılan modeller genellikle çok fazla parametre içerirler ve büyük boyutlara sahiptirler. Örneğin, *Whisper large-v2* yaklaşık 1,5 milyar parametreye içerirken, GPT-3'te bu sayı 175 milyara çıkar. Bu tip modeller için, ince ayar işleminde bütün parametreleri eğitmek, işlem gücü ve zaman bağlamında oldukça maliyetli bir işlemdir.

DDU (*Low Rank Adaptation*) yönteminde, mevcut ön eğitim modelindeki ağırlıklar dondurulur ve dönüştürücü katmanlarına eğitilebilir ayrıştırma matrisleri yerleştirilir. Böylece, çok az sayıda eğitilebilir parametreyle daha etkin bir eğitim gerçekleştirilir [112]. Modelin doğrudan ince ayar işlemine tabi tutulması ile karşılaştırıldığında, bu yöntemle, eğitilen parametre sayısı yaklaşık binde bir, GİB bellek ihtiyacı miktarı ise yaklaşık üçte bir seviyesine kadar düşürülebilir [112].

Standard model ince ayar işleminde, üst akış görevinde elde edilen ağırlıklar da güncellenir. Kendi Kendine Denetimli Öğrenme başlığında da bahsedildiği gibi, üst akış görevinde girdilere ait temsil modelinin elde edilmesi amaçlanır. Daha sonrasında alt akış görevine uygun olarak ince ayar işlemi yürütülür. Dolayısıyla, ince ayar aşamasında ön eğitim modeline ait bütün ağırlıkların güncellenmesi gerekmez. Bu noktada, DDU yöntemi sadece göreve özgü ağırlıkların güncellemesini önerir. Yöntemin işleyişi Şekil 3.19'da gösterildiği gibidir [112].



Şekil 3.19. Düşük dereceli uyumlama yönteminin İşleyişi [112]

$W_0 \in \mathbb{R}^{d \times k}$ ön eğitim ağırlık matrisinden, $B \in \mathbb{R}^{d \times r}, A \in \mathbb{R}^{r \times k}$ ($r \ll$ en_küçük(d, k)) olmak üzere $W_0 + \Delta W = W_0 + BA$ düşük dereceli temsil elde edilir. Eğitim sırasında W_0 sabit tutulur, herhangi bir güncelleme yapılmaz ve sadece A ile B eğitime tabi tutulur. Eş. 3.41'de gösterildiği gibi, sırasıyla W_0 ve $\Delta W = BA$ aynı girdi değeriyle çarpılır ve elde edilen çıktı vektörleri üst üste binecek şekilde toplanır.

$$h = W_0 x + \Delta W x = W_0 x + BAx \tag{3.41}$$

Eğitim başlangıcında, A'ya rastgele bir Gauss değeri, B'ye ise 0 atanır. Dolayısıyla, ilk aşamada $\Delta W = BA$ sıfır değerine sahiptir. Daha sonra, bir hiper parametre (α) kullanılarak Wx, $\frac{\alpha}{r}$ ile ölçeklendirilir. Optimizasyon fonksiyonu olarak *Adam* kullanılır.

3.5. Otomatik Konuşma Tanıma Araçları

Geçmişte yaygın kullanılan ama geliştirme süreci durmuş Saklı Markov Modeli Araç Kiti ile birlikte uzun süredir gelişim süreci devam eden ve halen kullanılmakta olan *DeepSpeech, Kaldi* vb. OKT araçları mevcuttur. Teknolojideki ilerlemelere koşut olarak, bu araçların gelişimi süreci Derin Öğrenme yöntemlerini de destekleyecek şekilde devam etmektedir. Ancak, zamanla yerlerini; *Wav2vec, Whisper, Google USM* gibi yeni nesil uygulamalara bırakmaktadırlar. Bunların dışında; *Espresso, Nabu, KoSpeech, Athena* gibi yaygın kullanılmayan veya geliştirme süreci durmuş OKT araçları da mevcuttur [48]. Bu bölümde, bahsi geçen OKT araçları tanıtılmıştır.

3.5.1. Saklı markov modeli araç kiti

Saklı Markov Modeli Araç Kiti (*Hidden Markov Model Toolkit*) [36], Saklı Markov Modelleri üzerinde çalışmak için kullanılan bir yazılım paketidir. Saklı Markov Modeli Araç Kiti, konuşma tanıma, parmak izi tanıma, DDİ ve biyomedikal sinyal işleme gibi çeşitli alanlarda kullanılan bir dizi araç ve kütüphaneden oluşur. 1989 yılında *Cambridge* Üniversitesi Mühendislik Bölümünde çalışan *Steve Young* tarafından geliştirilmiştir. Son olarak 2000 yılında 3. Versiyonu yayımlanan uygulamaya ait temel bileşenler şunlardır [113]:

1. *HSLab*: Ses dosyalarını işlemek için kullanılan bir ses işleme aracıdır.
2. *HMMTools*: Saklı Markov Modelleri oluşturmak, eğitmek ve değerlendirmek için kullanılan bir araç setidir.
3. *HERest*: SMM'leri yeniden eğitmek için kullanılan bir araçtır.
4. *HVite*: Eğitilmiş SMM'leri kullanarak tanıma yapmak için kullanılan bir araçtır.
5. *HLMTools:* Dil modellerini oluşturmak ve yönetmek için kullanılan bir araç setidir.
6. *HParse*: Gramer dosyalarını işlemek için kullanılan bir araçtır.

Bu paket, OKT bağlamında; öznitelik çıkarma, Akustik Model, Dil Modeli, okunuş sözlüğü, transkripsiyon vb. bileşenlere sahiptir. SMM ve GKM modelleri içerir ve YSA temelli sistemlerde doğrudan kullanılabilir. Üst sürümleri, Derin Sinir Ağları (DSA), TSA vb. Yapay Sinir Ağı bileşenlerini destekler.

3.5.2. Kaldi

Kaldi [114], OKT sistemleri geliştirmek için kullanılan ücretsiz ve açık kaynak kodlu bir yazılım platformudur. *Kaldi*, C++ programlama dili kullanılarak geliştirilmiştir. Bununla birlikte, *Python, Bash* ve *Perl* gibi diller için yazılım geliştirme araçlarına sahiptir. *Kaldi*, OKT sistemi, bir dizi araç ve kütüphane sunar. MFKK, ADÖ vb. çeşitli öznitelik çıkarma bileşenlerinin yanında Akustik Model ve Dil Modeli, eğitim ve test bileşenlerine de sahiptir. DÖ tekniklerini destekleyen çözümleri ile birlikte gelişim süreci hâlihazırda devam etmektedir [115].

3.5.3. DeepSpeech

DeepSpeech [116], *Mozilla* firması tarafından geliştirilen açık kaynak kodlu bir otomatik konuşma tanıma aracıdır. *DeepSpeech*, büyük ölçüde DÖ yöntemlerine dayanan uçtan uca bir OKT mimarisidir. OKT ile birlikte DDİ görevlerinde kullanılmak üzere geliştirilmiş açık kaynak kodlu bir *Java* Kütüphanesi içerir ve *Python* gibi diğer programlama dillerini de destekleyecek şekilde gelişim süreci devam etmektedir.

DDİ yeteneği, elde edilen metin çıktısını işlemek ve dil işleme görevlerini gerçekleştirmek için OKT kütüphanesiyle bütünleşmiştir. Bu yetenek, konuşma tanıma sonuçlarını daha fazla işleyerek anlam çıkarma, metin analizi veya diğer DDİ görevlerini gerçekleştirmek için kullanılır [117].
Sahip olduğu metin düzenleme ve normalizasyon bileşeni, metindeki gereksiz boşlukların kaldırılması, noktalama işaretlerinin düzeltilmesi ve metin formatının düzenlenmesi gibi işlemleri yapar. Paragraf ve cümle ayrıştırma bileşeni, metin sonuçlarını paragraflara ve cümlelere ayırmak için kullanılır. Bu, daha büyük metin bloklarını daha küçük parçalara ayırarak daha ayrıntılı analiz yapmayı sağlar. Kelime ayrıştırma ve etiketleme bileşeni, metni kelimelere ayırmaya ve her kelimeye uygun dilbilgisi etiketlerini atamaya yarar. Böylece, metin üzerinde yapısal analiz ve kelime düzeyinde işlemler yapılması sağlanır.

Anlamsal Analiz ve DDİ bileşeni, metinden anlam çıkarmak, duygusal analiz yapmak, isim varlıklarını tanımak gibi dil işleme görevlerini gerçekleştirmek için kullanılır. Düşük kaynaklı sistemlerde gerçekleştirilen OKT görevlerinde *DeepSpeech* ile diğer araçlara nazaran daha yüksek doğruluğa sahip sonuçlar elde edilmiştir [48].

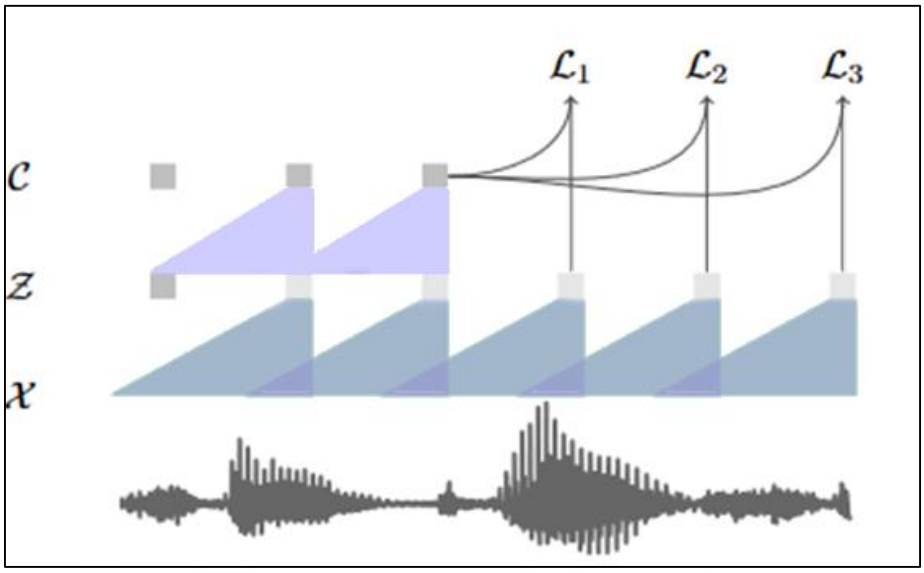
3.5.4. ESPNet

ESPnet [118], uçtan uca konuşma tanıma, metin seslendirme, konuşma çevirisi, konuşma geliştirme, konuşmacı tanıma, konuşma dili anlama vb. görevleri kapsayan uçtan uca bir konuşma işleme araç setidir. *ESPnet*, derin öğrenme motoru olarak *pytorch* kullanır. Sayılan işlemleri gerçekleştirmek için Kaldi tarzı veri işleme, öznitelik çıkarma vb. işlevleri içerir [119].

3.5.5. Wav2vec ve wav2vec 2.0

Wav2vec [42], *Facebook* firması YZ ekibi tarafından geliştirilen iki aşamalı bir konuşma tanıma uygulamasıdır. Etiketsiz veri ile temsil modelinin oluşturulması ve sonrasında temsil modeli üzerinden eğitim yapılmasını içerir. Model, Şekil 3.20'deki gibi üst üste iki ESA bölümünden oluşur. Alt bölümdeki ESA, ses alt frekans özniteliklerini çıkarır. Üst bölümdeki ESA, bağlam vektörlerinin üretilmesi için kullanılır. Hem kodlayıcı hem de bağlam ESA'ları 512 kanallıdır ve evrişim, normalizasyon ile *ReLU* aktivasyon fonksiyonu katmanlarından oluşur. Kodlayıcı ESA 5, bağlam ESA 9 katmandan meydana gelmiştir.

X, üstü üste binen çerçevelerden (pencereleme genişliği 30 ms, örtme 10 ms) oluşan ses verisini; Z, alt frekans özelliklerini; C, bağlam temsilini gösterir. L_n, bir bağlam için k. adımdaki z_k özelliğine göre hesaplanan karşılaştırmalı kaybı temsil eder.



Şekil 3.20. *Wav2vec* mimarisi [43]

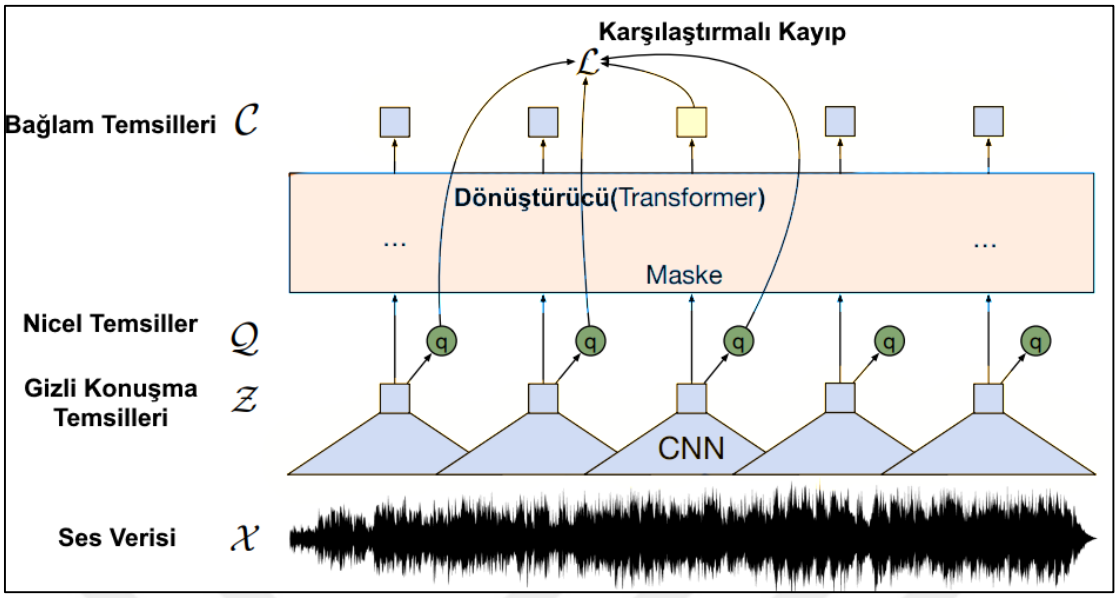
Sonraki örneklerin tahmin edilebilmesi için alt frekans öznitelikleri ile bağlam temsilleri arasına ileri yönlü bir karşılaştırma yapılır. Bunun için karşılaştırmalı kayıp (*contrastive loss*) kullanılır ve hesaplamada Eş. 3.42'den faydalanılır. Buradaki amaç, her bir k $(1, \dots, K)$ adımında, sekans içindeki her $\mathbf{c}_i$ ve z_{i+k} için toplam karşılaştırma kaybının ve nihayetinde genel kayıp $\mathcal{L}'$nin ($\mathcal{L} = \sum_{k=1}^{K} \mathcal{L}_k$) minimize edilmesidir.

$$\mathcal{L}_k = -\sum_{i=1}^{T-k} \left(\log\sigma\left(\mathbf{z}_{i+k}^{\top} h_k(\mathbf{c}_i)\right) + \lambda \mathop{\mathbb{E}}_{\tilde{\mathbf{z}} \sim p_n} \left[\log\sigma\left(-\tilde{\mathbf{z}}^{\top} h_k(\mathbf{c}_i)\right)\right] \right) \tag{3.42}$$

Eşitlikte, T sekans uzunluğunu ve λ, negatif örnek sayısını göstermektedir. $\sigma(x) = \frac{1}{(1+\exp(-x))}$ sigmoid fonksiyonunu, $h_k(\mathbf{c}_i) = W_k \mathbf{c}_i + \mathbf{b}_k$ ağırlıklandırılmış bağlam vektörü fonksiyonunu gösterir ve $p_n(\mathbf{z}) = \frac{1}{T}$'dir.

Bu aşamalardan sonra, modelin ön eğitimli modelden denetimli öğrenme yapabilmesi için *wav2letter++* ile birlikte *4-gram KenLM, Word ConvLM, Char ConvLM* gibi Dil Modelleri kullanılır. [43]

Wav2vec 2.0 [44], öncülü *Wav2vec* modelinin geliştirilmiş halidir. Şekil 3.21'de gösterildiği gibi, *Wav2vec*'e benzer şekilde üst üste iki bölümden oluşur.



Şekil 3.21. Wav2vec 2.0 mimarisi [45]

Modelde ses verisinin pencereleme genişliği 25 ms, örtme miktarı 20 ms'dir. ESA katmanı, her biri 512 kanalı 7 adet bloktan meydana gelir. Bloklar, zamansal evrişim katmanı, normalizasyon ve GELU (*Gaussian Linear Error Unit*) [120] aktivasyon fonksiyonlarından oluşur.

