T.C.
İSTANBUL ÜNİVERSİTESİ-CERRAHPAŞA
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

Python PyAudio Modülü Kullanarak Konuşma Dilinden İşaret Diline Makine Öğrenmesi Tabanlı Tercüme Sistemi

AATIF OSMAN ALTAHIR BAKR

DANIŞMAN
Dr. Öğr. Üyesi Rana ORTAÇ KABAOĞLU

Elektrik-Elektronik Mühendisliği Anabilim Dalı

Elektrik-Elektronik Mühendisliği Programı

Haziran, 2022

Anneme ithaf ediyorum…

BÜTÇE DESTEKLERİ

Python PyAudio Modülü Kullanarak Konuşma Dilinden İşaret Diline Makine Öğrenmesi Tabanlı Tercüme Sistemi

Bu tez çalışması için herhangi bir kurumdan bütçe desteği alınmamıştır.

TEŞEKKÜR

Bu tezin arkasındaki itici güç olan, rehberliği ve nazik sabrı ve desteği bana ilham kaynağı olan proje danışmanım değerli hocam Yrd. Doç. Dr. RANA OTRAÇ KABAOĞLU'na en derin şükranlarımı sunarım.

Tüm hocalarıma, çalışma arkadaşlarıma, arkadaşlarıma ve bu tezde bana doğrudan veya dolaylı yol gösteren herkese en içten teşekkürlerimi sunarım.

Son olarak, beş ay önce Allah'tan rahmet eylesin vefat eden anneme sonsuz desteği ve bana inandığı için sonsuza kadar minnettarım. Ve bu tezi onun ebediyen kalıcı ruhuna adadım.

Haziran 2022 AATIF OSMAN ALTAHIR BAKR

İÇİNDEKİLER

Sayfa No

TEŞEKKÜR ..... **iv**
ŞEKİL LİSTESİ ..... **vi**
TABLO LİSTESİ ..... **ix**
KISALTMA LİSTESİ ..... **x**
ÖZET ..... **xi**
ABSTRACT ..... **xii**
1. GİRİŞ **1**
1.1 Tez Konu Kapsamı 1
1.2 Motivasyon 1
1.3 Literatür Taraması ve Önceki Araştırmalar 2
1.4 Tez Anahatları 4
2. İŞARET DILI **5**
2.1 Arka plan 5
2.2 Amerikan İşaret Dili 6
2.3 ASL Alfabesi 6
2.4 İşaret Dili Unsuru 7
3. Konuşma Tanıma **11**
3.1 Arka Plan 11
3.2 Konuşma Tanıma Nasıl Çalışır 12
3.3 Konuşma Tanıma Sistemi 12
3.4 Web Konuşma API'sinde Konuşma Tanıma 13
3.5 Django'da konuşma tanıma 14
4. YAZILIM MIMARISI **18**
4.1 Arka Plan 18
4.2 Yazılım Sistem Bileşenleri 18
4.3 Python'da Doğal Dil İşleme 18
4.4 Django çerçevesi 33

4.5 Pycharm IDE 36
4.6 HTTP İstek Yöntemleri 37
4.7 Web sitesi ön uç oluşturma dilleri 39
4.8 Hareket Kitaplığı Geliştirme 40
5. TARTIŞMA VE SONUÇ 41
5.1 STSLS'de ASL çevirisinin gösterilmesi 41
5.2 Doğruluk Ölçümleri 42
5.3 Hesap Oluşturma Ekranı 46
5.4 Oturum Açma Ekranı 47
5.5 Ana STSLS Ekranı 48
6. SONUÇ VE ÖNERİLEN 49
6.1 Sonuç: 49
6.2 Öneriler 49
KAYNAKLAR 51

