# Yapay Zeka ile Sesten Yazıya Dönüştürme Final Raporu

Samet Duran

14.06.2024

# 1 GİRİŞ

Bu projede insanların konuşmalarının metne dönüştürülerek bilgisayar ortamına aktarılması amaçlandı bu yüzden yapay zeka ile sesten yazıya çevirme konusu seçildi. Bu konuda ki örnekler araştırıldı. Python dili kullanılmasına karar verildi. Openai şirketinin açık kaynak kodlu Whisper adlı modeli incelendi ve araştırıldı. Yapay zeka alanında ki dikkate alma konusu üzerinde araştırma yapıldı, Whisper modelinin gerçek zamanlı nasıl çalıştırılıcağı araştırıldı ve örnek kod bulundu, çalıştırıldı, incelendi. RealtimeStt kuruldu ve çalıştırıldı. DeepSpeech [1] modeli için türkçe veri seti ile eğitilmeye çalışıtırılması çeşitli donanım ve yazılım uyumsuzlukları nedeni ile başarısız olundu ve bırakıldı. Whisper Real Time [2] ve RealtimeStt [3] çıktılarının txt dosyasına kaydedilmesi sağlandı. Daha sonra Whisper Real Time [2] kullanarak Tkinter ile basit bir uygulama yapıldı. Daha sonra Tkinter kullanılarak yapılan uygulama yerine PyQt5 [4] kullanılarak RealtimeSTT [3] ve Whisper Real Time [2] içeren yeni bir uygulama yapıldı. Uygulamaya tasarımsal eklemeler yapıldı. Pyqt5 [4] ile yapılan uygulamada Whisper Real Time ekranında cümle cümle çıkması yerine ChatGPT [5] yardımı ile tüm çıktı gözükecek şekilde değiştirildi. Uygulama için tasarım değişiklikleri yapıldı. Whisper Real Time [2] ve RealtimeSTT [3] daha stabil çalışması için gerekli konfigürasyonlar yapıldı. Uygulama konuşarak ve video sesleri dinletilerek test edildi. Yapılan sesten yazıya dönüştürme uygulamasında bulunan gerçek zamanlı olan RealtimeSTT [3], Whisper Real Time [2]ve uygulamada bulunmayan gerçek zamanlı olmayan Whisper tiny, base, small modelleri örnek sesler ile test edilerek doğruluğu hesaplandı. Yapılan sesten yazıya çevirme uygulamasına ChatGPT [5] yardımıyla gerçek zamanlı olmayan Whisper [6] modeli kullanılan yeni bir ekran eklendi.

# 2 LİTERATÜR ARAŞTIRMASI

OpenAI'nin Whisper ile Mozilla'nın Deepspeech modelleri Konuşmadan Yazıya Dönüştürme konusunda geliştirilmiş modellerdir.

## 2.1 Genel Bilgi:

Konuşmadan yazıya (STT) dönüştürme, sesli konuşmayı metne dönüştüren bir teknolojidir.

## 2.2 Whisper:

- Açık kaynaklı bir ASR modelidir.
- Şuanlık gerçek zamanlı şekilde çalışmıyor, ses dosyası yüklenerek çalıştırılabiliyor.
- Çok dilli bir konuşma tanıma sistemi olarak öne çıkar.
- 680.000 saatten fazla çok dilli ve webden toplanan veriyle eğitilmiştir.
- Bu büyük veri havuzu, benzersiz vurguları, arka plan gürültüsünü ve teknik jargonu daha iyi tanımasına olanak sağlar
- Whisper, hızlı ve doğru sonuçlar elde etmek için tasarlanmıştır.

## 2.3 Deepspeech:

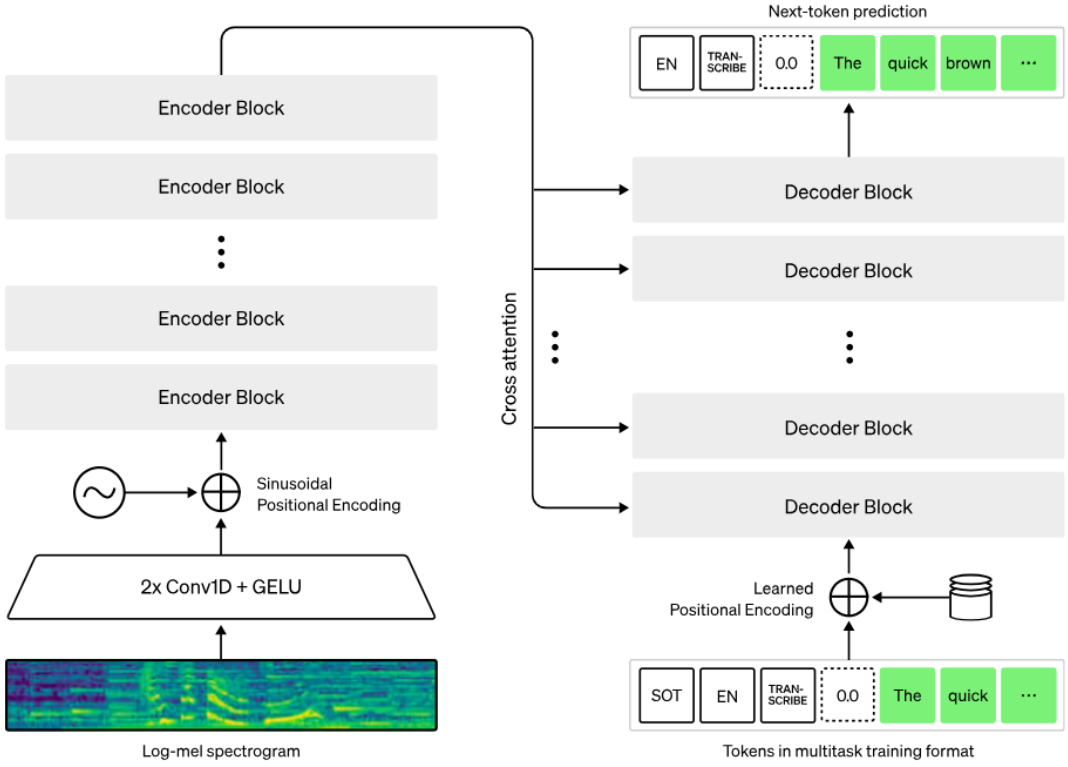
DeepSpeech DeepSpeech, Baidu'nun Deep Speech araştırma makalesine [**?**] dayanan makine öğrenimi teknikleriyle eğitilmiş bir model kullanan açık kaynaklı bir Konuşmadan Metne motorudur. DeepSpeech Projesi, uygulamayı kolaylaştırmak için Google'ın TensorFlow 'unu kullanır. TenserFlow Google tarafından geliştirilen uçtan uca açık kaynaklı bir makine öğrenmesi kütüphanesidir.