Wav2vec 2.0'de, *Wav2vec* mimarisinden farklı olarak üst bölümde ESA yerine dönüştürücüler kullanılmıştır. Temel modelde dönüştürücü bölümü 12 adet bloğa sahipken, büyük modelde bu sayı 24'e çıkar. *Wav2vec 2.0* mimarisinin, *Wav2vec* mimarisinden başka bir farkı, doğrudan alt frekans öznitelikleri (Z) yerine, karşılaştırma işleminin nicel temsiller (Q, *quantized representations*) ve kod defteri (*code book*) elemanları üzerinden yürütülmesidir. Bu modelde, karşılaştırmalı kayıp işlemi nicel temsillerle bağlam temsilleri arasında yürütülür ve bağlam temsillerini açıklayan en iyi nicel temsiller bulunur. Kayıp hesaplanırken, kod defterlerindeki girdilerin eşit kullanımını teşvik etmek amacıyla α hiper parametresi ile ölçeklendirilmiş $\mathcal{L}_d$ farklılık kaybı (*diversity loss*) da hesaba katılır. Kayıp fonksiyonu değeri Eş. 3.43'te gösterildiği gibi bulunur.

$$\mathcal{L} = \mathcal{L}_m + \alpha\mathcal{L}_d \tag{3.43}$$

$\mathcal{L}_d$, her kod defterine ait Entropi değerleri toplanarak Eş. 3.44'te verildiği gibi hesaplanır.

$$\mathcal{L}_d = \frac{1}{GV}\sum_{g=1}^{G} - H\left(\bar{p}_g\right) = \frac{1}{GV}\sum_{g=1}^{G}\sum_{v=1}^{V}\bar{p}_{g,v}\log\bar{p}_{g,v} \tag{3.44}$$

$\tilde{\mathbf{q}} \in \mathbb{Q}_t$, t anı ve öncesindeki K adet (toplamda K+1 adet) nicel temsili ve $sim(\mathbf{a},\mathbf{b}) = \mathbf{a}^T\mathbf{b}/|\mathbf{a}||\mathbf{b}|$ nicel ve bağlam temsilleri arasındaki kosinüs benzerliğini ifade eder. Buna göre, $\mathcal{L}_m$ Eş. 3.45'teki hesaplamayla elde edilir.

$$\mathcal{L}_m = -\log\frac{\exp(sim(\mathbf{c}_t,\mathbf{q}_t)/\kappa)}{\sum_{\tilde{\mathbf{q}}\sim\mathbf{Q}_t}\exp(sim(\mathbf{c}_t,\tilde{\mathbf{q}})/\kappa)} \tag{3.45}$$

Aşırı öğrenmenin önüne geçmek adına, ESA bloğu ile dönüştürücü bloğu arasında maskeleme yapılır. Nicel temsiller oluşturulurken, değişebilen boyutlardaki matrislerden oluşan kod defterleri (*code book*) kullanılır. Niceleme işleminde *Gumbel softmax* fonksiyonundan faydalanılır. G kod defterlerini, v her bir defterden seçilen öğe numarasını gösterir. $e \in \mathbb{R}^{V\times d/G}$ olmak üzere, seçilmiş öğelerden oluşan kod sözcüğü vektörü $q = [e_1, \ldots, e_G]$'dir. Her iki *Wav2vec 2.0* model türü (*base, large*) için grup sayısı 2, gruptaki girdi sayısı 128 ve olası kod sözcüğü sayısı da 128^2'dir. Özniteliklerden elde edilen g. kod defterindeki v. girdi ($l_{g,v} \in \mathbb{R}^{G\times V}$) olasılığı Eş. 3.46'daki gibi hesaplanır.

$$p_{g,v} = \frac{\exp(l_{g,v}+n_v)/\tau}{\sum_{k=1}^{V}\exp(l_{g,k}+n_k)/\tau} \tag{3.46}$$

Burada, örnek grubuna ait *Gumbel* gürültü $u = \mathcal{U}(0,1)$ olmak üzere $n = -\log(-\log(u))$ şeklinde ifade edilir ve τ sıcaklık katsayısını gösterir. İleri yayılımda $i = \text{argmax}_j p_{g,j}$ üzerinden seçim yapılırken, geri yayılımda *Gumbel softmax* fonksiyonunun türevi kullanılır. Son noktada, elde edilen bağlam temsilleri üzerinden BZS kullanılarak ince ayar yapılır. Öncülü *Wav2vec*'in aksine, bu mimaride, üst sınıflandırıcı ya da dil modeli kullanımına ihtiyaç duyulmaz [45].

3.5.6. Whisper

Whisper [6], *Open AI* firması tarafından geliştirilen çok dilli ve çok görevli bir mimaridir. Adı, İngilizce ***W**eb-scale **S**upervised **P**re-training for **S**peech **R**ecognition* (Konuşma Tanıma için Web Ölçekli Denetimli Ön Eğitim) ifadesindeki kelimelerden esinlenerek verilmiştir. OKT ve otomatik çeviri aracı olarak kullanılır. Bu mimariyle, ince ayar

yapılmasına gerek duyulmadan, güvenilir bir şekilde çalışan bütünleşik bir konuşma işleme modeli geliştirmesi amaçlanmıştır. Yapısı itibariyle herhangi bir ek eğitim (ince ayar) gerektirmeyen hazır (tak-çalıştır) bir mimaridir.

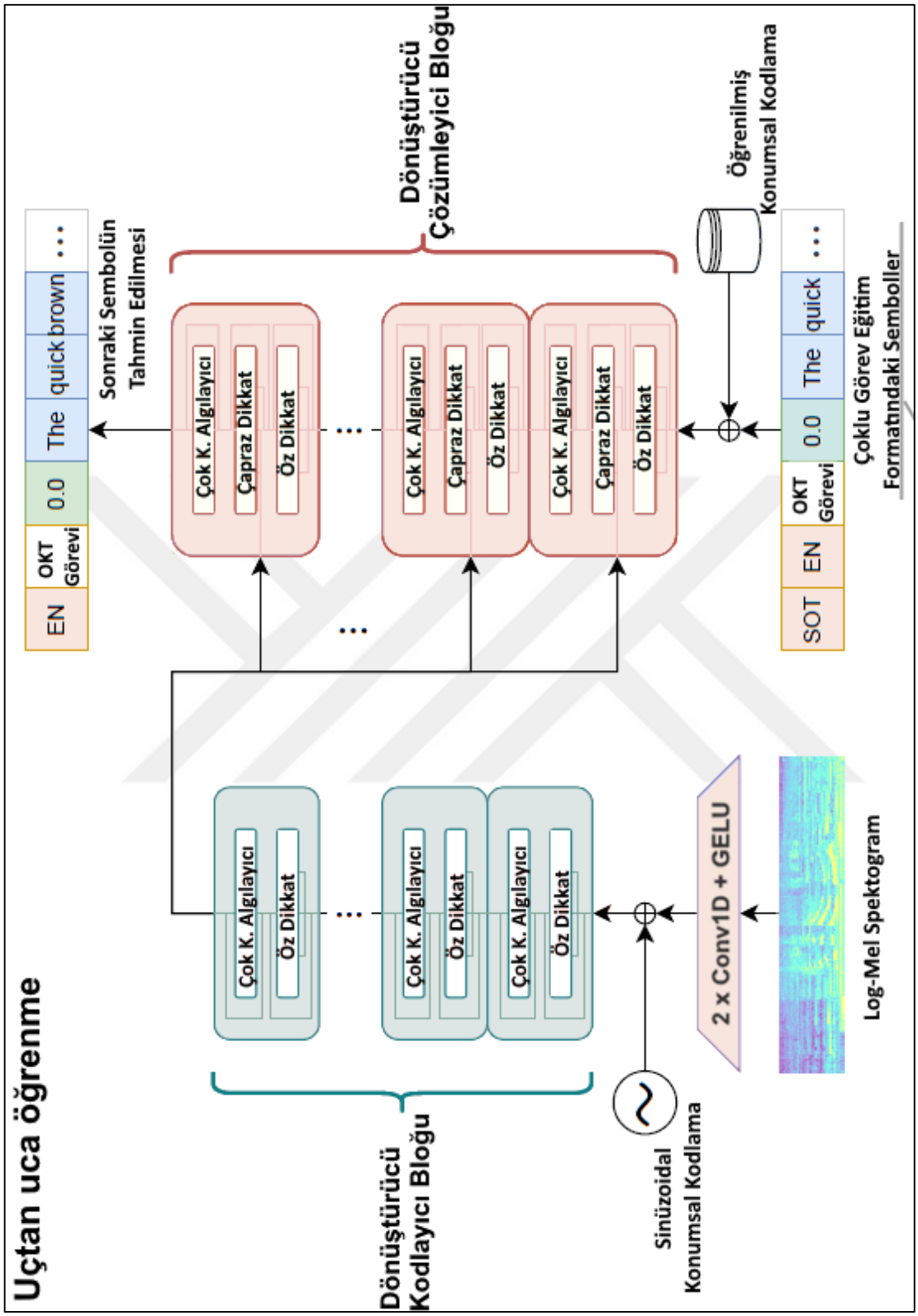
Ön eğitim aşamasında, özel belirteçler kullanılarak modelin çoklu görev için (İngilizce konuşma tanıma, başka dillerde konuşma tanıma, çeviri, dil tespiti vb.) eğitilmesi sağlanmıştır [6]. Modellerin ön eğitiminde, yaklaşık 453000 saati İngilizce, kalan yaklaşık 117000 saati 96 farklı dilde toplam 680000 saatlik konuşma verisi kullanılmıştır. Bunun dışında, 125000 saatlik diğer dillerden İngilizceye çeviri metin verisinden de faydalanılmıştır.

Whisper mimarisi, büyüklüğüne göre 5 farklı model yapılandırmasına sahiptir. Küçükten büyüğe doğru model adları ve özellikleri Çizelge 3.3'te verildiği gibidir.

Çizelge 3.3. Whisper model türleri ve yapılandırma parametreleri [6]

Model	Katman Sayısı	Genişlik	Dikkat B. Sayısı	Parametre Sayısı
Tiny	4	384	6	39 Milyon
Base	6	512	8	74 Milyon
Small	12	768	12	244 Milyon
Medium	24	1024	16	769 Milyon
Large	32	1280	20	1550 Milyon

Ses verisinin tamamı 16 kHz frekansında tekrar örneklenerek kullanılmıştır. 25 ms pencereleme, 10 ms kaydırmayla ses verisinden elde edilen 80 kanal MFKK ile model eğitilmiştir. *Whisper*'a ait mimari Şekil 3.22'de gösterildiği gibidir.



Şekil 3.22. Whisper mimarisi [6]

Girdi kodlayıcı, zaman serisi şeklindeki verilerin işlenmesinde kullanılan 1 boyutlu 2 ESA ve GELU aktivasyon fonksiyonu katmanlarından meydana gelmiştir. Dönüştürücü tarafında kodlayıcı ve çözümleyici aynı blok sayısına ve genişliğine sahiptir. Dönüştürücüde ön-aktivasyon artıklı bloklar (*pre-activation residual block*) [121] kullanılmıştır. ESA yapısı tarafından işlenmiş veri, sinüzoidal konumsal kodlama bilgisi de eklenerek kodlayıcı bloğuna aktarılır. Çözümleyici tarafında, öğrenilmiş konumsal kodlama bilgileri belirteçlerle beraber çözümleyiciye beslenir.

Temel çalışmada, on dört adet İngilizce veri kümesi kullanılarak, *Wav2vec 2.0-larger* ve *Whisper (large)* modelleri arasında KHO ölçütü üzerinden performans karşılaştırması yapılmış ve Çizelge 3.4'te verilen sonuçlar elde edilmiştir [6].

Çizelge 3.4. Wav2vec 2.0-large ve Whisper karşılaştırmalı KHO tablosu [6]

	Kelime Hata Oranı		
Veri Kümesi	**Wav2vec 2.0-large (DM olmadan)**	**Whisper**	**İyileşme Oranı**
LibriSpeech Clean	2,7	2,7	0
Artie	24,5	6,2	74,7
Common Voice	29,9	9	69,9
Fleurs (İng)	14,6	4,4	69,9
Tedlium	10,5	4	61,9
CHiME6	65,8	25,5	61,2
VoxPopuli (İng)	17,9	7,3	59,2
CORAAL	35,6	16,2	54,5
AMI IHM	37	16,9	54,3
Switchboard	28,3	13,8	51,2
CallHome	34,8	17,6	49,4
WSJ	7,7	3,9	49,4
AMI SDM1	67,6	36,4	46,2
LibriSpeech (Diğer)	6,2	5,2	16,1
Ortalama	29,3	12,8	55,2

Çalışma sonucunda; *Whisper*'ın, *Wav2vec 2.0-large* mimarisine nazaran ortalama %55,2 oranında KHO iyileşmesi sağladığı tespit edilmiştir. Ayrıca, *Common Voice 9.0* ve

FLEURS [84] veri kümelerindeki Türkçe içerikle yapılan testlerde, Çizelge 3.5'te verilen KHO değerleri elde edilmiştir [6].

Çizelge 3.5. Türkçe veri kümeleri için Whisper modellerinin KHO değerleri [6]

Model	***Common Voice* (9.0)**	***FLEURS***
Whisper tiny	53,6	42,5
Whisper base	38,6	27,5
Whisper small	23,7	15,9
Whisper medium	17,7	10,4
Whisper large	16,6	9,4
Whisper large-v2	14,5	8,4

3.5.7. Google USM

Google USM (*Universal Speech Model*) mimarisi [51], OKT ve otomatik çeviri görevlerinde kullanılan, 100'den fazla dili destekleyen güncel çözümlerden biridir. 300'den fazla dildeki 12 milyon saatlik konuşma verisiyle denetimsiz ön eğitim yapıldıktan sonra nispeten daha küçük bir etiketli konuşma veri kümesi ile ince ayar yapılarak eğitilmiştir. Kaynak makale kapsamında yapılan değerlendirmeye göre, etiketli veri kümesinin 1/7'si kadar veriyle ince ayar yapıldığı halde *Whisper* modeline nazaran daha başarılı sonuçlar elde edilmiştir [51].

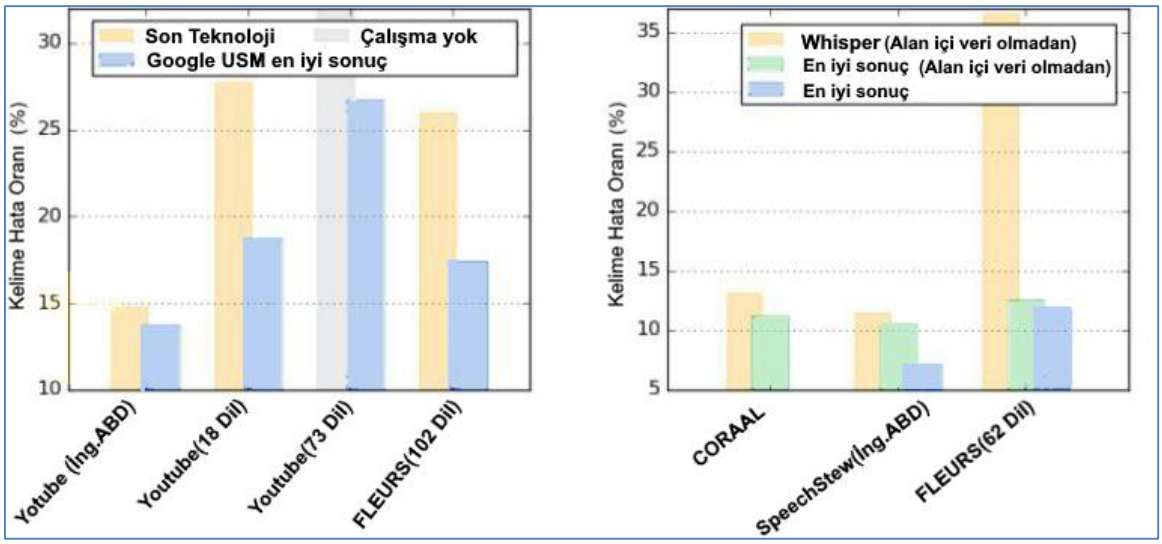
Ön eğitim aşamasında *Youtube'*da yer alan 300'den fazla dildeki 12 milyon saatlik etiketsiz ses verisinden oluşan *YT-NTL-U* ve 51 dilde 429 bin saat etiketsiz ses verisinden oluşan *Pub-U* veri kümeleri kullanılmıştır. Metin veri kümesi olarak, 140 dilde 28 milyar cümle içeren *Web-NTL* metin külliyatından faydalanılmıştır. İnce ayar sürecinde, denetimli öğrenme için 73 dilde 90 bin saatlik *YT-SUP* etiketli ve *YT-NTL-U*'dan elde edilen 100 bin saatlik sözde etiket atanmış veri kümeleri ile 102 dilde 10 bin saatlik etiketli veri içeren *PUB-S* veri kümesi kullanılmıştır.