ŞEKİL LİSTESİ

Sayfa No

Şekil 2.1: ASL Alfabesi. [17]'dan basılmıştır. 6
Şekil 2.2: El şekli ayırt edici özellikleri. [17]'dan basılmıştır. 7
Şekil 2.3: STSLS'nin temel yapısı 9
Şekil 3.1: API'nin çalışma şematiği 13
Şekil 3.2: Python'da konuşma tanıma işlemi 14
Şekil 3.3: Konuşma Tanıma API'si oluşturma 15
Şekil 3.4: Get komutunda metin işleme 16
Şekil 4.1: Doğal Dil İşlemenin (NLP) Bileşenleri. 21
Şekil 4.2: Gerekli kitaplıkları pycharm'da içe aktarma. 25
Şekil 4.3: Metni cümle olarak belirtmek için sent_tokenize( ) kullanımı. 26
Şekil 4.4: diğer dillerden cümleleri tokenize edin 26
Şekil 4.5: Metni kelimeler olarak belirtmek için word_tokenize() kullanımı. 27
Şekil 4.6: Durdurulan kelimeler listesini içe aktarma. 27
Şekil 4.7: Durdurulan kelimeleri kaldırma. 27
Şekil 4.8: NLTK çerçevesi ile basit lemmatizasyon örneği. 29
Şekil 4.9: Varsayılan değerler oluşturmak için lemmatizasyonu kullanma. 29
Şekil 4.10: "am", "are", "is", "was", "were" kelimelerinin lemmatizasyonu 30
Şekil 4.11: Şekil 17: Farklı Konuşma Bölümü değerleriyle lemmatizasyon. 30
Şekil 4.12: PoS etiketleme örneği. 32
Şekil 4.13: PoS etiketlemesini gösteren tam Python örneği. 33
Şekil 4.14: pycharm'da Django ortamı 34
Şekil 4.15: Django'da MTV çerçeve mimarisi 35
Şekil 4.16: Pycharm IDE 37
Şekil 4.17: HTTP istek yöntemleri 37
Şekil 4.18: STSLS'de POST yönteminin kullanılması 38
Şekil 4.19: ASL hareketleri kütüphanesi 40
Şekil 5.1: STSLS'de ASL çevirisinin gösterimi 42
Şekil 5.2: Yanlış tanımlanmış kelimelerin basit bir örneği. 43
Şekil 5.3: Genel sistemin ön ucu, Pycharm'da Django kullanılarak tasarlanmıştır. 45
Şekil 5.4: Hesap Oluşturma Ekranı 46
Şekil 5.5: Oturum Açma Ekranı 47

Şekil 5.6: Ana STSLS Ekranı 48

TABLO LİSTESİ

Sayfa No

Tablo 4.1: Kural Tabanlı NLP ve İstatistiksel NLP:.....20
Tablo 4.2: NLTK çerçevesini kullanmanın artıları ve eksileri. 22
Tablo 4.3: spaCy çerçevesinin artıları ve eksileri. 23
Tablo 4.4: Gensim çerçevesinin artıları ve eksileri. 23
Tablo 4.5: Model çerçevesinin artıları ve eksileri. 24
Tablo 4.6: TextBlob kitaplığının artıları ve eksileri. 24
Tablo 4.7: Konuşma bölümü etiketlerinin listesi.....31
Tablo 5.1: Accuracy test of STSLS 43

KISALTMA LİSTESİ

Kısaltmalar Açıklama

API : Uygulama programlama Arayüzü (Application Programming Interface)
NLP : Doğal Dil İşleme (Natural Language Processing)
ASL : Amerikan İşaret Dili (American Sign Language)
TSL : Türk İşaret Dili (Turkish Sign Language)
BSL : İngiliz İşaret Dili (British Sign Language (BSL)
AUSLAN : Avustralya İşaret Dili (Australian Sign Language)
GUI : grafiksel kullanıcı arayüzü (Graphical User İnterface)
SL : İşaret Dili (Sign Language)
STT : Konuşmadan Metne (Speech-To-Text)
SVO : Özne-Fiil-Nesne (Subject-Verb-Object)
SE : İmzalı İngilizce (Signed English)
STSLS : Konuşmadan İşaret Diline Sistemi (Speech To Sign Language System)
ASR : Otomatik Konuşma Tanıma (Automatic Speech Recognition)
HMM : Gizli Markov Modeli (Hidden Markov Model)
VAD : Ses Etkinliği Algılama (Voice Activity Detection)
POS : Konuşma bölümleri (Part Of Speech)
MVT : Model-Görünüm-Şablon (Model-View-Template)
DSF : Django Yazılım Vakfı (Django Software Foundation)
IDE : Entegre Geliştirme Ortamı (Integrated Development Environment)
WER : Kelime Hata Oranı (Word Error Rate)
WAcc : kelime Doğruluğu (Word Accuracy)