- Mozilla tarafından geliştirilen bir ASR modelidir.
- Gerçek Zamanlı çalışıyor fakat sadece ingilizce destekliyor.
- Açık kaynaklı ve topluluk tarafından desteklenir.

- Wav2Vec 2.0 gibi unsupervised training temelinde çalışır.
- Raw waveform representation, context part ve linear layer olmak üzere üç bölümden oluşur.
- Wav2Vec 2.0, konuşma tanıma alanında önemli bir başarıya sahiptir

# 3 YÖNTEM

OpenAI Whisper modeli, sesli konuşmayı metne dönüştürmek için tasarlanmış bir yapay zeka modelidir. Bu model, ses sinyallerini analiz ederek ve her bir sesin hangi kelimelere karşılık geldiğini tahmin ederek çalışır. Whisper modeli, Şekil 1'te görüldüğü gibi bir kodlayıcı ve bir kod çözücü olmak üzere iki ana bileşenden oluşur.




Şekil 1: Whisper Çalışma Mantığı [7]

## 3.1 Kodlayıcı:

Kodlayıcı, ses sinyallerini alır ve bunları bir dizi sayıya dönüştürür. Bu sayılar, sinyallerin frekans ve genlik gibi özelliklerini temsil eder. Kodlayıcı, bir dizi sinir ağı katmanından oluşur ve her katman, sinyallerdeki daha karmaşık örüntüleri öğrenir.

## 3.2 Kod Çözücü:

Kod çözücü, kodlayıcı tarafından üretilen sayıları alır ve bunları metne dönüştürür. Kod çözücü, bir dizi sinir ağı katmanından oluşur ve her katman, sayı dizisinin hangi kelimelere karşılık geldiğini öğrenir.

### 3.3 Bloklar:

- Log-mel Spektrogram: Bu blok, ses sinyallerinin frekans ve genlik bilgilerini içeren bir görüntü oluşturur [8].
- 2x Conv1D + GELU(Gaussian Error Linear Unit): Bu blok, sinyallerdeki daha karmaşık örüntüleri öğrenir.
- Sinüsoidal Pozisyon Kodlama: Bu blok, ses sinyallerinin zamanla nasıl değiştiğini kodlar.
- Dikkate Alma: Bu blok, kodlayıcı tarafından üretilen sayıları ve kod çözücü tarafından üretilen kelimeleri ilişkilendirir.
  Dikkat Ağlarının Temeli: Dikkat ağları, bir girdinin belirli bölümlerine odaklanmak ve bu bölümlere dayalı bir çıktı üretmek için tasarlanmıştır. Bunu, "dikkat ağırlıkları" adı verilen bir dizi değer kullanarak gerçekleştirirler. Dikkat ağırlıkları, her bir girdi öğesinin önemini gösterir.

  Çapraz Dikkat: Çapraz dikkat, birden fazla girdi dizisini işlemek için kullanılan bir dikkat mekanizmasıdır. Bu mekanizma, her bir girdi dizisindeki öğelerin diğer girdi dizileriyle nasıl ilişkili olduğunu öğrenir.

  Çapraz Dikkatin Çalışma Şekli: Çapraz dikkat, her bir girdi dizisi için bir dizi dikkat ağırlığı hesaplayarak çalışır. Bu ağırlıklar, her bir girdi öğesinin diğer girdi dizilerindeki öğelerle ne kadar ilişkili olduğunu gösterir. Daha sonra, bu ağırlıklar, her bir girdi dizisinden gelen bilgileri birleştirmek için kullanılır.

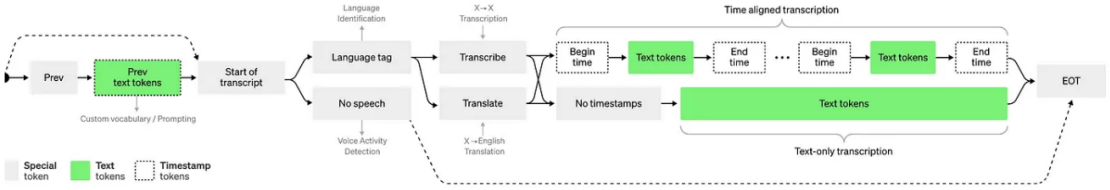
  Çapraz Dikkatin Uygulamaları: Çapraz dikkat, makine çevirisi, metin özetleme ve soru cevaplama gibi birçok NLP görevde kullanılır.
- Öğrenilmiş Pozisyon Kodlama: Bu blok, kodlayıcı tarafından üretilen sayı dizisinin zamanla nasıl değiştiğini kodlar.
- Sonraki Belirteç Tahmini: Bu blok, kodlayıcı tarafından üretilen sayılar ve kod çözücü tarafından üretilen kelimeler temelinde bir sonraki kelimenin ne olacağını tahmin eder.

### 3.4 Modelin Çalışma Aşamaları

- Ses sinyalleri kodlayıcıya girilir.
- Kodlayıcı, sinyalleri bir dizi sayıya dönüştürür.
- Kod çözücü, sayıları metne dönüştürür.

## 3.5 Whisper İş Akış Diyagramı

Aşağıda Şekil 2'de görüldüğü üzere Whisper modelinin iş akış diyagramı görülmektedir.