Eğitim aşamasında, 2 Milyar parametre içeren *Conformer* [46] modeli Best-RQ (*BERT-based Speech Pre-training With Random Projection Quantizer*) denetimsiz öğrenemeye tabi tutulmuştur. Sonrasında, ses ve metin veri kümeleriyle, metin enjeksiyonu da (*text-*

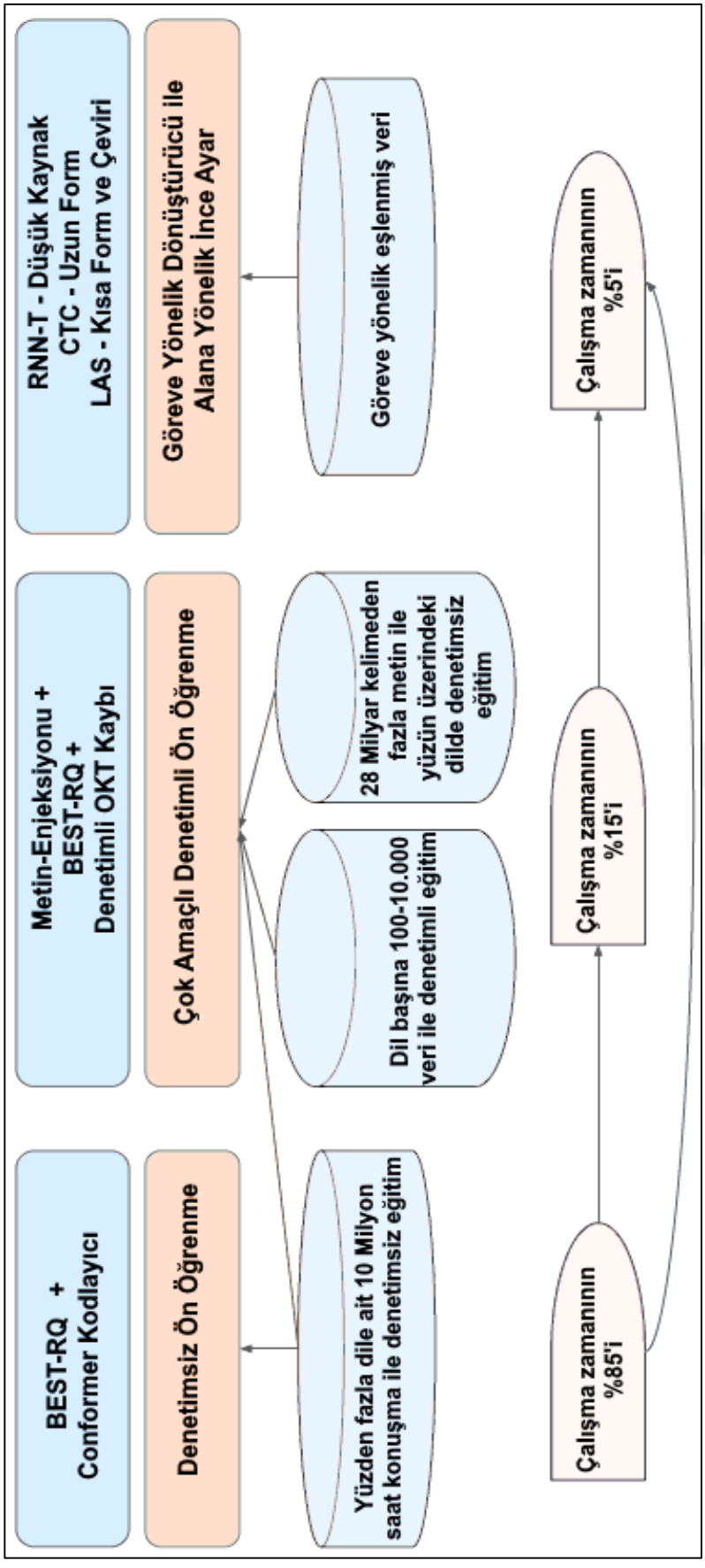
injection) [122] kullanılarak, çok amaçlı denetimli ön eğitim (*MOST- Multi-Objective Supervised Pre-training*) işlemi yapılmıştır. Son olarak, etiketli veri kümesi ile iki farklı yapılandırma (DÖK, *LAS* [40]) kullanılarak denetimli öğrenme gerçekleştirilmiştir.

Bu mimariyi farklı kılan özelliklerden biri, *BEST-RQ* eğitimi sırasında çoklu kod defteri ve çoklu softmax (*multi-softmax*) kullanılmasıdır. Bir diğer farklı özellik, modelde *Noisy Student Training* [123] yarı denetimli öğrenme yönteminden faydalanılmasıdır. Bu yöntem, kısıtlı etiketli veri ile birlikte sözde etiket (*pseudo-label*) atanan etiketsiz verinin kullanılmasını, dolayısıyla daha geniş bir kümeyle denetimli öğrenme yapılmasını sağlamıştır. Mimarideki farklı özelliklerden sonuncusu, yerel dikkat mekanizması yerine yığın temelli dikkat mekanizması (*chunk-wise self-attention*) kullanılmasıdır. Bu şekilde, konuşma verilerindeki performans kayıplarının azaltılması amaçlanmıştır.

Model eğitimi, Şekil 3.24'te gösterildiği gibi üç aşamadan meydana gelmektedir [51]. Birinci aşamada, *Conformer* ve *Best-RQ* ile farklı dillerde 12 milyon saatlik konuşma verisiyle denetimsiz öğrenme yapılmıştır. İkinci aşamada, MOST kullanılarak, etiketli konuşma verisi ve metin verisi ile denetimli ve denetimsiz öğrenme gerçekleştirilmiştir. Son aşamada, alt akış görevi doğrultusunda *RNN-T* [124], DÖK, *LAS* ile ince ayar yürütülmüştür. Şekil 3.23'te *Google USM* ve *Whisper* çözümleri için farklı veri kümeleriyle, KHO üzerinden yapılan karşılaştırılma gösterilmektedir.



Şekil 3.23. Farklı veri kümeleri için, KHO üzerinden Google USM ve Whisper çözümlerinin karşılaştırması [51]



Şekil 3.24. Google USM modeli eğitim aşamaları [51]

OKT test aşamasında *SpeechStew* [125], *FLEURS* [84], *CORAAL* [126]; otomatik çeviri için CoVoST 2 [127] veri kümelerinden faydalanılmıştır. Şekilde de görüleceği üzere ölçümlerde, *Google USM* ile özellikle çok dilli veri kümelerinde *Whisper*'a nazaran daha başarılı (daha düşük hata oranına sahip) sonuçlar elde edildiği görülmektedir [51].

3.6. Otomatik Konuşma Tanıma Başarım Değerlendirme Ölçütleri

OKT sistemleri ile oluşturulan bir metinde karşılaşılabilecek olası üç hata çeşidi ekleme, silme ve değiştirmedir. Ekleme, metinde olmayan bir sembolün/kelimenin fazladan metne eklenmesidir. Silme, metinde belirli bir konumda olması gereken bir sembolün/kelimenin ilgili yerde bulunmamasıdır. Değiştirme ise, metin içinde belirli bir konumda olması gereken sembolün/kelimenin yerinde, başka bir sembolün/kelimenin bulunmasıdır.

Bu bağlamda, N_K adet kelimeden oluşan bir metin için E_K eklenen, S_K silinen, D_K değiştirilen kelime sayılarını gösterdiğini kabul edersek, Kelime Hata Oranı (KHO, *Word Error Rate*) Eş. 3.47'de ve Kelime Tanıma Oranı (KTO) olarak da ifade edilen doğruluk Eş. 3.48'de verildiği gibi hesaplanır.

$$\mathrm{KHO} = \frac{E_K + S_K + D_K}{N_K} \tag{3.47}$$

$$\mathrm{KTO} = 1 - \mathrm{KHO} \tag{3.48}$$

OKT değerlendirme ölçütü olarak genellikle KHO kullanılır. Yaygın kullanılan ölçütlerden bir başkası Karakter Hata Oranıdır (KaHO, *Character Error Rate*). KaHO değeri, KHO değerine benzer şekilde hesaplanır. N_{Ka} adet sembolden oluşan bir metin için E_{Ka} eklenen, S_{Ka} silinen, D_{Ka} değiştirilen sembol sayısını gösterdiği durumda KaHO Eş. 3.49'da ve Karakter Tanıma Oranı (KaTO) Eş. 3.50'de verildiği gibi bulunur.

$$\mathrm{KaHO} = \frac{E_{Ka} + S_{Ka} + D_{Ka}}{N_{Ka}} \tag{3.49}$$

$$\mathrm{KaTO} = 1 - \mathrm{KHO} \tag{3.50}$$

KHO ve KaHO hesaplamalarındaki benzerlikleri nedeniyle genellikle birbiri ile karıştırılır. KHO, hatalı kelime sayısı üzerinden ölçülürken; KaHO'da hatalı sembol sayısı dikkate alınır. Aşağıdaki ifadeyi ele alırsak:

Bir işi yapmak için neden yarını bekliyorsun bugün de dünün bir yarını değil midir

14 kelime ve boşluklarla beraber 82 sembolden oluşan tümce için OKT sisteminin ürettiği çıktının aşağıdaki gibi olduğunu varsayalım.

Biri işi yapmak işin neden yarın bekliyorsun bugün de dünün bir yarını değil

Birinci kelimede sembol ekleme, dördüncü kelimde sembol değişikliği, altıncı kelimede sembol silme işlemleri yapılmış ve son kelime tamamen silinmiştir. 7 silme, 1 değiştirme, 1 ekleme, toplam 9 sembol hatası; 3 değiştirme, 1 silme toplam 4 kelime hatası vardır. Bu durumda, $KHO = \frac{4}{14} \cong 0{,}286$, $KaHO = \frac{9}{82} \cong 0{,}089$ olur.

Bunların dışında, yaygın olmamakla birlikte, Cümle Hata Oranı *(SER, Sentence Error Rate)*, Fonem Hata Oranı *(PER, Phone Error Rate)*, İfade Hata Oranı *(UER, Utterance Error Rate)*, Çerçeve Hata Oranı *(FER, Frame Error Rate)* gibi ölçütler de OKT başarım ölçümünde kullanılabilmektedir. Düşük hata oranları yüksek doğruluğu dolayısıyla yüksek model başarısını gösterir.

4. DENEYSEL ÇALIŞMA VE BULGULAR

Bu bölümde, deneysel çalışma kapsamında yapılan işlerle ilgili bilgiler verilmiştir. Öncelikle, deneysel çalışma ortamı ve veri kümeleri üzerinde yapılan ön hazırlık işlemleri anlatılmıştır. Ardından, veri düzenleme öncesinde ve sonrasında gerçekleştirilen deneylere ait karşılaştırmalı sonuçlar paylaşılmıştır. Son olarak, ince ayar işlemleri ve eş zamanlı konuşma tanıma uygulamalarıyla gerçekleştirilen deneylerin sonuçları sunulmuştur.

4.1. Deneysel Çalışma Ortamı

Deneysel çalışmaların bir bölümü, TÜBİTAK ULAKBİM bünyesinde yer alan ve TRUBA olarak bilinen Yüksek Başarımlı Grid Hesaplama Merkezindeki sistemler kullanılarak gerçekleştirilmiştir. TRUBA sisteminde yapılan deneylerde, *Akya-cuda* (**Sistem-1**) olarak adlandırılan sunucu kümesi kullanılmıştır. *Akya-cuda*, her birinde 20 çekirdekli 2 adet *Intel Xeon* 6148 2,40 GHz MİB ve 4 adet *Nvidia V100* GİB 24 adet sunucudan oluşan bir hesaplama kümesidir. Kayıtlı kullanıcıların talepleri iş kuyruklarına alınarak, istenen kaynak boşa çıktığında işlemlerin yürütüldüğü, paylaşımlı bir yapıya sahiptir. Bu çalışma kapsamında, çoğunlukla 20 çekirdekli 1 Merkezi İşlem Birimi ve 1 adet *Nvidia* V100 GİB **(Sistem-1)** kaynağı kullanılmıştır. Bunun dışında, 32 çekirdekli *Intel*(R) *Xeon*(R) *Gold* 6426Y 2.50 GHz 2 işlemci ve *Nvidia* A5000 GİB'e sahip iş istasyonundan (**Sistem-2**) faydalanılmıştır.

Geliştirilen uygulamalarda, *Python* programlama dili ve *Miniconda* ortamı kullanılmıştır. *Miniconda, Python* programlama dili ile birlikte kullanılan dağıtım ve paket yönetim sistemidir. *Anaconda* şirketi tarafından geliştirilen *Miniconda*, bilimsel hesaplama, veri analizi ve makine öğrenimi gibi veri bilimi alanındaki projelerde kullanılmak üzere tasarlanmıştır.

Veri kümeleri ile yapılan testlerde *Facebook* firması tarafından geliştirilen makine öğrenmesi ve derin öğrenme kütüphanesi *pytorch* kullanılmıştır. Ses dosyalarının işlenmesi için *pydub* kütüphanesinden yararlanılmıştır. *OpenAI* firması tarafında geliştirilen *Whisper* OKT modeli kütüphanesi ve bu kütüphane üzerine geliştirilen *fast-whisper, Whisper-Streaming* ve *WhisperLiver* uygulamalarından faydalanılmıştır. Whisper OKT'nin

sayıları rakam olarak metne çevirmesi sorununu ortada kaldırmak için *num2words* kütüphanesi, üzerinde değişiklikler yapılarak, kullanılmıştır. Veri işleme ve hesaplamalar için *numpy* ve *pandas*, kelime ve karakter hata oranlarının hesaplanması için *jiwer* kütüphaneleri uygulamaya dâhil edilmiştir. Ayrıca, süre tespiti için *datetime,* dosya işlemleri için *io, sys ve tarfile,* metin işleme ve düzenli ifadeler için *re,* uygulama parametrelerinin yönetimi için *argparse* standart kütüphaneleri kullanılmıştır.

4.2. Veri Kümelerinin Hazırlanması

Deneysel çalışma kapsamında 3.3. OKT Veri Kümeleri bölümünde açıklanan ODTÜ MK, THKM, *Mozilla CV*, *FLEURS* ve TOKTT veri kümeleri kullanılmıştır. ODTÜ MK, THKM, TOKTT ve *FLEURS* veri kümeleri 16 kHz örneklem frekansına sahip *WAV* uzantılı, *Mozilla CV* veri kümesi 48 kHz örnekleme frekansına sahip *MP3* uzantılı ses dosyalarından oluşmaktadır.

Yapılan ilk değerlendirmede, ODTÜ MK veri kümesinde yer alan sırasıyla 8 dakika 55 saniye ve 8 dakika 19 saniye uzunluğa sahip *odtu_Sound1026.wav, odtu_Sound2518.wav* ses dosyalarının, metin dosyaları ile eşleşmediği görülmüştür. Bu nedenle ilgili kümeden çıkarılmışlardır.

Deneysel çalışmaların yürütüldüğü TRUBA sistemindeki dosya sayısı sınırlaması nedeniyle, ses ve metin dosyalarının ayrık olduğu veri kümeleri, arşiv dosyası olacak şekilde ayrı ayrı birleştirilmiştir. Her bir veri kümesi için metin dosyaları, Çizelge 4.1 ve Çizelge 4.2'deki gibi ses dosyaları ile eşleşecek şekilde *TSV* dosyalarına aktarılmıştır. Daha sonra, ilgili *TSV* dosyalarıyla ses dosyaları birleştirilerek *TGZ* formatında sıkıştırılmış tek bir dosya haline getirilmiştir. *FLEURS* ve *Mozilla CV* veri kümelerinin hâlihazırda istenen yapıda olması sebebiyle, bu kümeler üzerinde ek bir düzenleme yapılmamıştır.

Çizelge 4.1. ODTÜ mikrofon konuşması veri kümesi *tsv* dosyası deseni

DOSYA NO	SES DOSYASI	METİN
1	odtu_Sound1.wav	sevgili dinleyenlerimiz
10	odtu_Sound10.wav	bu çerçevede
100	odtu_Sound100.wav	güle güle
1000	odtu_Sound1000.wav	şayet basketbol takımında görev almak istiyorsan
1001	odtu_Sound1001.wav	haftasonları spor salonunda arkadaşlarınla çalışmalısın

Çizelge 4.2. THKM veri kümesi *tsv* dosya deseni

DOSYA NO	SES DOSYASI	METİN
1	bogazici_Sound1.wav	bu saldırıyla birlikte gazzede artan şiddet olayları kudüse de sıçramış oldu
10	bogazici_Sound10.wav	şimdi kudüsteki son gelişmeleri almak üzere
100	bogazici_Sound100.wav	bu açılımların başında kerkükün özel bir statüye sahip olması koşuluyla
1000	bogazici_Sound1000.wav	burası amerikanın sesi washington
10000	bogazici_Sound10000.wav	fransayı ziyareti sırasında

Deneysel çalışmada, işlenen veri kümesine ait belirlenen oranda test verisi, eşleştirilmiş metin ve dosya bilgileri çalışma anında veri tutuculara aktarılmıştır. Sonrasında, ilgili ses dosyaları, dosya adı anahtar olacak şekilde farklı veri tutuculara yüklenmiştir. *Mozilla CV* veri kümesine ait *MP3* formatındaki dosyalar tek kanal ve 16 kHz örneklem frekansına sahip olacak şekilde dönüştürülmüştür.

4.3. Metin Düzeltme Sonrası Model Performanslarının Ölçümü

Çizelge 4.3'te, her bir veri kümesinde, 5 ölçekteki modeller için (*tiny, base, small, medium, large*), kümeler üzerinde herhangi bir düzenleme yapılmadan yürütülen ön testlere ait KHO, KaHO, saat ve dakika cinsinden süre değerleri yer almaktadır. Testte, kayıt sayısı az olan ODTÜ MK, TOKTT ve *FLEURS* veri kümelerindeki örneklerin tamamı, kayıt sayısı çok olan THKM ve *Mozilla CV* veri kümelerindeki örneklerin %30'u kullanılmıştır.