ÖZET

YÜKSEK LİSANS TEZİ

PYTHON PYAUDİO MODÜLÜ KULLANARAK KONUŞMA DİLİNDEN İŞARET DİLİNE MAKİNE ÖĞRENMESİ TABANLI TERCÜME SİSTEMİ

AATIF OSMAN ALTAHIR BAKR

İstanbul Üniversitesi-Cerrahpaşa

Lisansüstü Eğitim Enstitüsü

Elektrik-Elektronik Mühendisliği Anabilim Dalı

Elektrik-Elektronik Mühendisliği Programı

Danışman : Dr. Öğr. Üyesi Rana ORTAÇ KABAOĞLU

Sağırlar ve işitme engelliler, belirli ihtiyaçlardan oluşan oldukça büyük bir topluluk oluşturur, ancak işaret dilinde iletişim süreci pek çok insan için zahmetlidir, bu nedenle operatörler ve teknoloji son zamanlarda bu topluluğu hedeflemeye başlamıştır. Ancak bu günlere kadar, konuşulan sözcüklerin yüksek doğrulukta ve daha az hatayla gerçek zamanlı olarak anlaşılabilir işaret diline yorumlanmasını sağlayacak makul maliyetli, özgürce güvenilir bir web sitesi veya tek bir yazılım yoktur.

İşaret dili konularında çok sayıda mevcut araştırma vardır. Ancak araştırma alanlarının çoğu, işaretleri metne veya konuşmaya dönüştürmekle ilgilidir. Bu tezde bunun tam tersi yapılırken, konuşmayı metinle birlikte bir makine öğrenme algoritması kullanarak işaret diline çeviren ve iletişim süreci problemine etki edecek ve daha da basitleştirecek olan bunun tersi yapılmıştır.

Yukarıda belirtilenlerden dolayı bu tez, python ve Django modülleri aracılığıyla bir makine öğrenmesi yaklaşımı kullanarak konuşmadan işaret diline çeviri sistemi sağlayarak bu soruna bir çözüm önermektedir. Sistemin önerilen tasarımı, konuşmadan metne API'leri kullanıp ses sinyallerini metne dönüştürmekle başlayarak, web konuşma API'sı olacak ve ardından Yapay Zeka parçası olarak makine Öğreniminde Doğal Dil İşleme semantiğini kullanacak, kodlama

için Python, ve çeviri sürecini gerçek zamanlı olarak gerçekleştirecek bir web sitesi geliştirmek ve işaret dilini her bir cümle veya kelimenin karşılık gelen metniyle birlikte video formatında görüntülemek için Django kullanılacaktır.

Bu tez, öncelikle sağır insanlara Amerikan işaret dilinde (ASL) bilgi ve hizmet sağlamaya odaklanacak ve manuel ve manuel olmayan işaretler aracılığıyla ASL'nin tüm kelime dağarcığını kapsayacak şekilde genişletilebilecek ölçeklenebilir bir proje geliştirecektir.

Haziran 2022 , ...sayfa.

Anahtar kelimeler: Makine Öğrenimi, Doğal Dil İşleme, Sesten Metne, Amerikan İşaret Dili

ABSTRACT

M.Sc. THESIS

Machine learning-based speech to sign language interpretation system using python PyAudio module.

AATIF OSMAN ALTHIR BAKR

İstanbul University-Cerrahpaşa

Institute of Graduate Studies

Department of Electrical and Electronics Engineering

Electrical and Electronics Engineering

Supervisor : Assist. Prof. Dr. Rana ORTAÇ KABAOĞLU

The deaf and hearing-impaired make up a sizable community of specific needs, yet the process of communicating in sign language is cumbersome to so many people, therefor operators and technology have recently begun to target this community. But until these current days there is no freely reliable accessible websites or a solitary software with a sensible cost to make an interpretation of spoken words into understandable sign language and in real time with high accuracy and less errors.

There is a wealth of existing researches in sign language topic but most of the research areas concern with converting signs to text or speech while in this thesis the opposite has been done which is converting the speech into sign language along with text using machine learning algorithm. and that will have an impact on the communicating process problem and make it even simpler.

Due to what stated above, this thesis proposes a solution to this problem by providing a speech to sign language translation system using machine learning approach through python and Django module. The proposed design of the system will start by changing over the sound signals into text utilizing speech to text Application Programming Interface (APIs) which will be web speech API, and then using the semantics of Natural Language Processing (NLP) in machine Learning as the Artificial Intelligence part, with Python for coding and Django for

developing a website to perform the translation process in real time and displaying the sign language in video format along with the corresponding text of each sentence or word.