Şekil 2: Whisper İş Akış Diyagramı [7]

- Ses Girişi yapılır. Ses dalgaları dijital sinyallere dönüştürülür.
- Ses dalgalarında ses olup olmadığı belirlenir. Ses yoksa, işlem durur ve sonraki ses girişine kadar bekler.
- Dil Tanımlaması Yapılır.
- Aynı dilde çıktı oluşturulur veya isteğe bağlı çeviri yapılarak çıktı oluşturulur.
- Duruma göre zaman hizalı transkripsiyon veya sadece metin transkripsiyonu yapılır.

## 3.6 Dikkate Alma

Dikkate Alma [9]: Bu blok, kodlayıcı tarafından üretilen sayıları ve kod çözücü tarafından üretilen kelimeleri ilişkilendirir.
Dikkat Ağlarının Temeli: Dikkat ağları, bir girdinin belirli bölümlerine odaklanmak ve bu bölümlere dayalı bir çıktı üretmek için tasarlanmıştır. Bunu, "dikkat ağırlıkları" adı verilen bir dizi değer kullanarak gerçekleştirirler. Dikkat ağırlıkları, her bir girdi öğesinin önemini gösterir.

Çapraz Dikkat: Çapraz dikkat, birden fazla girdi dizisini işlemek için kullanılan bir dikkat mekanizmasıdır. Bu mekanizma, her bir girdi dizisindeki öğelerin diğer girdi dizileriyle nasıl ilişkili olduğunu öğrenir.

Çapraz Dikkatin Çalışma Şekli: Çapraz dikkat, her bir girdi dizisi için bir dizi dikkat ağırlığı hesaplayarak çalışır. Bu ağırlıklar, her bir girdi öğesinin diğer girdi dizilerindeki öğelerle ne kadar ilişkili olduğunu gösterir. Daha sonra, bu ağırlıklar, her bir girdi dizisinden gelen bilgileri birleştirmek için kullanılır.

Çapraz Dikkatin Uygulamaları: Çapraz dikkat, makine çevirisi, metin özetleme ve soru cevaplama gibi birçok NLP görevde kullanılır.

## 3.7 Whisper Modelini Çalıştırma

OpenAI'nin Whisper Konuşmadan Yazıya Dönüştürme konusunda geliştirilmiş modeldir.

### 3.7.1 Genel Bilgi:

Konuşmadan yazıya (STT) dönüştürme, sesli konuşmayı metne dönüştüren bir teknolojidir.

### 3.7.2 Whisper:

- Açık kaynaklı bir ASR modelidir.
- Şuanlık gerçek zamanlı şekilde çalışmıyor, ses dosyası yüklenerek çalıştırılabiliyor.
- Çok dilli bir konuşma tanıma sistemi olarak öne çıkar [6].
- 680.000 saatten fazla çok dilli ve webden toplanan veriyle eğitilmiştir [7].
- Bu büyük veri havuzu, benzersiz vurguları, arka plan gürültüsünü ve teknik jargonu daha iyi tanımasına olanak sağlar
- Whisper, hızlı ve doğru sonuçlar elde etmek için tasarlanmıştır.

```
import whisper

model = whisper.load_model("base")
result = model.transcribe("audio.mp3")
print(result["text"])
```

Şekil 3: Çalıştırma Kodu

Yukarıda Şekil 3'de görüldüğü üzere whisper modelinin çalıştırmak için gerekli kod parçası örneği bulunmaktadır.

Aşağıda Şekil 4'de gösrüldüğü gibi whisper modelinin büyüklük ve hız karlışatırılması verilmiştir. Büyüklük arttıkça doğruluk artarken işlem hızı da yavaşlıyor.

| Size | Parameters | English-only model | Multilingual model | Required VRAM | Relative speed |
|---|---|---|---|---|---|
| tiny | 39 M | `tiny.en` | `tiny` | ~1 GB | ~32x |
| base | 74 M | `base.en` | `base` | ~1 GB | ~16x |
| small | 244 M | `small.en` | `small` | ~2 GB | ~6x |
| medium | 769 M | `medium.en` | `medium` | ~5 GB | ~2x |
| large | 1550 M | N/A | `large` | ~10 GB | 1x |

Şekil 4: Büyüklük ve hız karşılaştırması [6]

## 3.8 Whisper Modelini Gerçek Zamanlı Olarak Çalıştırma

Aşağıda Şekil 5'de Whisper modelini gerçek zamanlı olarak çalıştırmak için kullanılan kodun bir kısmı verilmiştir.

```
import argparse
import os
import numpy as np
import speech_recognition as sr
import whisper
import torch

from datetime import datetime, timedelta
from queue import Queue
from time import sleep
from sys import platform


def main():
    parser = argparse.ArgumentParser()
    parser.add_argument("--model", default="medium", help="Model to use",
                        choices=["tiny", "base", "small", "medium", "large"])
    parser.add_argument("--non_english", action='store_true',
                        help="Don't use the english model.")
    parser.add_argument("--energy_threshold", default=1000,
                        help="Energy level for mic to detect.", type=int)
    parser.add_argument("--record_timeout", default=2,
                        help="How real time the recording is in seconds.", type=float)
    parser.add_argument("--phrase_timeout", default=3,
                        help="How much empty space between recordings before we "
                             "consider it a new line in the transcription.", type=float)
    if 'linux' in platform:
        parser.add_argument("--default_microphone", default='pulse',
                            help="Default microphone name for SpeechRecognition. "
                                 "Run this with 'list' to view available Microphones.", type=str)
    args = parser.parse_args()

    # The last time a recording was retrieved from the queue.
    phrase_time = None
    # Thread safe Queue for passing data from the threaded recording callback.
    data_queue = Queue()
    # We use SpeechRecognizer to record our audio because it has a nice feature where it can detect when speech ends.
    recorder = sr.Recognizer()
```

Şekil 5: Whisper Gerçek Zamanlı Örnek Kod Parçası [2]

## 3.9 RealtimeSTT

RealtimeSTT, gerçek zamanlı uygulamalar için kullanımı kolay, düşük gecikmeli konuşmadan metne kütüphanesidir [3]. Kullanılan bileşenler:

### 3.9.1 Ses Etkinliği Algılama

İlk ses etkinliği tespiti için WebRTCVAD [10]. Daha doğru doğrulama için SileroVAD [11].