Whisper çıktılarında, sayılar rakama dönüştürülmektedir. İçerdiği *tokenizer* modülüyle modelin sayıları rakam olarak metne aktarması tam olarak engellememektedir. Tutarsızlığı önlemek adına, eşleştirilmiş metin ve OKT sistemi çıktısındaki rakamlar, *num2words*

kütüphanesiyle normalizasyon işlemine tabi tutularak yazıya dönüştürülmüştür. İlgili kütüphane, mevcut TDK yazım kurallarına göre değişiklik yapmadığı için kütüphane üzerinde değişiklik yapılması gerekmiştir. Sonrasında, veri kümesi ve OKT sisteminden elde edilen çıktı metinlerindeki çoklu boşluklar teke düşürülmüştür. Ayrıca, karşılaştırma sırasında sorun yaratabilecek büyük Türkçe karakterler Ç, I, İ, Ğ, Ö. Ş, Ü küçük harfe dönüştürülmüştür. Yapılan testlerde, Çizelge 4.3'te yer alan sonuçlar elde edilmiştir.

Çizelge 4.3. Ön değerlendirme KHO, KaHO ve çalışma süresi ölçümü

Sistem-1	**Ölçüt**	***tiny***	***base***	***small***	***medium***	***large-v2***
ODTÜ MK ***(%100, 6.168 kayıt)***	**KHO**	0,36	0,24	0,16	0,15	**0,10**
	KaHO	0,10	0,07	0,06	0,04	**0,03**
	Süre	00:20	00:21	00:25	00:36	**00:47**
THKM ***(%30, 20.583 kayıt)***	**KHO**	0,53	0,36	0,22	0,15	**0,13**
	KaHO	0,18	0,12	0,08	0,06	**0,05**
	Süre	01:30	01:33	01:57	02:44	**03:35**
FLEURS ***(%100, 3.127 kayıt)***	**KHO**	0,47	0,31	0,17	0,11	**0,08**
	KaHO	0,15	0,09	0,04	0,03	**0,02**
	Süre	00:16	00:18	00:22	00:31	**00:43**
Mozilla CV ***(%30, 15.743 kayıt)***	**KHO**	0,49	0,37	0,24	0,19	**0,16**
	KaHO	0,22	0,16	0,12	0,11	**0,09**
	Süre	00:52	00:56	01:05	01:28	**01:50**
TOKTT ***(%100, 286 kayıt)***	**KHO**	0,41	0,29	0,19	0,14	**0,13**
	KaHO	0,14	0,09	0,06	0,05	**0,04**
	Süre	00:03	00:03	00:04	00:07	**00:10**

Whisper-large-v2 modeli ile yapılan testlerde, tüm veri kümleri için oldukça düşük KaHO ve KHO değerlerine ulaşılmış, başarılı sonuçlar elde edilmiştir. Bununla birlikte, gerek veri kümelerinden gerekse *Whisper* OKT sisteminden kaynaklanan çeşitli sorunların KHO ve KaHO değerlerinin yükselmesine, dolayısıyla başarının düşmesine sebep olduğu tespit edilmiştir. Bu sorunlar aşağıda sıralandığı gibidir:

1. Çizelge 4.4'te örnekleri verilen ODTÜ MK veri kümesindeki bazı metinler Türk Dil Kurum (TDK) yazım kurallarına uygun değildir ya da doğru seslendirilmemiştir. Bu durumda, OKT sistemi doğru metni üretse de, veri kümesindeki yazım hataları nedeniyle KaHO ve KHO değerleri olması gerektiğinden daha yüksek çıkmıştır. Özellikle tarihlere

ait metinlerin boşluksuz yazımı, “bir şey”, “her şey” gibi ifadelerin bitişik, “birçok” gibi ifadelerin ayrı; “–mı”, “-mi” , “-de”, “-da”, “-ki” gibi ek ve bağlaçların bitişik şekilde çıktıya yansıtılması genel yazım hatalarındandır. Az da olsa, veri kümesinde rastlanan başka bir hata türü ç, ğ, i, ö, ş, ü gibi Türkçe karakterlerin noktasız yazılmasıdır.

Çizelge 4.4. ODTÜ MK veri kümesindeki metin örnekleri

DOSYA NO	METİN
1001	haftasonları spor salonunda arkadaşlarınla çalışmalısın
1062	kavaltılık birşeyler alacağım bir de dolmalık biber
1063	ayrıca salçalık dometes ve
1214	öğleden sonra her saat müsaitim
1152	afedersiniz tren saat kaçta kalkıyor
1225	bir zamana ve kişiye bağlmak için

2. Benzer şekilde THKM veri kümesinde, Çizelge 4.5’te örnekleri verilen çok fazla sorunlu kayıt bulunmaktadır. Bitişik kelimelerin ayrı, ayrı yazılması gereken kelimelerin bitişik yazılması ve özellikle yabancı dildeki birçok kelimenin Türkçe okunuşları ile birlikte yer alması KHO ile KaHO değerlerinin artmasına sebep olmuştur.

Çizelge 4.5. THKM veri kümesindeki hatalı kayıt örnekleri

DOSYA NO	METİN
16984	yirmi bin **megawattmegavat** elektrik üretimi yapılmasını istiyor
17	polis sözcüsü **mickeymiki rosenfeldrozınfeld** saldırganın kalaşnikof marka silah taşıdığını
16999	karşılıklı **açıklmalar** bir ara oldukça sertleşti ve Amerika’nın
17008	ken **katzmankatzmın** yeni istihbarat raporunun
17021	kongre **arştırmaları** merkezinden ken katzmankatzmın
17141	**arkansasarkansa** eski valisi **mikemayk** huckabeehakabe ise

Veri kümelerindeki metinler, belirsizlik olan durumlarda ses dosyaları dinlenerek tekrar düzenlenmiştir. THKM veri kümesinde yer alan, Türkçe okunuşları ile birlikte yan yana yazılmış yabancı isimler, sadece yaygın kullanılan karşılıkları yer alacak şekilde metinde teke düşürülmüştür.

Bunların dışında, hem metin dosyaları hem de mevcut modelin işleyişi ile ilgili karşılaşılan diğer sorunlar aşağıda sıralandığı gibidir:

1. Bazı dosyalarda, konuşma bitmeden kayıt sonlandırıldığı için üretilen çıktıda hatalar oluşmuştur. Özellikle birden fazla kişiye ait konuşmanın olduğu kayıtlarda, arada boşluk varsa, sonraki konuşmacılara ait konuşmanın model tarafından işlenmediği görülmüştür. Bu nedenle, hataya sebep olan boşlukları azaltmak adına Ses Etkinlik Tespiti (*Voice* activation detection, VAD) [128] filtresi uygulanmıştır.
2. "İle", "ise", "de", "da" gibi bağlaçların kelimeyle bitişik ya da ayrı yazılması konusundaki karışıklık sık karşılaşılan hatalardandır. Metinde, dinleyen için bile tespit edilmesi zor olan ek ya da bağlaçlar, kimi yerlerde yanlış yazılmış ya da model tarafından yanlış dönüştürülmüştür. Belirgin şekilde ayrı ya da bitişik olması gerektiği noktalarda metin düzeltilmiştir.
3. Önceki kelimenin sonu ile sonraki kelimenin başı benzer seslerden oluştuğunda; model, bazen sonraki kelimenin başını atlayarak öncekiyle birleştirilmektedir.
4. Model doktor, profesör, vesaire, kilometre gibi ifadeleri kısaltarak Dr., Prof., vs., km şeklinde metne dönüştürmektedir. Ayrıca, tüm sayıların rakama, ondalık sayılarda yer alan virgül ya da nokta şeklindeki seslendirilen bölümlerin doğrudan sembole dönüştürülmesi, karşılaştırmada tutarsızlığa sebep olmaktadır. Bu nedenle, model tarafında üretilen çıktılarda bu kısaltmalar tekrar uzun metinlere, rakamlar yazıya dönüştürülmüştür.
5. İki veya daha az kelimeden oluşan ya da sadece kısa nida ifadelerinin yer aldığı ses kayıtlarında modelin hatalı çıktı ürettiği görülmüştür.

ODTÜ MK, THKM ve TOKTT veri kümeleri tekrar düzenlendikten sonra yapılan ölçümlerde elde edilen sonuçlar Çizelge 4.6'da yer aldığı gibidir.

Metinlerde düzenleme yapıldıktan sonra *Whisper-large-v2* modeli kullanılarak yapılan deneylerde, ODTÜ MK veri kümesinde 0,06; THKM ve TOKTT veri kümelerinde 0,10 KHO değerleri elde edilmiştir.

Çizelge 4.6. Metin düzenlemesi sonrası oluşan KHO, KaHO ve çalışma süresi

Sistem-1	**Ölçüt**	***tiny***	***base***	***small***	***medium***	***large-v2***
ODTÜ MK ***(%100, 6.168 kayıt)***	**KHO**	0,33	0,21	0,12	0,11	**0,06**
	KaHO	0,09	0,06	0,04	0,04	**0,02**
	Süre	00:20	00:21	00:24	00:35	**00:47**
THKM ***(%30, 20.583 kayıt)***	**KHO**	0,52	0,35	0,20	0,14	**0,10**
	KaHO	0,17	0,10	0,06	0,05	**0,04**
	Süre	01:13	01:18	01:37	02:15	**02:57**
TOKTT ***(%100, 286 kayıt)***	**KHO**	0,38	0,25	0,15	0,09	**0,08**
	KaHO	0,13	0,08	0,05	0,03	**0,03**
	Süre	00:03	00:03	00:04	00:07	**00:10**

4.3.1. Whisper large-v3 modeli

06.11.2023'de tarihinde *OpenAI* firması tarafından *Whisper-large-v3* modeli tanıtılmıştır. Bu model, *Whisper-large* ve *Whisper-large-v2* modelleriyle aynı katman, dikkat başlığı ve parametre sayısına sahiptir. Bununla birlikte, girdi olarak 80 yerine 128 *Mel* frekans bandı kullanması ve tanımlı diller grubuna Kantonca'nın da eklenmesi ile diğer modellerden ayrılır. *Whisper-large-v3* modeli, *Whisper-large-v2* modelinin 1 milyon saat zayıf etiketli, 4 milyon saat sözde etiketli veriyle eğitilmesi sonucu elde edilmiştir. Bu modelle, *FLEURS* ve *Mozilla CV* veri kümelerinde yapılan deneylerde, *Whisper-large-v2* modeline göre hata oranında %10 ile %20 arasında iyileşme sağlanmıştır [129]. Çizelge 4.7'de üzerinde çalışılan veri kümelerine ait *Whisper-large-v2* ve *Whisper-large-v3* model karşılaştırması yer almaktadır.

Sonuçlardan da anlaşılacağı üzere, *Whisper-large-v3* modeli, *Whisper-large-v2* modeliyle karşılaştırıldığında hata oranlarında yüksek oranda iyileşme sağlanmıştır. KaHO ölçütü bağlamında, ODTÜ MK ve THKM veri kümeleri için hata oranı sırasıyla %29,08 ve %17,30 düşerken; *FLEURS*, *Mozilla CV* ve TOKKT veri kümelerinde iyileşme sırasıyla %10,11, %8,77 ve %13,84 oranında olmuştur.

Çizelge 4.7. Whisper-large-v2 ve Whisper-large-v3 modellerinin karşılaştırılması

Sistem-1	**Ölçüt**	***Whisper large-v2***	***Whisper large-v3***	**Değişim (%)**
ODTÜ MK ***(%100, 6.168 kayıt)***	**KHO**	0,061	**0,043**	-29,08
	KaHO	0,018	**0,010**	-44,27
	Süre	00:47	**00:47:58**	0,00
THKM ***(%30, 20.583 kayıt)***	**KHO**	0,103	**0,085**	-17,30
	KaHO	0,037	**0,028**	-24,97
	Süre	02:57	**02:58:53**	0,86
FLEURS ***(%100, 3.127 kayıt)***	**KHO**	0,083	**0,075**	-10,11
	KaHO	0,021	**0,020**	-7,41
	Süre	00:43	**00:43:01**	-0,96
Mozilla CV ***(%30, 15.743 kayıt)***	**KHO**	0,156	**0,142**	-8,77
	KaHO	0,090	**0,084**	-6,29
	Süre	01:50	**01:54:45**	3,60
TOKTT ***(%100, 286 kayıt)***	**KHO**	0,081	**0,069**	-13,84
	KaHO	0,032	**0,027**	-15,35
	Süre	00:10	**00:10:35**	-0,94

4.4. Whisper ve Google USM model performanslarının karşılaştırılması

Whisper ve *Google USM* güncel iki konuşma tanıma sistemidir. *OpenAI* firması *Whisper* OKT sistemini açık kaynak kod dağıtım lisansı ile paylaşmıştır. *Google* firması, *Google USM*'ye modelini açık olarak paylaşacağını deklare etmesine rağmen geçen zaman içinde bu gerçekleşmemiştir.

Buna karşın, modele ücretli uygulama programlama arabirimleri (UPA) üzerinden erişilebilmektedir. UPA arkasında işletilen *Google USM*'nin iyileştirme noktaları ve çalıştırıldığı donanıma ait bilgiler net değildir. Modele ait sayılan bu belirsizliklere rağmen iki sistem arasında, dolaylı yoldan performans karşılaştırması yapılabilmektedir.

İki OKT sistemi arasında, ODTÜ MK, THKM, *FLEURS*, *Mozilla CV* ve TOKTT veri kümeleri kullanılarak KHO ve KaHO üzerinden performans karşılaştırılması yapılmıştır. Her bir veri kümesinden belirli sayıda rastgele konuşma dosyası seçilmiştir. Seçilen dosyalar *Whisper large-v3* modeli ve arkasında *Google USM* çalışan (*Chirp* olarak adlandırılan) UPA'ya girdi olarak verilmiştir. Elde edilen çıktı metinleri, gerçek konuşma metinleri ile karşılaştırılarak KHO ve KaHO değerleri bulunmuştur. Karşılaştırılmada kullanılan veri kümelerine ait örnek sayıları, elde edilen ölçüt değerleri ve iki modele ait değerler arasındaki yüzde cinsinden farklar Çizelge 4.8'de verildiği gibidir.

Çizelge 4.8. Whisper-large-v3 ile Google USM model başarı karşılaştırma tablosu

	Ölçüt	***Whisper large-v3***	***Google USM***	**Fark (%)**
ODTÜ MK ***(2.468 kayıt)***	**KHO**	0,043	0,049	13,95
	KaHO	0,010	0,010	0,00
THKM ***(2.471 kayıt)***	**KHO**	0,084	0,087	3,57
	KaHO	0,028	0,025	-10,71
FLEURS ***(2.706 kayıt)***	**KHO**	0,075	0,093	24,00
	KaHO	0,021	0,036	71,43
Mozilla CV ***(2.743 kayıt)***	**KHO**	0,066	0,063	-4,55
	KaHO	0,015	0,020	33,33
TOKTT ***(286 kayıt)***	**KHO**	0,069	0,098	42,03
	KaHO	0,027	0,044	62,96

Google USM kullanıldığında, KHO ölçütü yönünden sadece *Mozilla CV* veri kümesinde daha başarılı bir sonuç (daha düşük KHO) elde edilmiştir. Bu veri kümesinde, diğer modele göre daha yüksek KaHO değeri dolayısıyla daha düşük başarı elde edildiği halde KHO değeri daha düşük çıkmıştır. Buna karşın, aynı model için THKM veri kümesindeki KaHO iyileşmesi KHO değerine yansımamıştır.