This thesis will primarily focus on providing deaf people with information and services in American sign language. and develop a scalable project which can be extended to capture whole vocabulary of American sign language (ASL) through manual and non-manual signs.

June2022, ...pages.

Keywords: : Machine Learning, Natural Language Processing, Audio-To-Text, American Sign Language.

1. GİRİŞ

İşaret dilinin tüm sağırların ana dili olduğu söylenmektedir. Bu, el, kol veya vücut hareketlerinin yanı sıra yüz ifadelerinin bir kombinasyonunu içermektedir. Dünyada 135 farklı işaret dili çeşidi vardır. Bunlardan bazıları Amerikan İşaret Dili (ASL), Türk İşaret Dili (TSL), İngiliz İşaret Dili (BSL), Avustralya İşaret Dili (Auslan) ve çok daha fazlasıdır. Bu projede Amerikan işaret dili kullanılmaktadır. Bu sistem, sağır topluluğun, normal insanlar gibi günlük etkileşimden bilgiye erişmeye ve normal insanlarla iletişim kurmaya kadar her türlü şeyden zevk almasını sağlamaktadır.

İşaret dili, sağır bireylerin ses yolları yerine yüzlerini, ellerini veya gözlerini kullanarak iletişim kurmak için kullandıkları bir iletişim türüdür. Temel dil dizisinin tek tek kelimeleri üzerinde işlemlerini gerçekleştiren doğrudan Çeviri Sistemleri vardır ve orijinal giriş metni üzerinde herhangi bir sözdizimsel analiz yapılmadan çeviri yapılmaktadır. Genel olarak, işaret dilinin kelime sırası İngilizce metindekiyle aynı kalmaktadır. Bu sistemler, kullanıcı girdisini sözdizimsel veya anlamsal düzeyde inceler ve daha sonra kaynak dil yapısından gelen bilgileri okuyan ve hedef dilde sözdizimsel veya anlamsal bir yapı oluşturan bir dizi aktarım kuralı uygulmaktadır. Daha sonra bu dilsel yapıyı hedef dil yüzey formuna dönüştürmek için bir nesil bileşeni kullanılmaktadır.

1.1 Tez Konu Kapsamı

Konuşmanın işaret diline dönüştürülmesi buradaki ana kapsamdır. Bu proje, öncelikle konuşmadan metne API (web konuşma API'sı) ile konuşulan ses sinyallerini metne dönüştürmeye başlayacaktır. Daha sonra, elde edilen metni daha küçük anlaşılabilir parçalara ayırmak için Doğal Dil İşlemenin anlamını kullanarak bu, giriş olarak önceden tanımlanmış işaret dilinin Veri kümeleriyle birlikte Makine Öğrenimi gerektirmektedir. Böylece nihayet sistem web sayfası dönüştürülen sesi işaret diline gösterebilmektedir.

Kullanılacak ana platform, sesi veya konuşmayı metin ve işaretlere başarılı bir şekilde dönüştürmek için bir web uygulaması tasarlamak için Python 3.10.4. kitaplıkları ve Django platformu aracılığıyla uygulanan yazılım olacaktır.

1.2 Motivasyon

Bu sistemin geliştirilmesi, eski tip İşaret Dili çevirme yöntemleri (yorumlama yapan gerçek bir kişi) veya karmaşık log yöntemi yerlerine, modern web uygulamaları, Makine Öğrenimi ve Doğal Dil İşleme teknolojilerinden yararlanarak işitme engelli kişiler ile normal insanlar arasındaki boşluğu kapatmak için bir çözüm olacaktır. Böylece sistem, medya ajansları için

özellikle vasıflı çalışanları istihdam etme maliyetini azaltacak ve haber bölümlerini veya talk show'larını yorumlamak için işçileri işaret dili tercümanı olarak işe almaları gerekmeyebilir ve fazladan tercüman kiralamak zorunda kalmayabilirler. Kesin olarak maaş maliyetini artıracaktır, ayrıca ara tercümanların bulunduğu tüm alanlar bu projeden tam olarak yararlanacaktır.