### 3.9.2 Konuşmadan Metne

Anında (GPU hızlandırmalı) transkripsiyon için Faster-Whisper [12].

### 3.9.3 Uyanık Kelime Algılama

Uyandırma kelimesi algılama için Porcupine [13].

RealtimeStt kuruldu,çalıştırıldı ve kodları incelendi. Şekil 6 de ornek kod parçası bulunmaktadır.

```
recorder_config = {
    'spinner': False,
    'model': 'small',
    'language': 'tr',
    'silero_sensitivity': 0.4,
    'webrtc_sensitivity': 2,
    'post_speech_silence_duration': 0.4,
    'min_length_of_recording': 0,
    'min_gap_between_recordings': 0,
    'enable_realtime_transcription': True,
    'realtime_processing_pause': 0.2,
    'realtime_model_type': 'tiny',
    'on_realtime_transcription_update': text_detected,
    #'on_realtime_transcription_stabilized': text_detected,
}
```

Şekil 6: RealtimeSTT örnek kod parçası [14]

### 3.10 Gerçek Zamanlı Sesten Yazıya Dönüştürmede Çıktıları Txt Dosyasına Kaydetme

Whisper Real Time ve RealtimeSTT kodları içine ChatGPT'den yardım alarak çıktıların text dosyasına kaydedilmesi için gerekli kodlar eklendi [5]. Aşağıda Şekil 7'de Whisper Real Time içine eklenen kod bulunuyor, bu kod anlık olarak çıktıları txt dosyasına ekliyor.

```
output_file_path = "deneme1.txt"
with open("deneme1.txt", "a", encoding="utf-8") as file:
    file.write(text + "\n")
```

Şekil 7: Whisperlive çıktılarını txt dosyasına kaydeden kod [5]

Aşağıda Şekil 8 ve Şekil 9'da bulunan kodlar RealtimeSTT içine eklenerek işlem sonlandırılıdıktan sonra çıktıların txt dosyasına kaydedilmesi sağlandı.

```
def write_to_file(text):
    with open(output_file, "a") as file:
        file.write(text + "\n")  # Dosyaya metin ekle
```

Şekil 8: RealtimeSTT çıktılarını txt dosyasına kaydeden birinci kod [5]

```
try:
    while True:
        recorder.text(process_text)
except KeyboardInterrupt:
    write_to_file(displayed_text)
```

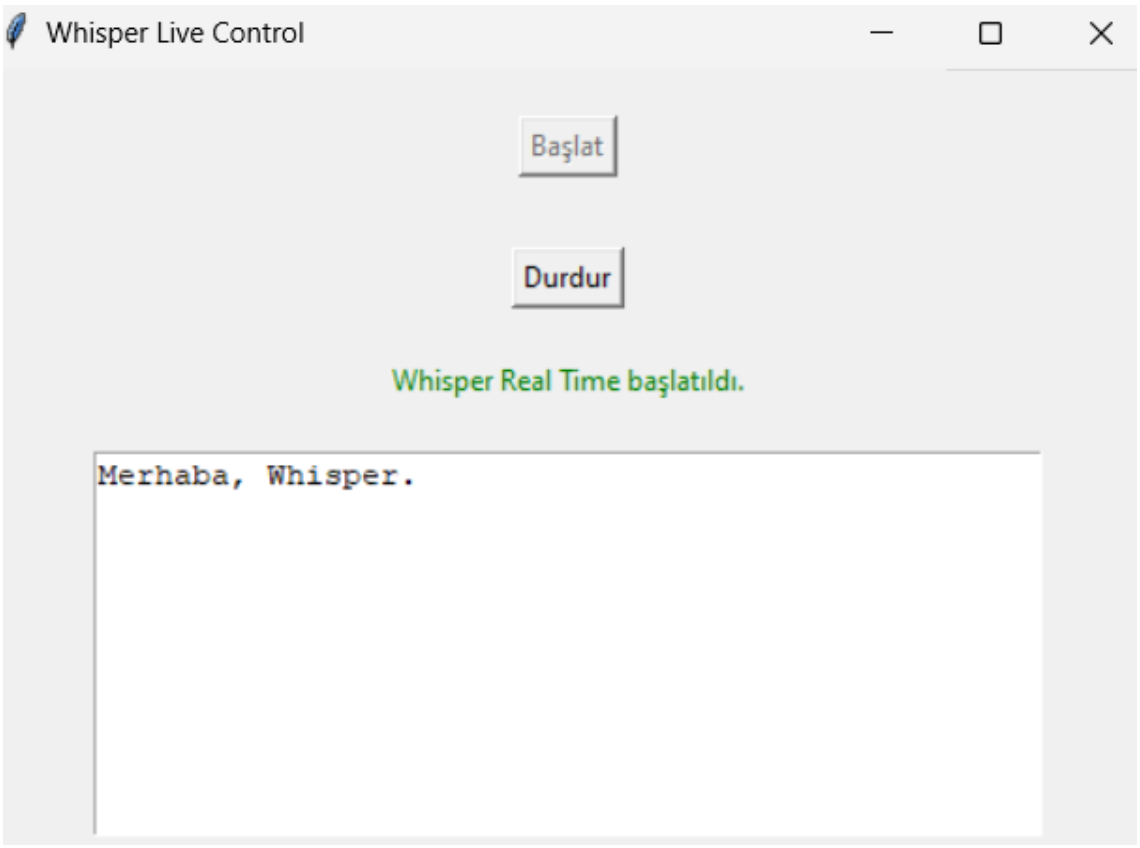
Şekil 9: RealtimeSTT çıktılarını txt dosyasına kaydeden ikinci kod [5]

### 3.11 Tkinter

Tkinter, Python programlama dili ile birlikte gelen grafiksel kullanıcı arayüzü (GUI) aracıdır. Python'la birlikte gelmesi ve basit bir yapıya sahip olması, Tkinter'in yaygın kullanımına neden olmuştur. Eleman (görsel nesne) eksikleri çeşitli paketlerle (ek kodlarla) kapatılmaya çalışılmaktadır [**?**].