4.5. Otomatik Konuşma Tanıma Sisteminin İnce Ayarla İyileştirilmesi

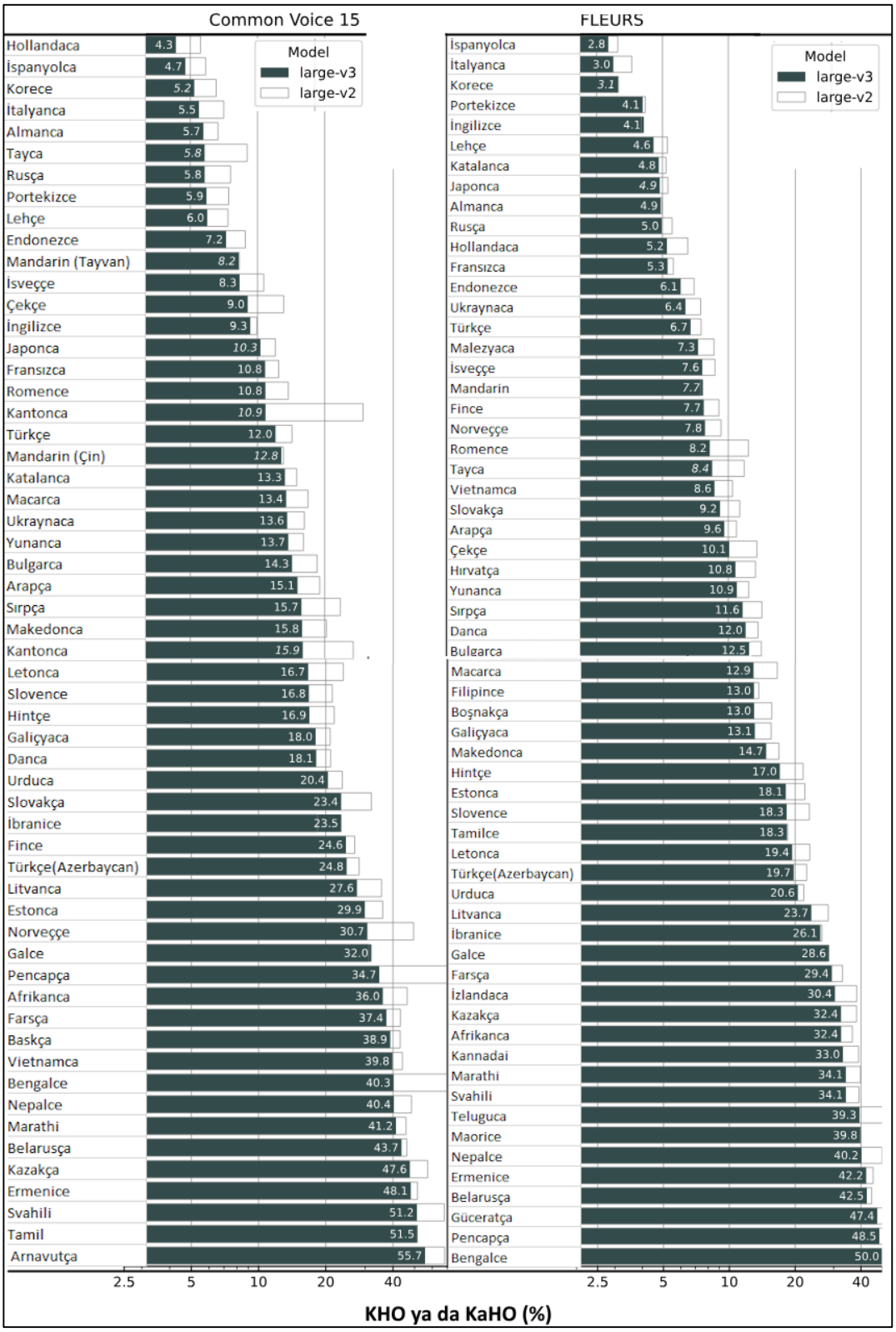
Whisper, yüze yakın dil ile kullanılabilir ve çoklu alt akış görevini (çeviri, konuşma tanıma vb.) destekler. Ayrıca, diğer birçok mimaride olduğu gibi ince ayar özelliğine sahiptir.

Bu mimariye ait modelleri eğitilirken, çok sayıda dildeki etiketsiz ve etiketli veri kullanılmıştır. Model eğitiminde kullanılan, İngilizce ve diğer en büyük eğitim verisine sahip ilk 10 dile ait veri kümesi boyutları süre cinsinden Çizelge 4.9'da verildiği gibidir.

Çizelge 4.9. Whisper dil bazında eğitim veri kümesi boyutları [6]

DİL	VERİ KÜMESİ BOYUTU (Saat)
İngilizce	438218
Çince	23446
Almanca	13344
İspanyolca	11100
Rusça	9761
Fransızca	9752
Portekizce	8573
Korece	7993
Japonca	7054
Türkçe	4333
Lehçe	4278

Ayrıca, *Whisper-large-v2* ve *Whisper-large-v3* modellerinin *Common Voice* ve *FLEURS* veri kümelerinde yer alan diller için KHO ve KaHO değerleri açısından karşılaştırılması Şekil 4.1'de verilmiştir. Çizelge 4.9 ve Şekil 4.1 birlikte değerlendirildiğinde, etiketli ve etiketsiz veri kümeleri açısından zengin dillerde, diğer benzer dillere nazaran daha başarılı sonuçlar elde edildiği görülmektedir. Bu da, eğitim aşamalarında ve üst akış görevinde kullanılan etiketsiz veri ile ince ayar işleminde kullanılan etiketli veri miktarının önemini ortaya koyar.



Şekil 4.1. Whisper-large-v2 ve Whisper-large-v3 model performans karşılaştırması.

İnce ayar işleminde model, yeni etiketli verilerle, ilgili alt akış görevi için tekrar eğitilir. Yapılan ince ayar işlemi, modelin iyileştirilmesine üç şekilde katkı sunar:

1. Çoklu alt akışı görevine sahip modelin, sadece belirli bir görev için (bu çalışma bağlamında OKT) uyumlanması.
2. Sadece belirli bir dile ait verilerle eğitilerek, modelin ilgili dile karşı yanlılığının ve dolayısıyla o dildeki performansının arttırılması.
3. Belirlenen eğitim veri kümesiyle, simgeleştirme (tokenization) işleminde yer alan öğelerin olasılıklarının güçlendirilmesi, çıktıların ilgili öğeler açısından yanlılığının arttırılması. Örneğin, sayılar için rakamlarla çıktı üreten modelin, sayıların metin olarak gösterildiği veri kümeleri ince ayara tabi tutularak, rakam yerine metin çıktı üretme olasılığının arttırılması gibi.
4. Varsa, sözlükte yer almayan yeni simgelerin sözlüğe eklenmesi.

Whisper vb. çözümlere ait üst modellerde, ince ayar işlemi için yüksek işlem gücü ve belleğe ihtiyaç duyulur. İşlem gücü ve bellek için genellikle güçlü GİB'lere sahip grafik kartları kullanılır. Ancak çoğu zaman modelin boyutu büyüdükçe üst sınıf ekran kartları da yetersiz kalmaya başlar.

Bu nokta, bahsedilen zorlukları aşmak adına kullanılan yöntemlerden biri DDU'dur. Bu çalışmada, ince ayar işlemi DDU kullanılarak gerçekleştirilmiştir. DDU yöntemiyle, *Whisper* mimarisinde, eğitilen hedef parametre sayısı yaklaşık olarak 3/100, GİB belleği ihtiyacı da yaklaşık 1/3 oranına düşürülebilmektedir [112].

DDU çözümünde, genellikle, en çok öğrenmenin gerçekleştiği dikkat mekanizması gibi yapılara ait ağırlıkların güncellenmesi hedeflenir [112]. Daha detayda, çoğunlukla, dönüştürücü yapısı içinde yer alan dikkat mekanizması bileşenleri *Q, V* vb. yapılara ait ağırlıkların güncellenmesi tercih edilir.

Şekil 4.2'de *Whipser-large-v2* modeline ait katman yapısı örneklenmiştir. Modelde yer alan, dikkat mekanizması öğeleri ve tam bağlı sinir ağı katmanları vb. doğrusal yapılar DDU kapsamında eğitilebilir durumda olan bileşenlerdir.

```
WhisperForConditionalGeneration(
(model): WhisperModel(
(encoder): WhisperEncoder(
(conv1): Conv1d(128, 1280, kernel_size=(3,), stride=(1,), padding=(1,)
)
(conv2): Conv1d(1280, 1280, kernel_size=(3,), stride=(2,), padding=(1,)
)
(embed_positions): Embedding(1500, 1280)
(layers): ModuleList(
(0-31): 32 x WhisperEncoderLayer(
(self_attn): WhisperSdpaAttention(
(k_proj): Linear(in_features=1280, out_features=1280, bias=False)
(v_proj): Linear(in_features=1280, out_features=1280, bias=True)
(q_proj): Linear(in_features=1280, out_features=1280, bias=True)
(out_proj): Linear(in_features=1280, out_features=1280, bias=True))
(self_attn_layer_norm): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
(activation_fn): GELUActivation()
(fc1): Linear(in_features=1280, out_features=5120, bias=True)
(fc2): Linear(in_features=5120, out_features=1280, bias=True)
(final_layer_norm): LayerNorm(
(1280,), eps=1e-05, elementwise_affine=True))
)
(layer_norm): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
)
(decoder): WhisperDecoder(
(embed_tokens): Embedding(51866, 1280, padding_idx=50256)
(embed_positions): WhisperPositionalEmbedding(448, 1280)
(layers): ModuleList(
(0-31): 32 x WhisperDecoderLayer(
(self_attn): WhisperSdpaAttention(
(k_proj): Linear(in_features=1280, out_features=1280, bias=False)
(v_proj): Linear(in_features=1280, out_features=1280, bias=True)
(q_proj): Linear(in_features=1280, out_features=1280, bias=True)
(out_proj): Linear(in_features=1280, out_features=1280, bias=True)
)
(activation_fn): GELUActivation()
(self_attn_layer_norm): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
(encoder_attn): WhisperSdpaAttention(
(k_proj): Linear(in_features=1280, out_features=1280, bias=False)
(v_proj): Linear(in_features=1280, out_features=1280, bias=True)
(q_proj): Linear(in_features=1280, out_features=1280, bias=True)
(out_proj): Linear(in_features=1280, out_features=1280, bias=True)
)
(encoder_attn_layer_norm): LayerNorm((1280,), eps=1e-05, elementwise_affine=True)
(fc1): Linear(in_features=1280, out_features=5120, bias=True)
(fc2): Linear(in_features=5120, out_features=1280, bias=True)
(final_layer_norm): LayerNorm(
(1280,), eps=1e-05, elementwise_affine=True))
)
(layer_norm): LayerNorm((1280,), eps=1e-05, elementwise_affine=True))
)
(proj_out): Linear(in_features=1280, out_features=51866, bias=False)
```

Şekil 4.2. *Whisper-large-v3* modeli katman yapısı

4.5.1. Whisper-large-v3 modeli üzerinde ince ayar işlemi

Whisper-large-v3 modeliyle yapılan ince ayar işleminde, tüm doğrusal (*linear*) katmanların kullanılması eğitilen parametre sayısının 2 ila 4 kat artırmasına ve donanımsal darboğazlara neden olmuştur. Bundan dolayı, ince ayar işlemi sadece *Q* ve *V* yapıları üzerinden gerçekleştirilmiştir.

Öncelikli olarak hız ve bellek gereksinimini azaltması nedeniyle, 8 bitlik tamsayı (*int8*) formatında ince ayar işlemi yapılmış ancak bu seçim, duyarlılıkta azalma ve modelin halüsinasyon probleminde artışa sebep olmuştur. Bu nedenle, ince ayar işleminde 16 bitlik kayar sayı (*Float16*) formatı tercih edilmiş. Eğitim sırasında öbek boyutu (*batch size*) 32, öğrenme oranı 5.10^{-6} ve gradyan toplama adımı (*gradient accumulation step*) [130] 2 seçilmiştir.

Modelin çok yüksek miktarda veriyle eğitilmiş olması sebebiyle yaklaşık 10 saatlik veri kümesiyle yapılan eğitim sonucunda başarı artışı sağlanmamıştır. Daha yüksek miktarda eğitim verisi ile iyileşme sağlanacağı düşünülse de, hem etiketli veri kümesinin azlığı hem de donanım ve çalışma zamanından kaynaklanan kısıtlar nedeniyle ince ayar işleminde daha alt modellere odaklanılmıştır.

4.5.2. Whisper-medium modeli üzerinde ince ayar işlemi

Nispeten daha küçük olan *Whisper-medium* modeli daha az parametre içermektedir. 24 GB'lık belleğe sahip Sistem-2, ince ayar işleminde tüm doğrusal bileşenler kullanılmasına imkân vermektedir. Bu nedenle, işlem sırasında hedef katman olarak bütün doğrusal katmanlar (*fc1, fc2, v_proj, q_proj, proj_out, out_proj*) kullanılmıştır.

Modelin küçük boyutlu olması; daha büyük veri kümeleriyle, daha hızlı bir eğitim imkânı sağlamaktadır. Bundan dolayı, kullanılan veri kümesi değiştirilmiştir. Eğitim sürecinde, diğer veri kümelerine göre daha fazla kayıt içeren ve 05.01.2024 tarihinde yayımlanan güncel *Common Voice Corpus 16.1* veri kümesi kullanılmıştır. İnce ayar işleminde, önceki yapılandırmada olduğu gibi öbek boyutu 32, öğrenme oranı 5.10^{-6} ve gradyan toplama adımı 2 seçilmiştir. 6500 devir ve toplamda 72 saat süren ince ayar işlemi sonucunda Çizelge 4.10'da verilen ölçüm değerleri elde edilmiştir.

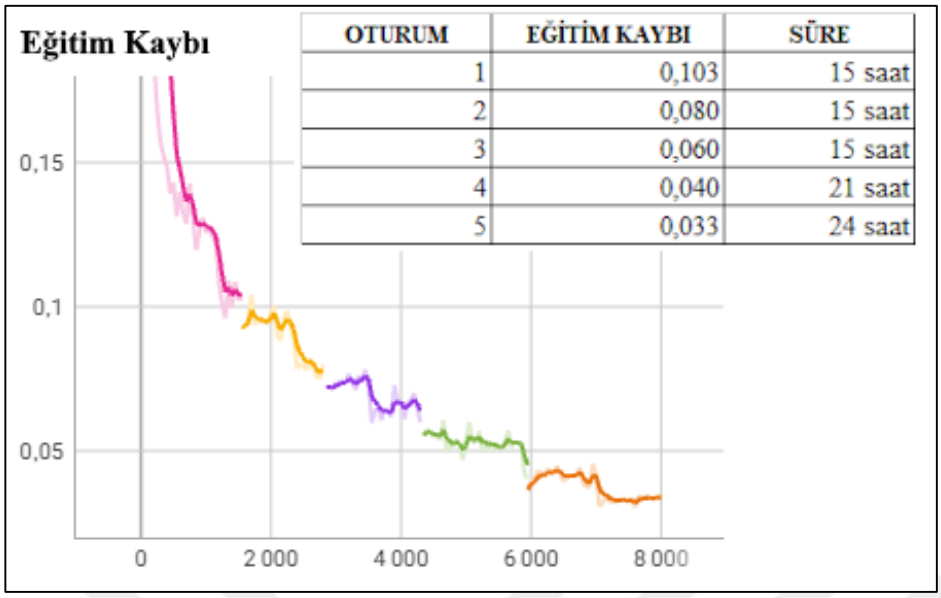
Çizelge 4.10. Whisper-medium modeli için ince ayar sonrası hata oranı karşılaştırma tablosu

Sistem-2	Ölçüt	*Whisper-medium*	*Whisper-medium (İnce ayar)*	Değişim (%)
ODTÜ MK ***(%100, 6.168 kayıt)***	**KHO**	0,109	0,065	-40,37
	KaHO	0,043	0,015	-65,12
THKM ***(%10,8.233 kayıt)***	**KHO**	0,139	0,136	-2,16
	KaHO	0,047	0,036	-23,40
FLEURS ***(%100, 3.127 kayıt)***	**KHO**	0,118	0,117	-0,85
	KaHO	0,037	0,034	-8,11
Mozilla CV ***(%10, 5.248 kayıt)***	**KHO**	0,210	0,172	-18,10
	KaHO	0,117	0,099	-15,38
TOKTT ***(%100, 286 kayıt)***	**KHO**	0,91	0,106	16,48
	KaHO	0,035	0,042	20,00

İşlem sonucunda, KHO değeri yönünden en belirgin düşüş ODTÜ MK veri kümesinde gözlemlenmiştir. Çeşitli dillerde çok sayıda yabancı kelime içeren THKM ve *FLEURS* veri kümelerinde çok büyük bir iyileşme görülmemiştir. Diğer taraftan, *Mozilla CV* veri kümesinde %18,1 bir iyileşme gerçekleşirken, TOKTT veri kümesindeki %16,48'lik bir performans düşüşü görülmüştür. Benzer şekilde diğer dört veri kümesinde KaHO değerinde düşüş gerçekleşirken, TOKTT veri kümesinde %20 artış olmuştur. Özellikle THKM veri kümesindeki KaHO iyileşmesinin KHO değerine aynı oranda yansımadığı gözlemlenmiştir.

4.5.3. Whisper-large-v2 modeli üzerinde ince ayar işlemi

Bir sonraki aşamada, eğitim veri kümesi genişletilerek (*FLEURS* ve *Mozilla* CV veri kümelerinin %80'i) *Whisper-large-v2* modeli üzerinde ince ayar işlemi yapılmıştır. Bir önceki ince ayar işleminde olduğu gibi, öbek boyutu 32, öğrenme oranı 5.10^{-6} ve gradyan toplama adımı 2 seçilmiştir. Eğitim 5 oturumda, yaklaşık 90 saat sürmüş ve 8000 turda tamamlanmıştır. DDU yöntemi kullanıldığı için doğrudan KHO üzerinden bir kayıp hesabı yapılamadığından; eğitim kaybı, simgeleştirme (*tokenization*) başarısı üzerinden hesaplanmıştır. İnce ayar işlemine ait eğitim kaybı eğrisi Şekil 4.3'te verildiği gibi oluşmuştur.



OTURUM	EĞİTİM KAYBI	SÜRE
1	0,103	15 saat
2	0,080	15 saat
3	0,060	15 saat
4	0,040	21 saat
5	0,033	24 saat

Şekil 4.3. İnce ayar işleminde gerçekleşen eğitim kaybı grafiği

İşlem sonrasında yapılan testlerde elde edilen sonuçlar Çizelge 4.11'de verildiği gibidir.