1.3 Literatür Taraması ve Önceki Araştırmalar

Dünyanın her yerinden araştırmacılar otomatik işaret dili üretimi üzerinde çalışılmaktadır. Bu otomatik sistemler, metin veya konuşmayı girdi olarak almak ve bunun için animasyon üretmektedir. Bir araştırma groubu [1], canlı modda girilen İngilizce konuşmayı eşdeğer metin ve video imzalı İngilizceye aynı anda dönüştürmek için son teknoloji GUI yazılımı sunulmuşlardır. Bunun merkezinde, konuşma tanıma sistemi olarak Sphinx 3.5 kullanılmıştır. Doğruluk açısından, veri tabanı oluşturma zorluğu nedeniyle doğruluğun oldukça düştüğü çalışma boyunca gösterilmiştir. Natural Language işaret diline (SL) çevirisi, araştırmanın ana kapsamıymıştır [2]. Sistemlerinin temel fikri, insan sesini SL'ye çevirmektir. Sistem, uygun işareti/hareket göstermek için veri tabanında önceden depolanmış SL videoları ile yakalanan sesi eşleştirecek ve böylece normal bir kişi ile işitme engelli bir kişi arasında alternatif bir etkileşimli iletişim yolu sağlayacaktır. Sistemin doğruluğu %80,3 olarak ölçülmüştür.

Diğer çalışmada [3], çoklu sensörlerden oluşan akıllı eldivenler geliştirildi ve farklı işaret dili hareketlerini karşılık gelen metne çevirmek için düşük güçlü bir baskılı devre kartı kullanılmıştır. Bluetooth aracılığıyla bir akıllı telefona veya bilgisayar ekranına iletilen Amerikan İşaret Dilinin her harfi için belirli bir kombinasyon vardır. Devre kartı, yalnızca 26 harfle sınırlı olan özel yapım bir çözümdür.

Konuşma tanımanın temel görevi, bir akustik bilgi akışından bir dizi sözcük türetmektir. Otomatik konuşmayı anlama, anlamı çıkarmayı (örneğin, bir veritabanı sorgusu) ve konuşmaya yanıt olarak eylemler üretmeyi kapsayan daha geniş bir sorundur. Semantik, diyalog oluşturma ve diğer işlevlere ayrılmış sistem bileşenleri ile konuşma tanıma alt sistemi arasındaki etkileşim, birçok uygulamada çok önemli olabilmektedir [4].

Araştırmacılara göre [5] İşaret dili dönüştürme için bir hareket yakalama sistemi (Microsoft Kinect Sensor XBOX 360) ve ses dönüştürme için bir ses tanıma sistemi kullanılmıştır. İşaretleri yakalar ve ekranda yazı olarak üretmektedir. Ayrıca kullanıcının sesini kaydederek işaret dili anlamını ekranda hareketli görüntü veya video olarak göstermiştir. İşaret dili tanıma, en önemli araştırmalardan biridir ve Amit Kumar Shinde'nin Marathi'deki işaret dilini metne

ve tam tersi çalışmasına göre, işitme sorunları olan bireyler için en doğal ve yaygın iletişim aracıdır [6]. Bir tercümanın yokluğunda, bir el hareketi tanıma sistemi, sağır kişilerin normal insanlarla iletişim kurmasına yardımcı olabilmektedir. Sistem hem çevrimdışı modda hem de web kamerası aracılığıyla çalışılmıştır.

Anbarasi Rajamohan, Hemavathy R. ve Dhanalakshmi [7] eldiven tabanlı bir sağır-dilsiz iletişim yorumlama sistemi geliştirmişlerdir. Eldivende beş esnek sensör, dokunmatik sensör ve bir ivmeölçer bulunmaktadır. Hareket, kontrolör tarafından önceden programlanmış çıkışlarla eşleştirilmektedir. Tercümanın performansı on harf (A,B,C,D,F,I,L,O,M,N,T,S,W) boyunca değerlendirilmiştir.

Neha V. Tavari A. V. Deorankar Dr. P. N. Chatur'un raporunda [8] belirtildiği gibi, birçok fiziksel engelli insanın düşüncelerini ifade etmek ve dünyanın geri kalanıyla iletişim kurmak için işaret dili tercümanlarına güvendiğini tartışmıştır. Proje, bir web kamerası kullanılarak çekilen elin görüntüsünü tanıtmıştır. Alınan görüntü daha sonra işlenip özellikler çıkarılmıştır. Özellikler, tanıma için bir sınıflandırma algoritmasına girdi olarak kullanılmıştır. Hareket tanımlanıp konuşma veya metin oluşturmak için kullanılmıştır. Bu sistemde flex sensör kararsız analog çıkış verir ve ayrıca birçok devre gerektirebildiği için çok pahalıdır.