### 3.12 Whisper Real Time İçin Basit Bir Uygulama

Whisper Real Time [2] kullanılarak ve ChatGPT [5] yardımı ile gerçek zamanlı sesten metine çeviren ve çıktıları txt dosyasına kaydeden basit bir Tkinter uygulaması yapıldı. Uygulama arayüz görüntüsü Şekil 10'de bulunmaktadır.




Şekil 10: Whisper Gerçek Zamanlı Tkinter Uygulaması Arayüzü

## 3.13 PyQt

PyQt, bir Python eklentisi olarak uygulanan, platformlar arası GUI araç seti Qt'nin bir Python bağlantısıdır [4].

PyQt, İngiliz Riverbank Computing firması tarafından geliştirilmiş ücretsiz bir yazılımdır.

PyQt, yaklaşık 440 sınıftan oluşmaktadır. Ayrıca, aşağıdakiler dahil 6.000'den fazla işlev ve yöntem uygulamaktadır:

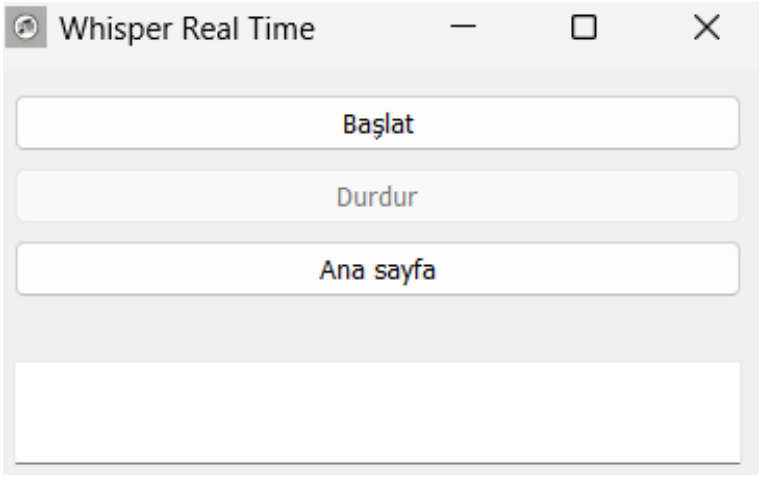
- Önemli bir GUI widget setidir.
- SQL veri tabanlarına erişim sınıflarına sahiptir (ODBC, MySQL, PostgreSQL, Oracle, SQLite)
- QScintilla, Scintilla tabanlı zengin metin düzenleyici gerecidir.
- Bir veri tabanından otomatik olarak doldurulan verilere duyarlı pencere öğeleridir.
- XML ayrıştırıcıdır.
- SVG desteği sağlamaktadır
- Windows'a ActiveX denetimlerini katmak için sınıflar sunmaktadır (yalnızca ticari sürümde)

Bu bağlamaları otomatik olarak oluşturmak için Phil Thompson, diğer projelerde de kullanılan SIP aracını geliştirmiştir.

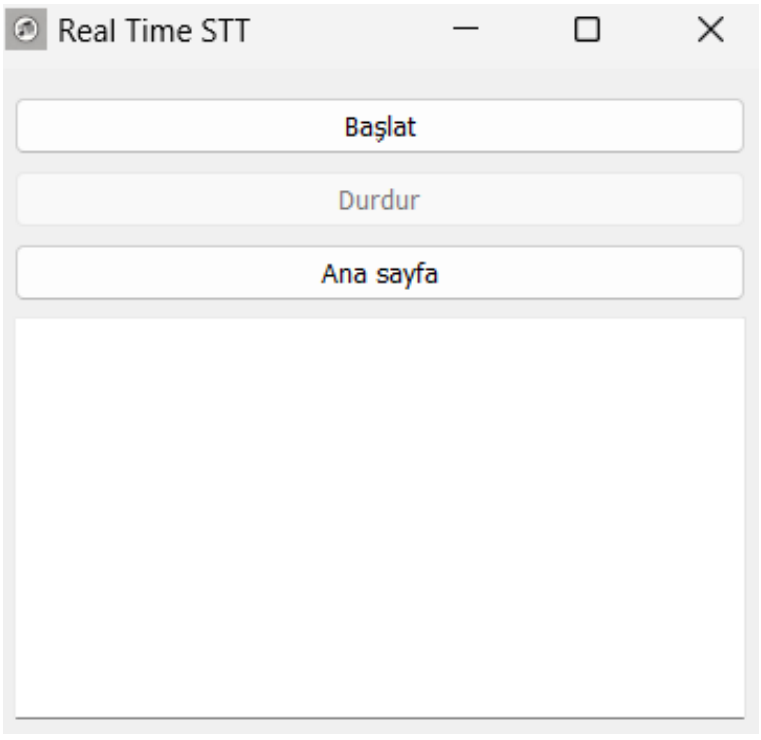
## 3.14 Gerçek Zamanlı Sesten Yazıya Dönüştürmede Uygulaması

Whisper Real Time [2] ve RealtimeSTT [3] içeren PyQt5 kullanılarak ChatGPT [5] yardımı ile gerçek zamanlı sesten yazıya dönüştürme uygulaması yapıldı. Uygulama ilk açılışta ana sayfa ile karşılıyor. Ana sayfada iki adet buton bulunuyor. Butonlardan biri Whisper Real Time çalıştırma sayfasını açarken diğer buton RealtimeSTT çalıştırma sayfasını açıyor. Bu çalıştırma sayfalarında başlat, durdurma ve ana sayfa butonları ile çıktıların gözükeceği bir text alanı bulunuyor. Whisper Real Time sayfasında anlık olarak çıktılar txt dosyasına kaydediliyor ve cümle cümle uygulama ekranında gözüküyor. RealtimeSTT sayfasında ise durdur butonuna basınca çıktılar txt dosyasına kaydediliyor ve bütün cümleler eklemeli olarak ekranda gözüküyor. Her iki çalıştırma sayfasında da bulunan ana sayfa butonu ile ilk açılışta açılan sayfaya geri dönülüyor. Tasarımsal ekleme olarak ise ana sayfa da bulunan butonlar renklendirildi ve Leonardo .Ai [**?**] kullanılarak tasarlanan uygulama ikonu eklendi.