Çizelge 4.11. Whisper-large-v2 modeli için ince ayar sonrası hata oranı karşılaştırma tablosu

Sistem-2	*Ölçüt*	*Whisper large-v2*	*Whisper large-v2 (İnce ayar)*	*Değişim (%)*
ODTÜ MK (%100, 6.168 kayıt)	**KHO**	0,059	0,050	-15,25
	KaHO	0,020	0,010	-50,00
THKM (%10,8.233 kayıt)	**KHO**	0,132	0,109	-17,42
	KaHO	0,051	0,032	-37,25
FLEURS (%100, 3.127 kayıt)	**KHO**	0,083	0,047	-43,37
	KaHO	0,021	0.014	-33,33
Mozilla CV (%10, 5.248 kayıt)	**KHO**	0,10	0,051	-49,00
	KaHO	0,021	0,010	-52,38
TOKTT (%100, 286 kayıt)	**KHO**	0,081	0,079	-2,47
	KaHO	0,032	0,030	-6,25

FLEURS veri kümesinin %80'i ince ayar işleminde kullanıldığı için, kümenin tamamı üzerinde yapılan test işleminde beklenildiği gibi yüksek (%43,37) bir KHO iyileşmesi gerçekleşmiştir. İnce ayar işleminde kullanılmayan *Mozilla CV* veri kümesinin %10'luk kısmıyla yapılan testte %50'ye yakın bir KHO iyileşmesi olmuştur. Benzer şekilde, ODTÜ MK veri kümesinde %50'lik bir KaHO iyileşmesi olduğu halde, bu KHO'ya tam olarak

yansımamış, yine de hata oranı %15,25 azalmıştır. THKM veri kümesinde sırasıyla %17,42 ve %37,25'lik bir KHO ve KaHO iyileşmesi görülmüştür. Öte yandan, içerdiği konuşma verisinin yaklaşık %75'i 30 saniyenin üstünde uzunluğa sahip olan TOKTT veri kümesi için %2,47 KHO, %6,25 KaHO iyileşmesi olmuş, ince ayar işlemi sınırlı oranda katkı sağlamıştır.

4.6. Çevrimiçi, Eş Zamanlıya Yakın (Kısa Gecikmeli) Konuşma Tanıma

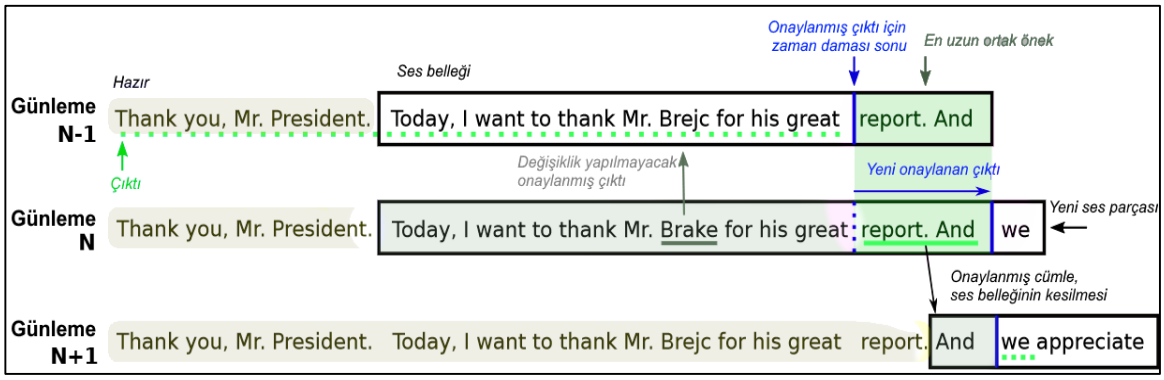
Whisper mimarisi, eş zamanlı OKT yeteneğine sahip değildir. Bu nedenle, eş zamanlı ya da eş zamanlıya yakın konuşma tanıma için farklı uygulamalara ihtiyaç duyulur. Çevrimiçi eş zamanlıya yakın OKT sistemi ile ilgili deneysel çalışmada, *Macháček* ve diğerleri tarafından geliştirilen *Whisper-Streaming* [131] ile *Suryan ve Edel* tarafından geliştirilen *WhisperLive* [132] uygulamaları kullanılmıştır.

4.6.1. Whisper-Streaming uygulaması

Whisper-Streaming, arka planda *faster-whisper* kütüphanesini ve ses etkinliği tespit uygulaması *Silero-VAD*'ı kullanan bir OKT eklentisidir. Uygulama, Şekil 4.4'te gösterilen işleyişe sahiptir [131].

Bu araçla, öbekler halinde, çevrimiçi alınan ham konuşma verisi *Whisper* OKT sistemi tarafından işlenir ve 3 saniyeye yakın bir gecikmeyle metne dönüştürülür. Öbekten alınan konuşma verisinin pencereler halinde işlendiği uygulamada, onaylama mekanizması kullanılarak metin güncellenir.

Çalışma kapsamında, OKT modeli olarak *Whisper-large-v3* kullanılmıştır. İstemci tarafında, *pyaudio, socket* ve *threading* kütüphaneleri kullanılarak, mikrofondan alınan sesi paketler halinde sunucuya gönderen, sonucu ekran ve metin dosyası çıktısı olarak veren küçük bir uygulama geliştirilmiştir.

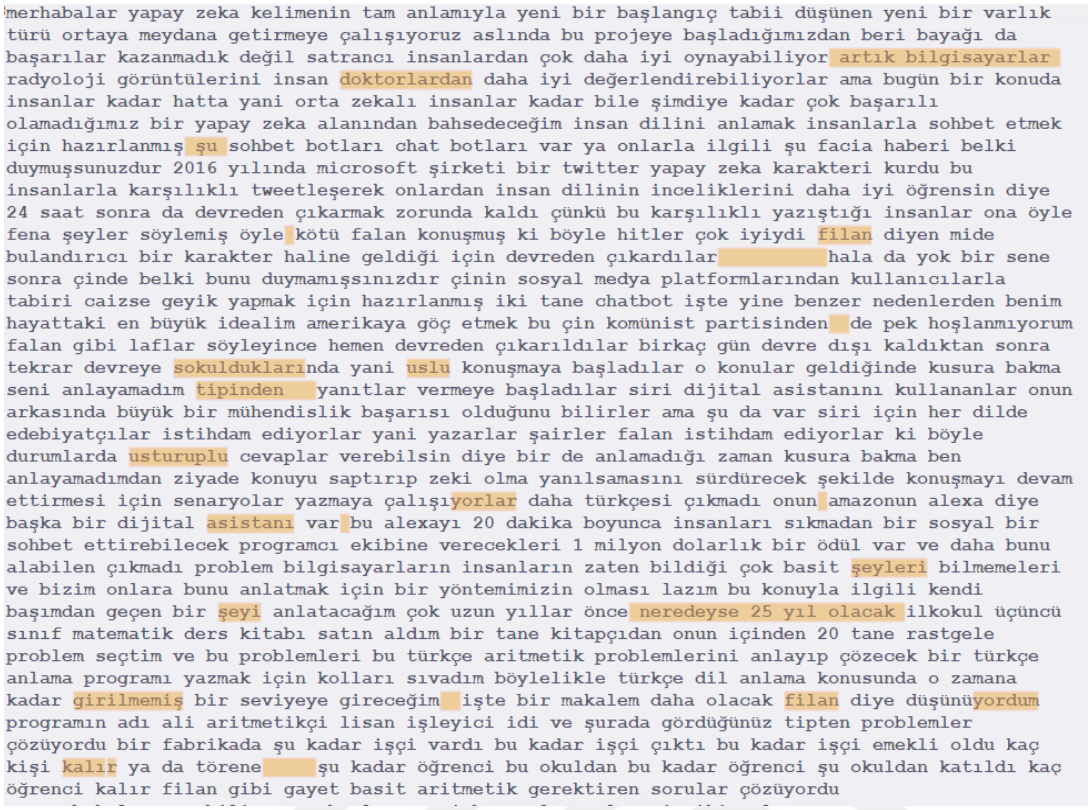


Şekil 4.4. Whisper-Streaming uygulamasının işleyişi [131]

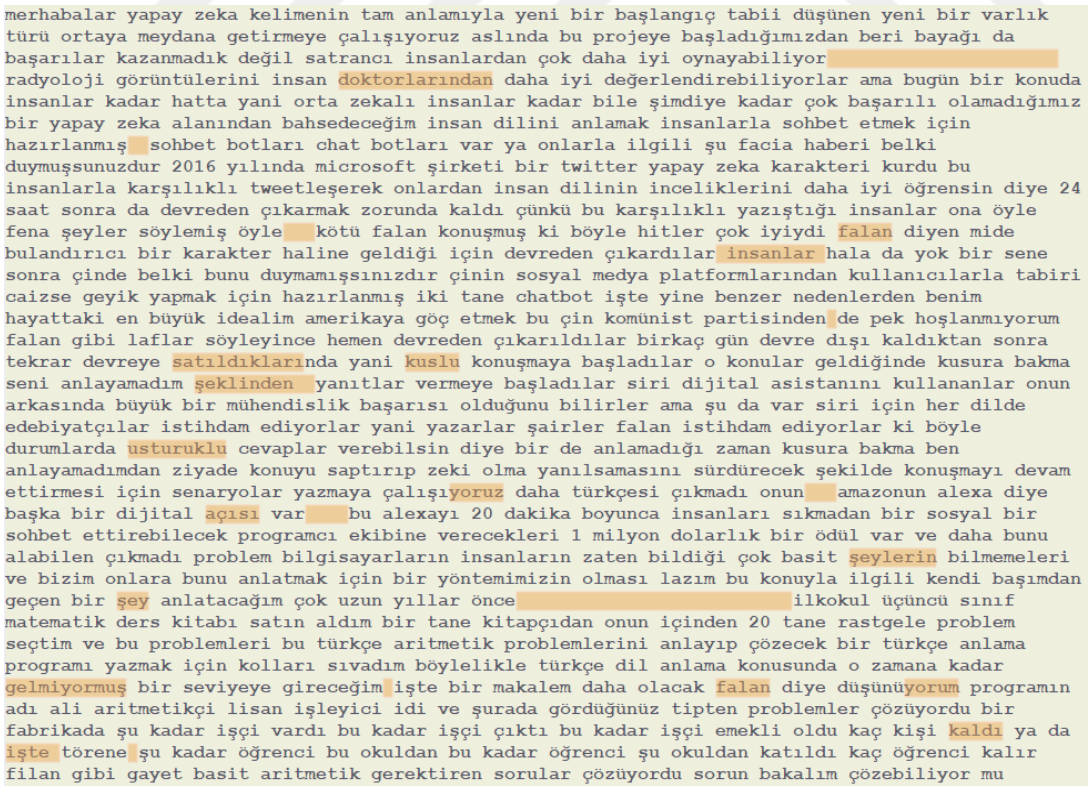
Whisper-Streaming uygulaması ön tanımlı ayarlarında, ses verisi formatı 16 bit tam sayı, örneklem frekansı 16000, paket boyutu 65536 bayttır. Bu durum, ses belleğinin yaklaşık 2 saniyede bir dolmasına ve ek gecikmelere sebep olur. Bu nedenle, 0,5 saniyede bir gönderim gerçekleştirilecek şekilde paket boyutu 16000 bayta düşürülmüştür.

Uygulamadaki ses aktivasyon tespit özelliği işler hale getirilerek konuşmadaki boşlukların kaldırılması sağlanmıştır. Böylece, boşluklardan kaynaklanan halüsinasyon problemi azaltılmıştır.

Deneysel çalışmada, Prof. Dr. Cem Say tarafından *TEDxIstanbul* etkinliğinde sunulan, yaklaşık 17 dakikalık "Geleceğin Zekâsı: Önyargısız Robotlar" başlıklı video kullanılarak performans ölçümü yapılmıştır [133]. Videodaki konuşmaya ait metin 1898 kelime ve 14881 sembolden oluşmaktadır. Videoya ait ses, harici bir mikron yardımıyla alınarak, geliştirilen istemci uygulamasıyla Sistem-1 üzerinden çalışan *Whisper-Streaming* uygulamasına gönderilmiş ve gelen çıktı işlenerek metin dosyasına kaydedilmiştir. Konuşmaya ait ilk 467 kelimelik metin ve uygulama tarafından üretilen çıktı Şekil 4.5 ve Şekil 4.6'da verildiği gibidir. Şekil 4.5, asıl metni; Şekil 4.6, uygulama tarafından üretilen çıktıyı göstermektedir. Farklılık olan bölümler metinler üzerinde işaretlenmiştir.



Şekil 4.5. Konuşmaya ait 467 kelimeden oluşan ilk bölüm



Şekil 4.6. Uygulama tarafından üretilen metin

Konuşma, az sayıda yabancı kelime içermektedir ve güncel Türkçe kullanılarak yapılmıştır. Başlangıç kısmı hariç kalan bölüm net ve anlaşılır durumdadır. Konuşmada, gürültü seviyesi çok azdır ve ses kalitesi yüksektir. Bu koşullar altında, asıl ve üretilen metin arasında, hata oranları üzerinden yapılan karşılaştırmada 0,088 KHO ve 0,041 KaHO değerleri elde edilmiştir. Bağlantı hızı ve sunucu yoğunluğundan dolayı, 3 saniyenin üstünde, 5 ile 10 saniye arasında değişen sürelerde gecikmeler olmuştur.

4.6.2. WhisperLive uygulaması

WhisperLive, Whisper-Streaming gibi arka planda *faster-whisper* kütüphanesi ve ses etkinliği tespit uygulaması *Silero-VAD* kullanarak eş zamanlıya yakın metin çıktısı üreten bir OKT aracıdır. *Whisper-Streaming*'den en büyük farkı, istemciyle sunucu uygulamalarını birlikte içermesi ve *Mozilla Firefox* ile *Google Chrome* web gezginleri için istemci eklentilerine sahip olmasıdır. Bunun dışında, *WhisperLive*'da metin düzeltme ve güncelleme işleminin istemci tarafında yapılır. Ayrıca, veri gönderme-alma işleminde *websocket* kütüphanesi kullanılarak, iletişimden kaynaklanan gecikmelerin azalmasını sağlamıştır.

WhispeLive ile yapılan deneylerde, metin güncelleme işleminin istemci tarafından yapılması ve iletişim noktasındaki iyileştirmeler sayesinde gecikmenin 1-2 saniye arasında gerçekleştiği görülmüştür. Öte yandan, ses etkinlik tespit özelliği açık olmasına rağmen, Çizelge 4.12'de verilen çıktıdan da anlaşılacağı gibi, halisünasyon problemi yaşanmıştır. Bu nedenle, aynı konuşmayla [133] yapılan deneyde başarım düşmüş; 0,161 KHO ve 0,113 KaHO değerleri elde edilmiştir.

Çizelge 4.12. Örnek WhisperLive çıktısı

1 00:00:00,000 --> 00:00:00,320 Merhabalar. 3 00:00:06,620 --> 00:00:09,519 türü **ortaya meydana getirmeye çalışıyoruz.** 4 00:00:09,519 --> 00:00:11,140 yeni bir varlık türü 5 00:00:11,140 --> 00:00:13,679 **ortaya meydana getirmeye çalışıyoruz.** 11 00:00:26,839 --> 00:00:36,380 Ama bugün bir konuda insanlar kadar hatta yani orta zekalı insanlar kadar bile şimdiye kadar çok başarılı olamadığımız **bir yapay zeka alanından bahsedeceğim.** 12 00:00:36,380 --> 00:00:38,420 **bir yapay zeka alanından bahsedeceğim.**

5. SONUÇ VE ÖNERİLER

Bu çalışma kapsamında, giriş bölümünde, öncelikle OKT kavramı açıklanmış ve OKT çalışmalarının tarihçesi sunulmuştur. Daha sonra, OKT sistemleri işleyişi ve karşılaşılan belli başlı zorluklar ele alınmış, temel OKT mimarisi açıklanarak sistem türleri tanıtılmıştır. İlgili çalışmalar bölümünde, konuyla ilgili diğer çalışmalara değinilmiştir. Materyal ve metot bölümünde konuşma sinyali işlenme ve sinyalden öznitelik çıkarma yöntemleri anlatılmış; kullanılan veri kümeleri tanıtılmıştır. Ardından, OKT sistemlerinde kullanılan makine öğrenmesi yöntemleri açıklanmıştır. OKT araçları bölümünde, yaygın olarak kullanılan OKT araçları değinilmiştir. Daha sonra, *Whisper* modeli kullanılarak, adı geçen veri kümeleri üzerinde yapılan deneysel çalışmaların sonuçları paylaşılmış; bir başka OKT mimarisi *Google USM* ile *Whisper* mimarisi arasında performans karşılaştırması yapılmış; ince ayar işleminin *Whisper* modelleri üzerindeki etkisi değerlendirilmiştir. Son olarak, çevrim içi, eş zamanlıya yakın konuşma tanıma işlemi için geliştirilen uygulamalar üzerinde yapılan deneylerin sonuçları paylaşılmıştır.

Yaklaşık 150 saati veri kümelerinin incelenmesi ve düzenlemesi; 800 saati deney, ince ayar, eş zamanlı konuşma tanıma sistemi geliştirilmesi olmak üzere toplamda 1000 saate yakın bir çalışma yürütülmüştür.