Diğer bir araştırma alanı da Görüntü Tabanlı İşaret Dili Çevirmenleriydi, Görüntü İşleme ve makine öğrenimi tekniklerini kullanarak, işaretin görüntüsü karşılık gelen kelimeye çevrilmektedir. Görüntünün özniteliklerinin çıkarılması, görüntü işleme ile yapılmış ve ardından öznitelik vektörü, makine öğrenmesi teknikleri kullanılarak sınıflandırılmıştır. Akıllı telefon ([9], [10], [11], [12]) dışında herhangi bir özel donanıma ihtiyaç yoktur ve bu sistemlerin dezavantajları şu şekildeymiştir:

Akıllı telefon cihazları genellikle güç tüketmektir.

Düşük ışık koşullarında her zaman iyi çalışmazlardır.

İşaretleri yapan kişiden uzak bir yere yerleştirilmesini zorunda olduğu için her zaman pratik değildir.

1.4 Tez Anahatları

1. Bölüm:

Bu bölüm, önceki çalışmaların kısa bir literatür taraması ile birlikte bu yüksek lisans tezinin arka planını ve hedeflerini tanıtmaktadır.

2. Bölüm:

Bu bölüm işaret dilinin teorik temellerini, unsurlarını, gramerini ve yapısını açıklamaktadır.

3. Bölüm:

Bu bölüm, sistemin konuşma tanıma kısmını ve çalışmanın bir parçası olarak kullanılan konuşmadan metne (STT) öğelerini tartışmaktadır.

4. Bölüm:

Bu bölüm, kayıtları insan hareketi tanımayla eşleştirmek için kullanılan makine öğrenimi tabanlı NLP tercihleri sisteminin yazılım geliştirmesinin kullanımını tartışmakatdır. Ayrıca jest kitaplığının geliştirilmesini de içermektedir.

5. Bölüm:

Bu bölüm, kullanıcı arayüzü ile birlikte sistemin Uygulanmasını ve sonuçlarını sunmaktadır.

6. Bölüm:

Bu bölümde tezin sonuçları ve gelecekte yapılacak çalışmalar tartışılmaktadır.

2. İŞARET DILI

2.1 Arka plan

İşaret dili, bir mesajı ifade etmek için el hareketlerini ve vücut hareketlerini kullanan bir iletişim tarzıdır. Bu, bir konuşmacının düşüncelerini sorunsuz bir şekilde tasvir etmek için el biçimlerinin, ellerin, kolların veya vücudun yönelimi ve hareketinin yanı sıra yüz ifadelerini birleştirmeyi içerebilmektedir. En az 70 milyon insan konuşma ve işitme bozukluğu ile doğmakta veya bir kaza sonucu bunları geliştirmektedir [13]. Farklı iletişim yöntemleri kullandıkları için onlarla iletişim kurmak bizim için zor. İşaret Dili (SL), konuşma ve işitme engelliler arasında sıklıkla kullanılan bir iletişim şeklidir. Bu nedenle, muhtemelen onlarla daha kolay iletişim/etkileşim kurmanın tek yolu onların dilini - işaret dilini öğrenmektir [14].

İşaret dili, görsel-mekansal bir iletişim aracıdır. Öncelikle Sağır insanlar tarafından geliştirilen ve kullanılan dil, herhangi bir konuşma (vokal-işitsel) dili kadar karmaşık ve karmaşıktır. İşaret dili, konuşulan dillerden ayrı olarak kendi başına mevcuttur. Örneğin Auslan (Avustralya İşaret Dili) ve Ameslan (Amerikan İşaret Dili) İngilizce'nin imzalı çevirileri değildir ve buna dayalı değildir. İşaretli dillerin, konuşulan diller gibi, kendi gramer kuralları, sözdizimi, fonoloji, morfoloji ve diğer dilbilimsel yönleri ile kendi karmaşıklıkları vardır. İşaret ve konuşma gibi amodal diller, kiplikten etkilenmez. İletişim kurma yeteneği insan beynine dayanmaktadır.