Şekil 11: Uygulama ana sayfası




Şekil 12: Uygulama Whisper Real Time sayfası

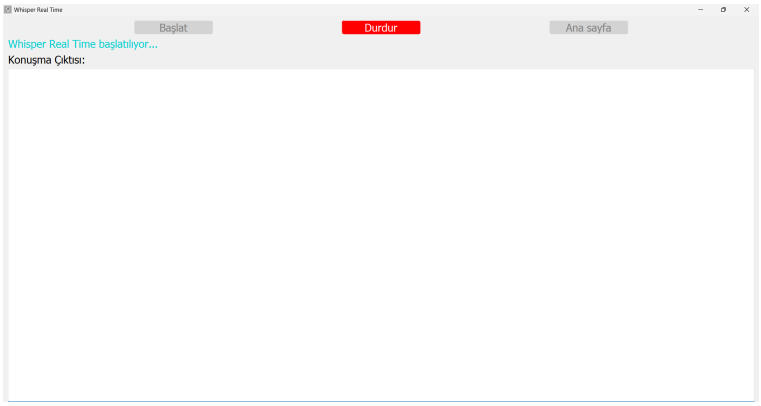



Şekil 13: Uygulama RealtimeSTT sayfası

### 3.14.1 Tasarımsal Değişiklikler

Genel olarak uygulamada tasarımsal değişikler yapıldı. Whisper Real Time ve RealtimeSTT ekranları pencereli tam ekran açılacak şekilde ayarlandı. Bütün ekranlarda ki butonlar renklendirildi ve hizalandı. Uygulamada ki her yazının fontu büyütüldü böylece daha okunaklı bir görüntü elde edilmiş oldu. Whisper Real Time Ekranında ChatGPT yardımıyla tek cümle çıkacak şekilden bütün cümlelerin gözükeceği şekle getirildi.


Şekil 14: Uygulama ana sayfası




Şekil 15: Uygulama Whisper Real Time sayfası


Şekil 16: Uygulama RealtimeSTT sayfası

### 3.14.2 Konfigürasyon Değişiklikleri

Whisper Real Time ve RealtimeStt daha stabil çalışabilmesi için gerekli konfigürasyon değişikleri yapıldı. Whisper Real Time Şekil 17'de ki kodda record timeout değeri 2'den 1'e indirilerek ses kaydının uzunluğu 1 saniyeye düşürüldü ve phrase timeout değeri 3'den 1'e indirilerek ses parçaları arasında ki uzunluk 1 saniyeye düşürüldü. Böylece transkripsiyon hızı arttırıldı.

```
parser.add_argument("--record_timeout", default=1,
                    help="How real time the recording is in seconds.", type=float)
parser.add_argument("--phrase_timeout", default=1,
                    help="How much empty space between recordings before we "
                        "consider it a new line in the transcription.", type=float)
```

Şekil 17: Whisper Real Time konfigürasyon

RealtimeSTT kodunda ise Şekil 18'de Silero sensivity 4'den 2'ye, webrtc sensitivity 2'den 1'e, post speech silence duration 0.4'den 0.2'ye, realtime processing pause 2'den 1'e indirildi. Realtime model type ise tiny değerinden base değerine çevrildi.

```
recorder_config = {
    'spinner': False,
    'model': 'small',
    'language': 'tr',
    'silero_sensitivity': 0.2, #Daha düşük değerler daha fazla algılamaya ve hatalı tanımaya neden olabilirken
    'webrtc_sensitivity': 1, #Bu, gerçek zamanlı işleme duyarlılığını kontrol eder.
    'post_speech_silence_duration': 0.2, #Konuşma algılandıktan sonraki sessizlik süresini belirtir. Bu, konuş
    'min_length_of_recording': 0,
    'min_gap_between_recordings': 0,
    'enable_realtime_transcription': True,
    'realtime_processing_pause': 0.1, #Gerçek zamanlı işleme arasındaki duraklama süresini belirtir. Saniye ci
    'realtime_model_type': 'base', #Gerçek zamanlı transkript için kullanılacak model tipini belirtir.
    'on_realtime_transcription_update': text_detected,
}
```

Şekil 18: RealtimeSTT konfigürasyon

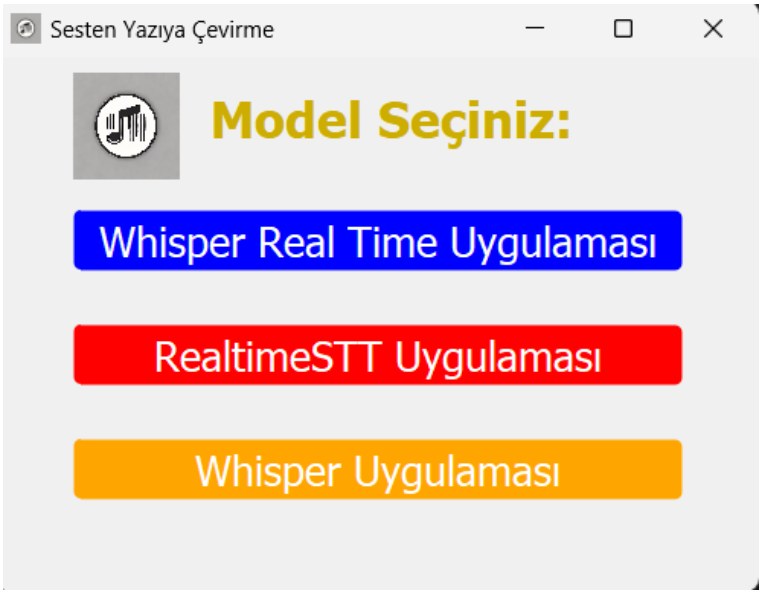
## 3.15 Copyleaks

Copyleaks [19] metin karşılaştırması yapay zekayı kullanarak iki belge arasındaki benzerlikleri kontrol etmek için makine öğrenmesinden yararlanır. Yapay zeka teknolojisi, karşılaştırma kontrolleri sırasında iki metin arasında yeniden ifade edilen cümleleri bulmaya yardımcı olur.