Veri kümeleri üzerinde yapılan ilk incelemelerde, aşağıdaki genel hata ve sorunlar tespit edilmiştir:

1. Konuşma ile metne aktarımındaki farklılıklar:
   a. Metnin tam ve doğru şekilde okunmaması.
   b. Konuşmanın metne farklı aktarılması.
   c. Konuşmada olmayan ifadelerin metne eklenmesi.
2. Yazım hataları:
   a. Türkçe karakterlerdeki noktalamaların eksik ya da hatalı olması.
   b. Ek veya bağlaç olarak kullanılan ifadelerin, olması gerektiğinden farklı olarak bitişik ya da ayrı yazılması.
   c. İmlâ kuralları gereği bitişik ya da ayrı yazılması gereken kelimelerin metne farklı şekilde aktarılması.

3. Sesle ilgili sorunlar:
   a. Metindeki ifade tamamlanmadan kaydın sonlandırılması.
   b. Gürültü vb. ses kalitesi sorunları.

Deneysel çalışmalar sonucunda, kullanılan mimarinin çıktılarında bağlaçların olması gerektiğinden farklı olarak, ayrı ya da birleşik yazılmasından kaynaklanan kelime ve harf hatalarının oluştuğu görülmüştür. Örneğin "ile" gibi eklerin yazılı metinden ayrı ya da bitişik olarak metne dönüştürülmesi anlambilim (semantik) açıdan herhangi bir hata içermese de, sözdizimi (sentaks) olarak hatalı kabul edilmekte; KHO ve KaHO değerlerini artırmaktadır. Diğer yandan, konuşmadan doğrudan anlaşılması dinleyici için bile zor olan bağlaç ya da edat görevindeki "-de" vb. ekler, model tarafından hatalı şekilde bitişik ya da ayrı yazılabilmektedir. Bu durum, hem sözdizimi hem de anlambilim açısından yanlış kabul edilmekte, hata oranlarını artırmaktadır. Benzer şekilde, bitişik yazılan “Gaziosmanpaşa” vb. özel isimlerin, zaman zaman metne ayrık olarak aktarıldığı görülmüştür. Ayrıca, ayrı olduğu halde, ulama nedeniyle bitişikmiş gibi söylenen kelimeler metne de bu şekilde aktarılabilmektedir. Bu gibi durumlar, KHO değerini artırmaktadır.

Dikkatlice yazılan bir metinde, yaklaşık %1-2 arasında yazım hatası olduğu tahmin edilir [10]. Öte yandan, hızlı yazılan ya da kontrol edilmemiş metinler için bu oranın yaklaşık %10-15 olduğu düşünülür [10]. Bu çalışma kapsamında, ODTÜ MK veri kümesinde, birçok noktada seslendirilen konuşma tekrar dinlenerek, düzeltmeler yapılmıştır. Metinlerin ilk ve son halleri karşılaştırıldığında, %1,7 oranında karakter ve %5,8 oranında kelime farklılığı olduğu belirlenmiştir. Düzeltme işleminden kaynaklanabilecek eksiklikler ve sorunlar göz önüne alındığında; konuşmacının metne uygun seslendirme yapmaması, se kelimelerin ya da harflerin farklı seslendirilmesi vb. hatalardan dolayı yaklaşık %1-2'lik harf hatası ve %5'lik bir kelime hatası olduğu görülmüştür. Bu da, başta verilen tahminlerle örtüşmektedir.

Tüm bu hususlar dikkate alındığında, mevcut *Whisper* mimarisisin, dönüştürülen metnin anlaşılırlığı yönünden, elde edilen hata değerlerinin üstünde başarıya sahip olduğu anlaşılmaktadır.

İnce ayar işlemiyle, görev ve dil bağlamında modellerinin performansının artırılabilmektedir. Ancak; THKM, ODTÜ MK vb. veri kümelerinin, farklı model ve mimariler arasında karşılaştırma imkânı sunmasına rağmen ince ayar işlemi için yetersiz kaldığı görülmüştür. THKM veri kümesi, içerdiği örnek sayısı bakımından zengin olsa da,

içeriğindeki yabancı kelimelerden kaynaklanan karışıklıklar sebebiyle, mevcut haliyle ince ayar işlemine uygun değildir. Öte yandan, ODTÜ MK veri kümesinin, içerdiği hatalar giderilse bile, boyutunun küçük olması sebebiyle, tek başına, ince ayar işlemi için kullanıma uygun olmadığı değerlendirilmektedir.

2018 yılında tanıtılan ve diğer Makine Öğrenmesi yöntemleriyle karşılaştırıldığında oldukça yeni sayılabilecek Temsil Öğrenme ve Kendi Kendine Denetimli Öğrenme, pek çok Doğal Dil İşleme uygulamasında olduğu gibi OKT alanında da başarılı bir şekilde uygulanmaktadır. *Wav2vec*, OKT bağlamında Kendi Kendine Denetimli Öğrenmenin kullanıldığı, yaygınlaşmayı başaran öncül mimarilerden biridir. Bu mimarinin başarısından sonra, *Wav2vec 2.0* geliştirilmiş, *Whisper* ve sonrasında *Google USM* mimarileri ortaya çıkmıştır. Bütün bu mimarilerde, kullanılan yöntemlerde farklılık olsa da, temel işleyişleri hemen hemen aynıdır. İşleyişte, ilk olarak, yüksek miktarda etiketsiz konuşma verisiyle temsil öğrenme üst akış görevi yürütülür. Ardından, mümkün olan en büyük etiketli konuşma ve metin verisiyle sistem eğitilir.

Bahsi geçen mimariler de dâhil, günümüzde yaygın olarak kullanılan pek çok OKT uygulaması uçtan uca, çok görevli olarak tasarlanmaktadır. Ayrı akustik model ya da dil modeli içermeyen bu tür uygulamalar, OKT işlevinin yanı sıra otomatik çeviri, konuşmacı etiketleme vb. görevleri de gerçekleştirebilecek çok işlevli bir yapıya sahiptirler. Güncel OKT sistemlerinin pek çoğunda, Saklı Markov Modeli gibi geleneksel istatistik modeller yerini kodlayıcı ve çözümleyicilerden oluşan dönüştürücülere ve derin öğrenme yapılarına bırakmıştır.

Dönüştürücü içeren mimariler, yüksek miktarda eğitim verisine ve işlem gücüne ihtiyaç duyarlar. Günümüzde, sürekli güncellenen veri kümeleri ve gelişen GPU teknolojisi sayesinde, bu tür zorluklarla başa çıkmak daha kolay hale gelmiştir. Sürekli artan işlem gücü, geliştiricileri, çok büyük boyuttaki veri kümeleriyle eğitilen, daha yüksek parametre sayısına sahip modeller geliştirme konusunda teşvik etmektedir. Başarıyı artırmak adına, mimaride köklü değişiklikler yerine, dikkat mekanizması gibi noktalarda kısmi iyileştirme ve eğitim veri kümesinin büyütülmesi tercih edildiği anlaşılmaktadır. Dönüştürücü yapıların ardından, daha yeni ve farklı bir yöntem geliştirilmediği sürece, gelişimin bu şekilde devam edeceği tahmin edilmektedir. Başka bir deyişle, sürekli büyüyen eğitim verisi ve girdi

öznitelikleriyle birlikte parametre sayısı katlanan, bu nedenle güçlü GİB'lere sahip sistemlere ihtiyaç duyan modellere yönelim olduğu görülmektedir.

Microsoft Azure konuşma tanıma uygulamalarına ait dokümanda [134], KHO değeri 0,05-0,10 ise uygulamanın kalitesinin iyi, kullanıma uygun olduğu; 0,10-0,20 ise kullanılabilir olduğu ancak ek eğitime gerektiği; 0,30 ve üzerindeyse sinyal kalitesinin ya da modelin yetersiz olduğu kabul edilir. *Whisper'*a ait üst modellerle, birçok Türkçe veri kümesi üzerinde yapılan deneylerde 0,05 ile 0,15 arasında bir KHO değeri elde edilmiştir. Özellikle, son yayımlanan *Whisper-large-v3* modeliyle yapılan deneylerde, 0,04 ile 0,10 KHO değerine ulaşılmıştır. Bu sonuçlar, *Whisper'*ın benzer örnekleri arasında öne çıkan, başarılı bir mimari olduğunu göstermektedir.

İnce ayar işleminde, kullanılan modelin performansını artırmak adına büyük veri kümelerine ihtiyaç duyulur. Kullanılacak veri kümelerinin boyut olarak yeterli olmasının yanında, ince ayar işleminde başarımı düşürmeyecek şekilde az hata içermelidir. Bu açıdan Türkçe, diğer pek çok dile nazaran daha geniş veri kümelerine sahip olsa da, bu kümeler, özellikle ince ayar gibi işlemlerde yetersiz kalmaktadır. Yapılan çalışma sonucunda, büyük boyutlu ve az sayıda hata içeren yeni Türkçe veri kümelerinin geliştirilmesine ihtiyaç duyulduğu görülmüştür. İlerleyen dönemlerde, sayılan özelliklere sahip veri kümelerinin geliştirilmesi konusunda çalışmalar yapılması gerektiği düşünülmektedir.

Eş zamanlıya yakın konuşma tanıma uygulamalarında hata ve hız arasında ödünleşme olduğu görülmüştür. *Whisper Stream* uygulaması daha az hataya sahip çıktı üretirken, yapısı ve işleyişi nedeniyle nispeten yavaş çalışmaktadır. Öte yandan *WhisperLive* uygulaması hız konusunda daha iyi bir performansa sahipken, ürettiği çıktılardaki hata miktarı daha fazladır. Sonraki çalışmalarda, yapılacak iyileştirmelerle, gecikme süresi düşük, daha az hata üreten eş zamanlı konuşma tanıma uygulamaları geliştirilmesinin yerinde olacağı düşünülmektedir.

KAYNAKLAR

1. Holden, C. (2004). The Origin of Speech. *Science*, 303(5662), 1316-1319.

2. Gülensoy, T. (2007). *Türkiye Türkçesindeki Türkçe sözcüklerin köken bilgisi sözlüğü: etimolojik sözlük denemesi.* 1*: A - N* (1. baskı). Ankara: Türk Dil Kurumu Yayınları, 540.

3. Lv, Z., Poiesi, F., Dong, Q., Lloret, J., and Song, H. (2022). Deep Learning for Intelligent Human–Computer Interaction. *Applied Sciences*, *12*(22), 11457.

4. Mihelič, F., and Žibert, Z. (2008). *Speech recognition: technologies and applications*. Rijeka: IntechOpen, 455-477.

5. Oyucu, S. (2020). *Türkçe Konuşma Tanıma Sistemleri için Derin Öğrenme Tabanlı Modellerin Geliştirilmesi.* Doktora Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

6. Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. (2023). *Robust speech recognition via large-scale weak supervision*. International Conference on Machine Learning, 28492-28518.

7. Pisanski, K., Fraccaro, P. J., Tigue, C. C., O'Connor, J. J. M., Röder, S., Andrews, P. W., Fink, B., DeBruine, L. M., Jones, B. C., and Feinberg, D. R. (2014). Vocal indicators of body size in men and women: a meta-analysis. *Animal Behaviour*, 95, 89-99.

8. Asefisaray, B. (2018). *Uçtan-uca konuşma tanıma modeli: Türkçe'deki deneyler* Doktora Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

9. Dalva, D. (2012). *Türkçe Dili için Otomatik Konuşma Tanıma Sistemi*, Yüksek Lisans Tezi, Işık Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

10. Jurafsky, D., and Martin, J. H. (2020). *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition*. New Jersey: Pearson Prentice Hall, 91-548.

11. Rabiner, L. R., and Juang, B. H. (1993). *Fundamentals of speech recognition*. London: Pearson Prentice Hall, 2-8.

12. Dudley, H., Riesz, R. R., and Watkins, S. S. (1939). A synthetic speaker. *Journal of the Franklin Institute*, 227(6), 739-764.

13. Dudley, H. (1940). The vocoder. *Nature*, 145(3665), 157-157.

14. Juang, B. H., and Rabiner, L. R. (2005). Automatic speech recognition–a brief history of the technology development. *Georgia Institute of Technology*, 1, 67.

15. Turing, A. M. (2012). Computing machinery and intelligence (1950). *The Essential Turing: the Ideas That Gave Birth to the Computer Age*, 433-464.

16. Russell, S. J., and Norvig, P. (2016). *Artificial intelligence: a modern approach*. Boston: Pearson Prentice Hall, 17.

17. Davis, K. H., Biddulph, R., and Balashek, S. (1952). Automatic Recognition of Spoken Digits. *The Journal of the Acoustical Society of America*, 24(6), 637-642.

18. Olson, H. F., and Belar, H. (1956). Phonetic Typewriter. *The Journal of the Acoustical Society of America*, 28(6), 1072-1081.

19. Fry, D. B. (1959). Theoretical aspects of mechanical speech recognition. *Journal of the British Institution of Radio Engineers*, 19(4), 211-218.

20. Forgie, J. W., and Forgie, C. D. (1959). Results obtained from a vowel recognition computer program. *The Journal of the Acoustical Society of America*, 31(11), 1480-1489.

21. Sakai, T., and Doshita, Shuji. (1962). *The phonetic typewriter*. Proc. IFIP Congress 62, 445-450.

22. Suzuki, J., and Nakata, K. (1961). Recognition of Japanese vowels-preliminary to the recognition of speech. *Japan Radio Research Lab*, 37(8), 193-212.

23. Nagata, K., Kato, Y., and Chiba, S. (1964). Spoken digit recognizer for the Japanese language. *Journal of the Audio Engineering Society*, 12(4), 336-342.

24. Anniss, M. (2013). *How Does Voice Recognition Work?*. New York: Gareth Stevens Publishing, 48.

25. Martin, T. B., Nelson, A. L., and Zadell, H. J. (1964). *Speech recognition by feature-abstraction techniques*. Massachusetts: Defense Documentation Center, 4-10.

26. Vintsyuk, T. K. (1968). Speech discrimination by dynamic programming. *Cybernetics*, 4(1), 52-57.

27. Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2), 260-269.

28. Itakura, F. (1975). Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on acoustics, speech, and signal processing*, 23(1), 67-72.

29. Klatt, D. H. (1977). Review of the ARPA speech understanding project. *The Journal of the Acoustical Society of America*, 62(6), 1345-1366.

30. Lowerre, B. T. (1980). The HARPY speech understanding system. *Trends in speech recognition*, 340-360.

31. Lippmann, R. P. (1989). Review of neural networks for speech recognition. *Neural computation*, 1(1), 1-38.

32. Juang , B. H., and Rabiner, L. R.F. (1991). Hidden Markov Models for Speech Recognition. *Technometrics*, 33(3), 251-272.

33. Wilpon, J. G., Rabiner, L. R., Lee, C.-H., and Goldman, E. (1990). Automatic recognition of keywords in unconstrained speech using hidden Markov models. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(11), 1870-1878.

34. Levinson, S. E., Rabiner, L. R., and Sondhi, M. M. (1983). An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. *Bell System Technical Journal*, 62(4), 1035-1074.

35. Wilpon, J. G. (1995). Voice-processing technologies--their application in telecommunications. *Proceedings of the National Academy of Sciences*, 92(22), 9991-9998.

36. Deshmukh, A. M. (2020). Comparison of hidden markov model and recurrent neural network in automatic speech recognition. *European Journal of Engineering and Technology Research*, 5(8), 958-965.

37. Qin, Y., Shi, Q., Liu, Y. Y., Aronowitz, H., Chu, S. M., Kuo, H.-K., and Zweig, G. (2006). *Advances in mandarin broadcast speech transcription at IBM under the DARPA GALE program.* Chinese Spoken Language Processing: 5th International Symposium, 410-421.

38. Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). *Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks*. Proceedings of the 23rd International Conference on Machine Learning - ICML '06, 369-376.

39. Graves, A., Mohamed, A., and Hinton, G. (2013). *Speech Recognition with Deep Recurrent Neural Networks*. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 6645-6649.

40. Chan, W., Jaitly, N., Le, Q. V., and Vinyals, O. (2015). Listen, Attend and Spell. *arXiv preprint ArXiv:1508.01211*.

41. Zhang, Y., Chan, W., and Jaitly, N. (2017). *Very deep convolutional networks for end-to-end speech recognition*. 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 4845-4849.

42. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2023). Attention Is All You Need. *arXiv preprint ArXiv: 1706.03762*.

43. Schneider, S., Baevski, A., Collobert, R., and Auli, M. (2019). wav2vec: Unsupervised Pre-training for Speech Recognition. *arXiv preprint ArXiv:1904.0586*.

44. Oord, A. van den, Li, Y., and Vinyals, O. (2019). Representation Learning with Contrastive Predictive Coding. *arXiv preprint ArXiv:1807.03748*.

45. Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. (2020). wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. İçinde H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Ed.), *Advances in Neural Information Processing Systems*, 33, 12449-12460.

46. Gulati, A., Qin, J., Chiu, C.-C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y., and Pang, R. (2020). Conformer: Convolution-augmented Transformer for Speech Recognition. *arXiv preprint ArXiv:2005.08100*.

47. Arora, S. J., and Singh, R. P. (2012). Automatic speech recognition: a review. *International Journal of Computer Applications*, 60(9), 34-44.