Semboller (kelimeler), sözdizimi ve cümle yapısının tümü dilin bileşenleridir. Her ne kadar - konuşmanın "onomatopoeia"sına benzeyen- bazı ikoniklikler olsa da - ASL işaretleri/kelimeleri herhangi bir konuşulan dilinkiler kadar soyuttur. Dilbilim ve sinirbilim çalışmalarına göre, işaretli ve sözlü diller, benzer bir dil edinimi zaman çizelgesi de dahil olmak üzere beynin aynı dil bölgelerinde işlemektedir. Modalitelere gelince, bunlar beynin farklı bölümlerinde işlenmektedir [15].

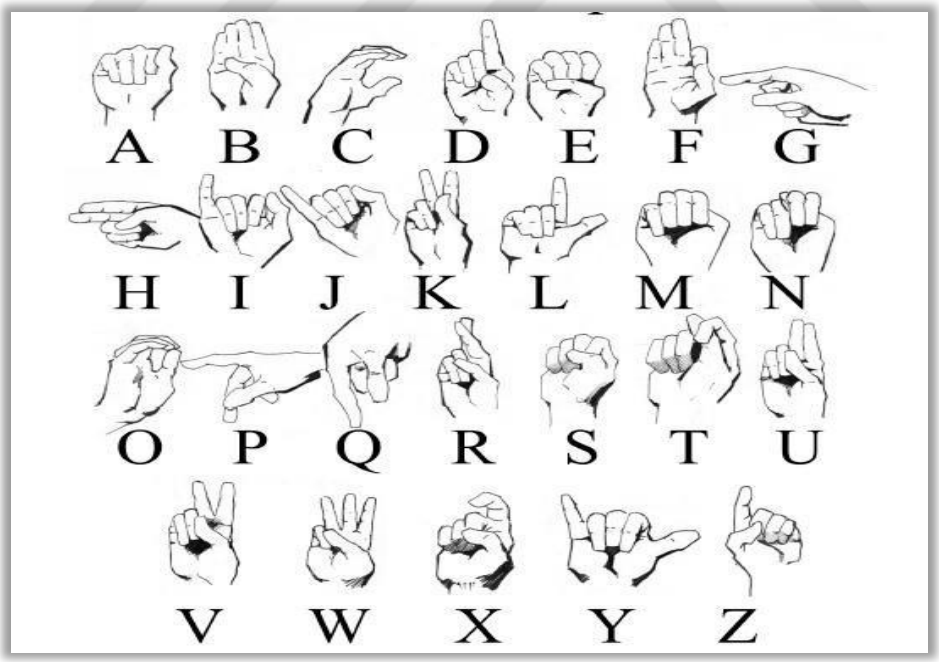
Dünyada birçok farklı konuşulan dil olduğu gibi, birçok farklı işaret dili vardır. Başka bir deyişle, işaret dili her yerde bulunmazdır. Ayrıca işaret dilleri konuşma dillerinden farklıdır. Örnek olarak, Kuzey Amerika'da ASL (Amerikan İşaret Dili), Birleşik Krallık'taki BSL'den (İngiliz İşaret Dili) ve Avustralya'daki Auslan'dan (Avustralya İşaret Dili) farklıdır. ASL, birçok kişinin inandığı gibi "imzalı İngilizce" veya "eldeki İngilizce" değildir. İngilizce'den tamamen farklı bir dildir.

2.2 Amerikan İşaret Dili

ASL, iletişim kurmak için eller ve yüzler gibi görsel artikülatörleri kullanan görsel-uzaysal bir dildir. Kendi grameri, yapısı ve kelime hazinesi ile eksiksiz bir dildir. Fonoloji, morfoloji ve sözdizimi dahil olmak üzere tüm dil özelliklerine sahiptir. ASL ayrıca, kişisel adlar, marka adları (ASL işaretlerinin olmadığı durumlarda), şehir adları ve belirli yabancı sözcükler (dahil İngilizce).

2.3 ASL Alfabesi

Tek elle kullanılan ASL'nin alfabesi, İngiliz alfabesine karşılık gelen 26 manuel alfabetik harften oluşur. İngilizce bir kelimeden, bir kişinin adından veya başka herhangi bir şeyden bir dizi alfabetik harfin parmakla hecelenmesi için bir tekniktir. Dünyanın her yerinde farklı manuel alfabeler bulunabilmektedir. Amerikan manuel alfabesi, birkaç değişiklikle 18. yüzyılın Fransız manuel alfabesine dayanmaktadır. Kökenleri, çeşitli işaretli dillerin manuel alfabelerinde keşfedilebilmektedir. Örnek olarak Alman ve Amerikan manuel alfabeleri çok benzemektedir. Yine de ASL ve Alman İşaret Dili çok farklıdır [16].