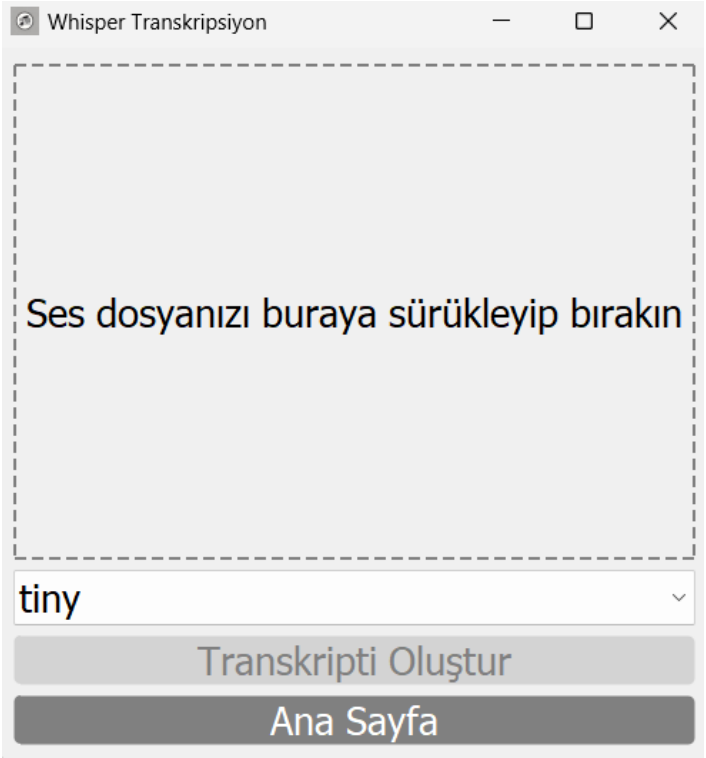
## 3.16 Sesten Yazıya Dönüştürme Uygulaması Son Sürüm

Sesten yazıya çevirme uygulamasına gerçek zamanlı olmayan, ses dosyası yükleyerek yapay zeka ile sesi yazıya çeviren ekran eklendi. Bu ekranda transkripsiyon yapacak Whisper modelini tiny, base, small olarak

seçebilme özelliği eklendi. Uygulama ana sayfasına Şekil 19'de görüldüğü gibi Whisper ekranını açan buton eklendi.




Şekil 19: Uygulama Yeni Ana Sayfa




Şekil 20: Uygulama Whisper Sayfası

# 4 BULGU ve TARTIŞMA

## 4.1 DeepSpeech

DeepSpeech [1] modelini eğitmeye çalışırken Şekil 21'da bulunan hata alınmıştı, bu hata gemini kullanalırak import-cv2.py dosyasında bulunan kodlara Şekil 22'de bulunan kodlar eklenerek çözüldü.

```
%cd /content/DeepSpeech/
! bin/import_cv2.py ../tr/cv-corpus-17.0-2024-03-15/tr
```

```
/content/DeepSpeech
Traceback (most recent call last):
  File "/content/DeepSpeech/bin/import_cv2.py", line 18, in <module>
    from deepspeech_training.util.downloader import SIMPLE_BAR
ModuleNotFoundError: No module named 'deepspeech_training'
```

Şekil 21: DeepSpeech modeli eğitilmeye çalışırken alınan hata

```
import sys
sys.path.append('/content/DeepSpeech/training')
```

Şekil 22: Alınan hatayı düzelten kod [15]

Daha sonra Şekil 23'da bulunan hata alındı.

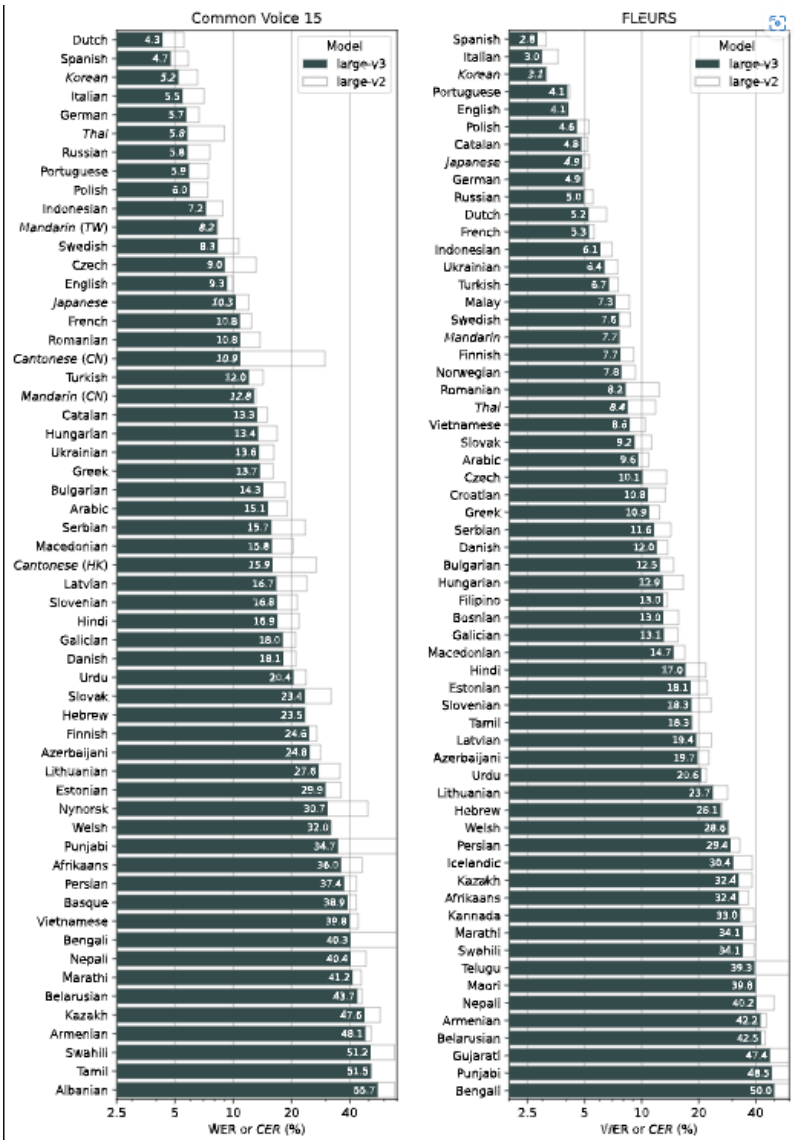
```
Traceback (most recent call last):
  File "/content/DeepSpeech/bin/import_cv2.py", line 31, in <module>
    from ds_ctcdecoder import Alphabet
ModuleNotFoundError: No module named 'ds_ctcdecoder'
```