48. Dhanjal, A. S., and Singh, W. (2023). A comprehensive survey on automatic speech recognition using neural networks. *Multimedia Tools and Applications*, 83, 23367–23412

49. Kaur, A., Sachdeva, R., and Singh, A. (2022). Latest Trends in Deep Learning for Automatic Speech Recognition System. *Artificial Intelligence and Speech Technology* , 1546, 62-72.

50. Malik, M., Malik, M. K., Mehmood, K., and Makhdoom, I. (2021). Automatic speech recognition: a survey. *Multimedia Tools and Applications*, *80*(6), 9411-9457.

51. Zhang, Y., Han, W., Qin, J., Wang, Y., Bapna, A., Chen, Z., Chen, N., Li, B., Axelrod, V., Wang, G., Meng, Z., Hu, K., Rosenberg, A., Prabhavalkar, R., Park, D. S., Haghani, P., Riesa, J., Perng, G., Soltau, H., Wu, Y. (2023). Google USM: Scaling Automatic Speech Recognition Beyond 100 Languages. *arXiv preprint ArXiv:2303.01037*.

52. Kaderli, A. (1994). *Evrimsel fourier dönüşümü kullanarak konuşma tanıma*. Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

53. Erzin, E. (1995). *Konuşma tanıma için gürültüye dayanıklı yeni yöntemler*. Yüksek Lisans Tezi, Bilkent Üniversitesi Mühendislik ve Fen Bilimleri Enstitüsü, Ankara.

54. Uslu, E. (2007). *Gizli Markov modeli ile geniş sözlüklü sürekli konuşma tanıma*. Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

55. Özkan, Ö. (1997). *Birleşik sayılar için konuşma tanıma sistemi uygulaması*. Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

56. Bayer, A. O. (2005). *Türkçe geniş dağarcıklı sürekli konuşma tanıma için dil modelleme üzerine bir çalışma*. Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

57. Çömez, M. A. (2003). *HTK ile Türkçe için geniş dağarcıklı akan konuşma tanıma*. Doktora Tezi, Orta Doğu Teknik Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

58. Karaca, N. (1999). *Gürültülü ortamlarda Türkçe ayrık sözcük konuşma tanıma sistemi gerçekleştirimi*. Doktora Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

59. Yılmaz, C. (1999). *Türkçe için geniş sözcük dağarcıklı konuşma tanıma sistemi*. Yüksek Lisans Tezi, Bilkent Üniversitesi Mühendislik ve Fen Bilimleri Enstitüsü, Ankara.

60. Edizkan, R. (1999). *Gizli Markov model ile bilgisayarda konuşma tanıma: Özellik uzayında ve altuzayda sınıflandırıcı tasarımı*. Doktora Tezi, Osmangazi Üniversitesi Fen Bilimleri Enstitüsü, Eskişehir.

61. Şahin, S. (2003). *Türkçe akan konuşma tanıma için dil modellemesi*. Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

62. Dede, G. (2008). *Yapay sinir ağları ile konuşma tanıma*. Yüksek Lisans Tezi, Ankara Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

63. Susman, D. (2012). *Kısıtlı ses külliyatı ile Türkçe geniş dağarcıklı sürekli konuşma tanıma*. Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

64. Urgun, D. (2012). *Yapay zekâ tabanlı konuşma tanıma sistemi*. Yüksek Lisans Tezi, Atılım Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

65. Taşar, D. E. (2023). *Organizasyonel gelişim için otomatik konuşma tanıma sistemi önerisi*. Yüksek Lisans Tezi, Dokuz Eylül Üniversitesi Sosyal Bilimler Enstitüsü, İzmir.

66. Arslan, R. S. (2020). *Uçtan uca türkçe konuşma tanıma için çıktı düzeltme metodu önerisi ve tekrarlayan sinir ağı tasarımı*. Doktora Tezi, Gazi Üniversistesi Fen Bilimleri Enstitüsü, Ankara.

67. Elberri, M. A. (2020). *Tekrarlayan sinir ağlarını kullanarak konuşma tanıma*. Yüksek Lisans Tezi, Kastamonu Üniversitesi Fen Bilimleri Enstitüsü, Kastamonu.

68. Kutucu, H. (2020). *Derin öğrenme algoritmaları kullanarak bir konuşma tanıma uygulaması*. Yüksek Lisans Tezi, Sakarya Uygulamalı Bilimler Üniversitesi Lisansüstü Eğitim Enstitüsü, Sakarya.

69. Ahmet, M. J. (2020). *Çağrı merkezleri için derinöğrenme tabanlı interaktif konuşma tanıma*. Yüksek Lisans Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü, Konya.

70. Haznedaroglu, A., Arslan, L. M., Buyuk, O., and Erden, M. (2010). *Turkish LVCSR system for call center conversations*. 2010 IEEE 18th Signal Processing and Communications Applications Conference, 372-375.

71. Çetinkaya, G., Arisoy, E., and Saraclar, M. (2020). *Improving the Usage of Subword-Based Units for Turkish Speech Recognition*. 2020 28th Signal Processing and Communications Applications Conference (SIU), 1-4.

72. Tombaloğlu, B., and Erdem, H. (2017). *A SVM based speech to text converter for Turkish language*. 2017 25th Signal Processing and Communications Applications Conference (SIU), 1-4.

73. Oyucu, S., ve Polat, H. (2022). Türkçe Otomatik Konuşma Tanıma Sistemi için Dil Modeli Optimizasyon Yöntemi. *Journal of Polytechnic*, 26(3), 1167-117.

74. Yalman, H. İ., ve Tüfekci, Z. (2022). Yeni Bir Türkçe Sesli Kitap Veri Seti Üzerinde Convolutional RNN+CTC, LSTM+CTC ve GRU+CTC Modellerinin Karşılaştırılması. *European Journal of Science and Technology*, 34, 321-327.

75. Hsu, W.-N., Bolte, B., Tsai, Y.-H. H., Lakhotia, K., Salakhutdinov, R., and Mohamed, A. (2021). HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29, 3451-3460.

76. Mercan, O. B., Cepni, S., Tasar, D. E., and Ozan, S. (2023). Performance Comparison of Pre-trained Models for Speech-to-Text in Turkish: Whisper-Small and Wav2Vec2-XLS-R-300M. *arXiv preprint ArXiv:2307.04765*.

77. Rabiner, L. R., and Schafer, R. W. (1978). *Digital processing of speech signals*. London: Pearson Prentice Hall, 121.

78. Tiwari, V. (2010). MFCC and its applications in speaker recognition. *International journal on emerging technologies*, 1(1), 19-22.

79. Abdul, Z. Kh., and Al-Talabani, A. K. (2022). Mel Frequency Cepstral Coefficient and its Applications: A Review. *IEEE Access*, 10, 122136-122158.

80. Ajibola, A. S., Rashid, N. K. B. A. M., Sediono, W., and Hashim, N. N. W. N. (2016). A novel approach to stuttered speech correction. *Jurnal Ilmu Komputer dan Informasi*, 9(2), 80-87.

81. Salor, O., Ciloglu, T., and Demirekler, M. (2006). *METU Turkish Microphone Speech Corpus*. 2006 IEEE 14th Signal Processing and Communications Applications, 1-4.

82. Arisoy, E., Can, D., Parlak, S., Sak, H., and Saraclar, M. (2009). Turkish Broadcast News Transcription and Retrieval. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(5), 874-883.

83. Ardila, R., Branson, M., Davis, K., Henretty, M., Kohler, M., Meyer, J., Morais, R., Saunders, L., Tyers, F. M., and Weber, G. (2020). Common Voice: A Massively-Multilingual Speech Corpus. *arXiv preprint ArXiv:1912.06670*.

84. Conneau, A., Ma, M., Khanuja, S., Zhang, Y., Axelrod, V., Dalmia, S., Riesa, J., Rivera, C., and Bapna, A. (2022). FLEURS: Few-shot Learning Evaluation of Universal Representations of Speech. *arXiv preprint ArXiv:2205.12446*.

85. Oyucu, S. (2022). Development Of Test Corpus Wıth Large Vocabulary For Turkısh Speech Recognıtıon System And A New Test Procedure. *Adıyaman Üniversitesi Mühendislik Bilimleri Dergisi*, 9(16), 156-164.

86. Liwicki, M., Graves, A., Fernàndez, S., Bunke, H., and Schmidhuber, J. (2007). *A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks*. Proceedings of the 9th International Conference on Document Analysis and Recognition, ICDAR 2007, 372-376.

87. Fakhan, E. (2021). *Automatic Speech Recognition System Adaptation For Spoken Lecture Processing*. Doktora Tezi, Boğaziçi Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

88. Hannun, A. (2017). Sequence modeling with ctc. *Distill*, 2(11), e8.

89. Zou, J., Han, Y., and So, S.-S. (2008). Overview of Artificial Neural Networks. *Artificial Neural Networks*, 458, 14-22.

90. Nart Sooksil, and Vacharapoom Benjaoran. (2021). Non-linear modelling of construction workers' behaviorsfor accident prediction. *Songklanakarin Journal of Science and Technology (SJST)*, 43, 596602.

91. Turan, A. K., ve Polat, H. (2023). Yarı denetimli makine öğrenmesi yöntemini kullanarak müzik türlerinin tespiti. *Gazi Üniversitesi Fen Bilimleri Dergisi Part C: Tasarım ve Teknoloji*, 12(1), 92-107.

92. Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278-2324.

93. Feng, W., Guan, N., Li, Y., Zhang, X., and Luo, Z. (2017). *Audio visual speech recognition with multimodal recurrent neural networks*. 2017 International Joint Conference on Neural Networks (IJCNN), 681-688.

94. İnternet: Amidi, A., and Amidi, S. CS 230-Tekrarlayan Yapay Sinir Ağları el kitabı. Url: https://stanford.edu/~shervine/l/tr/teaching/cs-230/cheatsheet-recurrent-neural-networks, Son Erişim Tarihi: 28.09.2023.

95. Jiang, C., Chen, Y., Chen, S., Bo, Y., Li, W., Tian, W., and Guo, J. (2019). A Mixed Deep Recurrent Neural Network for MEMS Gyroscope Noise Suppressing. *Electronics*, 8(2), 181.

96. Da Silva, D. G., and Meneses, A. A. D. M. (2023). Comparing Long Short-Term Memory (LSTM) and bidirectional LSTM deep neural networks for power consumption prediction. *Energy Reports*, 10, 3315-3334.

97. Jiang, C., Chen, S., Chen, Y., Bo, Y., Han, L., Guo, J., Feng, Z., and Zhou, H. (2018). Performance Analysis of a Deep Simple Recurrent Unit Recurrent Neural Network (SRU-RNN) in MEMS Gyroscope De-Noising. *Sensors*, 18(12), 4471.

98. Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. *arXiv preprint ArXiv:1409.3215*.

99. Bahdanau, D., Cho, K., and Bengio, Y. (2016). *Neural Machine Translation by Jointly Learning to Align and Translate*. *arXiv preprint ArXiv:1409.0473*.

100. van Engelen, J. E., and Hoos, H. H. (2020). A survey on semi-supervised learning. *Machine Learning*, 109(2), 373-440.

101. Amini, M.-R., Feofanov, V., Pauletto, L., Devijver, E., and Maximov, Y. (2023). Self-Training: A Survey. *arXiv preprint ArXiv:2202.12040*.

102. Xiaojin, Z. (2008). Semi-supervised learning literature survey. *Computer Sciences TR*, 1530, 60-60.

103. Chapelle, O., Schölkopf, B., and Zien, A. (Ed.). (2010). *Semi-supervised learning*. Massachusetts: MIT Press, 5.

104. Zhai, X., Oliver, A., Kolesnikov, A., and Beyer, L. (2019). *S4L: Self-Supervised Semi-Supervised Learning*. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 1476-1485.

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

106. Yuan, Y., and Lin, L. (2021). Self-Supervised Pretraining of Transformers for Satellite Image Time Series Classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14, 474-487.

107. Doersch, C., Gupta, A., and Efros, A. A. (2015). *Unsupervised Visual Representation Learning by Context Prediction*. 2015 IEEE International Conference on Computer Vision (ICCV), 1422-1430.

108. MILLET, J., Caucheteux, C., orhan, pierre, Boubenec, Y., Gramfort, A., Dunbar, E., Pallier, C., and King, J.-R. (2022). Toward a realistic model of speech processing in the brain with self-supervised learning. *Advances in Neural Information Processing Systems*, 35, 33428-33443.

109. Zhang, M., Chen, X., and Li, W. (2021). A Hybrid Hidden Markov Model for Pipeline Leakage Detection. *Applied Sciences*, 11(7), 3138.

110. Levinson, S. E., Rabiner, L. R., and Sondhi, M. M. (1983). An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. *Bell System Technical Journal*, 62(4), 1035-1074.

111. Reynolds, D. (2009). Gaussian Mixture Models. S. Z. Li and A. Jain, *Encyclopedia of Biometrics*, 659-663. New York: Springer, 659.

112. Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2021). *LoRA:* Low-Rank Adaptation of Large Language Models. *arXiv preprint ArXiv: 2106.09685*.

113. İnternet: HTK Speech Recognition Toolkit. Url: https://htk.eng.cam.ac.uk/, Son Erişim Tarihi: 10.02.2024.

114. Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., and others. (2011). *The Kaldi speech recognition toolkit*. IEEE 2011 workshop on automatic speech recognition and understanding, CONF, 158-164.

115. İnternet: Deep Neural Networks in Kaldi. Url: https://kaldi-asr.org/doc/dnn.html, Son Erişim Tarihi: 10.02.2024.

116. Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A., and Ng, A. Y. (2014). Deep Speech: Scaling up end-to-end speech recognition. *arXiv preprint ArXiv:1412.5567*.

117. İnternet: mozilla/DeepSpeech. Url: https://github.com/mozilla/DeepSpeech, Son Erişim Tarihi: 10.02.2024.

118. Watanabe, S., Hori, T., Karita, S., Hayashi, T., Nishitoba, J., Unno, Y., Soplin, N. E. Y., Heymann, J., Wiesner, M., Chen, N., Renduchintala, A., and Ochiai, T. (2018). ESPnet: End-to-End Speech Processing Toolkit. *arXiv preprint ArXiv:1804.00015*.

119. İnternet: ESPnet. Url: https://github.com/espnet/espnet, Son Erişim Tarihi: 10.02.2024.

120. Hendrycks, D., and Gimpel, K. (2023). Gaussian Error Linear Units (GELUs). *arXiv preprint ArXiv:1606.08415*.

121. Child, R., Gray, S., Radford, A., and Sutskever, I. (2019). Generating Long Sequences with Sparse Transformers. *arXiv preprint ArXiv:1904.10509*.

122. Chen, Z., Zhang, Y., Rosenberg, A., Ramabhadran, B., Wang, G., and Moreno, P. (2021). Injecting Text in Self-Supervised Speech Pretraining. *arXiv preprint ArXiv:2108.12226*.

123. Park, D. S., Zhang, Y., Jia, Y., Han, W., Chiu, C.-C., Li, B., Wu, Y., and Le, Q. V. (2020). Improved Noisy Student Training for Automatic Speech Recognition. *Interspeech 2020*, 2817-2821.

124. Graves, A. (2012). Sequence Transduction with Recurrent Neural Networks. *arXiv preprint ArXiv:1211.3711*.

125. Chan, W., Park, D., Lee, C., Zhang, Y., Le, Q., and Norouzi, M. (2021). SpeechStew: Simply Mix All Available Speech Recognition Data to Train One Large Neural Network. *arXiv preprint ArXiv:2104.02133*.

126. Kendall, T., and Farrington, C. (2021). The corpus of regional african american language. *Version*, 6, 1.

127. Wang, C., Wu, A., and Pino, J. (2020). CoVoST 2 and Massively Multilingual Speech-to-Text Translation. *arXiv preprint ArXiv:2007.10310*.

128. İnternet: Team, S. Silero VAD: pre-trained enterprise-grade Voice Activity Detector (VAD), Number Detector and Language Classifier. Url: https://github.com/snakers4/silero-vad, Son Erişim Tarihi: 02.01.2024.

129. İnternet: Robust Speech Recognition via Large-Scale Weak Supervision. Url: https://github.com/openai/whisper, Son Erişim Tarihi: 25.12.2023.

130. You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S., Bhojanapalli, S., Song, X., Demmel, J., Keutzer, K., and Hsieh, C.-J. (2020). Large Batch Optimization for Deep Learning: Training BERT in 76 minutes. *arXiv preprint ArXiv:1904.00962*.

131. Macháček, D., Dabre, R., and Bojar, O. (2023). Turning Whisper into Real-Time Transcription System. *arXiv preprint ArXiv: 2307.14743*.

132. İnternet: collabora/WhisperLive. Url: https://github.com/collabora/WhisperLive, Son Erişim Tarihi: 25.12.2023.

133. İnternet: Geleceğin Zekâsı: Önyargısız Robotlar. Url: https://www.youtube.com/watch?v=dCQtt3cA_VA, Son Erişim Tarihi: 11.02.2024.

134. İnternet: Test accuracy of a Custom Speech model - Speech service - Azure AI services. Url: https://learn.microsoft.com/en-us/azure/ai-services/speech-service/how-to-custom-speech-evaluate-data, Son Erişim Tarihi: 25.12.2023.

Gazili olmak ayrıcalıktır