Şekil 23: DeepSpeech modeli eğitilmeye çalışırken alınan ds-ctcdecoder hatası

Alınan hatalar sonucunda DeepSpeech eğitimi bırakıldı.

## 4.2 Whisper Sözcük Hata Oranları

Aşağıda Şekil 24'de görüldüğü gibi whisper modelinin large-v2 ve large-v3 büyüklüklerinin Common Voice 15 ve Fleurs veri kümelerinin dillere göre sözcük hata oranları verilmiştir. Değer azaldıkça hata oranı azalmaktadır.




Şekil 24: Sözcük hata oranı [6]

### 4.3 Sesten Yazıya Dönüştürme Modellerinin Örnek Sesler ile Test Edilmesi

İlk olarak YouTube'dan test amaçlı ses için videolar bulundu [ [16], [17], [18] ]. Bulunan videoların belirli kısımları RealtimeSTT [3], Whisper Real Time [2], Orijinal Whisper tiny, base, small modellerine dinletildi. Çıkan sonuçlar ile asıl metin Copyleaks [19] sitesi üzerinden metin karşılaştırma özelliğiyle karşılaştırılarak benzerlik hesaplandı. Birinci videoda arka planda çok düşük düzeyde gürültü vardır. İkinci ve üçüncü videolarda röportaj olmasından dolayı iki kişinin sesini içermektedir.

Table 1: Modellerin Çıktılarının Asıl Metne Benzerlik Oranları [19].

| | **Video 1** | **Video 2 (Röportaj)** | **Video 3 (Röportaj)** |
|---|---|---|---|
| **RealtimeSTT** | %92,3 | %84,3 | %71,6 |
| **Whisper Real Time** | %89,7 | %53,9 | %52,7 |
| **Whisper Tiny** | %73 | %83,2 | %29 |
| **Whisper Base** | %91,2 | %73,9 | %72,8 |
| **Whisper Small** | %100 | %85,5 | %77,9 |

# 5 SONUÇ

Yapılan çalışmalar sonucunda yapay zeka ile hem gerçek zamanlı hem de ses dosyası yükleyerek konuşma sesini yazıya dönüştürme uygulaması yapılmış oldu. Böylece herhangi bir video veya anlık konuşma sesinden yazıya çevrilmesi problemi bu uygulama ile çözülmüş oldu.

# Kaynakça


[1] Mozilla, "Training your own model." https://deepspeech.readthedocs.io/en/r0.9/TRAINING.html, 2020.

[2] davabase, "Whisper real time." https://github.com/davabase/whisper_real_time, Mar 14, 2024.

[3] K. Beigel, "Realtimestt." https://github.com/KoljaB/RealtimeSTT, April 14, 2024.

[4] Vikipedi, "Pyqt nedir?." https://tr.wikipedia.org/wiki/Pyqt, Jan 24, 2024.

[5] OpenAI(2024), "Chatgpt(3.5 version)." https://chat.openai.com, May 29, 2024.

[6] openai, "Whisper," Nov 17, 2023.

[7] openai, "Research introducing whisper," September 21, 2022.

[8] L. Roberts, "Understanding the mel spectrogramr," Mar 6, 2020.

[9] G. Kalra, "Attention networks: A simple way to understand cross-attention." https://medium.com/@geetkal67/attention-networks-a-simple-way-to-understand-cross-attention-3b396266d82e, Jul 18, 2022.

[10] wiseman, "py-webrtcvad." https://github.com/wiseman/py-webrtcvad, Feb 15, 2021.

[11] snakers4, "Silero vad." https://github.com/snakers4/silero-vad, Apr 2, 2024.

[12] otakutyrant, "Faster whisper transcription with ctranslate2." https://github.com/SYSTRAN/faster-whisper, Apr 2, 2024.

[13] ErisMik, "Porcupine." https://github.com/Picovoice/porcupine, Apr 15, 2024.

[14] K. Beigel, "realtimestt-test." https://github.com/KoljaB/RealtimeSTT/blob/master/tests/realtimestt_test.py, April 14, 2024.

[15] Google(2024), "Gemini(2024.04.30 version)." https://gemini.google.com, April 29, 2024.

[16] B. Bilim, "Bill gates'e gÖre 2024 sonrasi yapay zeka kehanetlerİ." https://youtu.be/RdZHLiV0UnA?si=rjZSwqrD0njByIAF, Dec 28, 2023.

[17] TV100, "Okan buruk — candaş tolga işık ile az Önce konuştum." https://youtu.be/HbUe1tCbFg8?si=I6r0S8EDpU6QMz9i, June 13, 2023.

[18] K. Sports, "Şampiyonlar ligi'ni kazanmak İstiyorum! — arda güler İle Özel röportaj." https://youtu.be/BSo2KdOb2bI?si=x1023uLvUF68f-Lm, April 29, 2024.

[19] Copyleaks, "Copyleaks." https://app.copyleaks.com/tr/text-compare, 2015.