Şekil 0.1: ASL Alfabesi. [17]'dan basılmıştır.

Genellikle manuel alfabe olarak bilinen işaret dili alfabesi, başka bir dilde sözlü veya yazılı bir kelime oluşturmak için bir dizi alfabetik harfin parmakla hecelenmesi için kullanılmaktadır. Bir Ameslan (Sağır/ASL-hoparlör) İngilizce veya yabancı bir ifadeyle konuştuğunda, parmakla

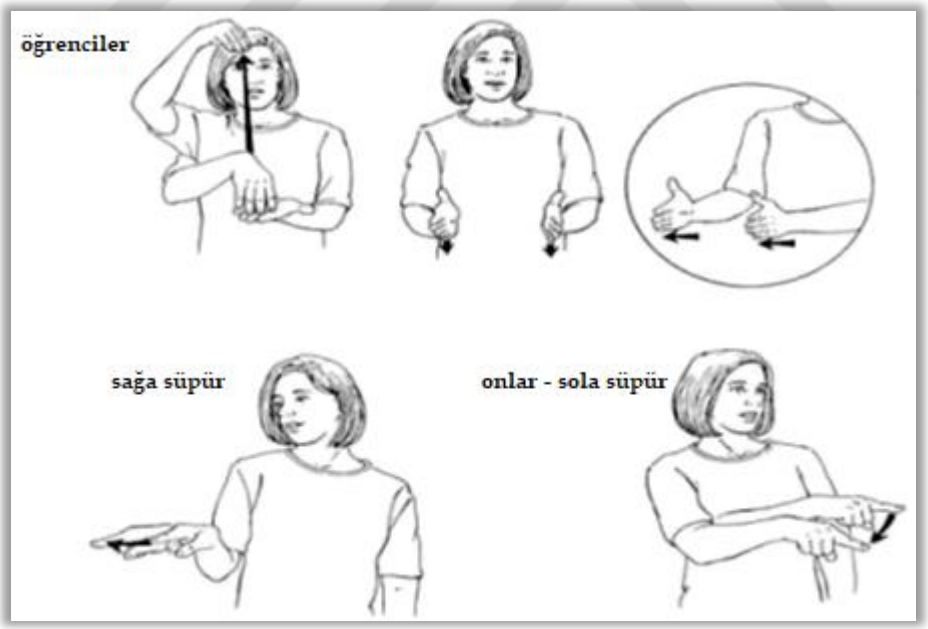
yazım kullanılmaktadır. Bir kişinin adı, bir yerin adı, başka bir sözlü veya yazılı dilden bir kelime vb.

2.4 İşaret Dili Unsuru

Bu tez için işaret dili olarak Amerikan İşaret Dili (ASL) seçilmiştir. Amerikan İşaret Dili (ASL), dünyada en sık kullanılan işaret dilidir. Amerika Birleşik Devletleri ve Kanada'da bu dil yaygın olarak işaret dili olarak kullanılmaktadır. Ayrıca, ASL'nin sayısız lehçesi öncelikle Batı Afrika ve Güneydoğu Asya'da kullanılmaktadır. ASL ve TSL çok sıkı bir ilişkiye sahiptir.

2.4.1 Fonetik

Her ASL işareti hareketi birkaç benzersiz bileşenden oluşmaktadır. Duruma bağlı olarak, bir işaret iki veya bir eli kullanabilmektedir. El, vücut üzerinde belirli bir pozisyonda veya belirli bir oryantasyonla "imza boşluğunda" olabilmek (örnek olarak, bir işaret parmağı uzatılmış kapalı yumruk) ve hareket edebilmektedir. Bu yönlerden biri değiştirildiğinde, işaretin anlamı önemli ölçüde değişebilmektedir.



Şekil 0.2: El şekli ayırt edici özellikleri. [17]'dan basılmıştır.

2.4.2 Dilbilgisi

2.4.2.1 **Parmakla Yazım**

Amerikan manuel alfabesi, İngilizce dilinde kelimeleri hecelemek için kullanılabilecek ASL'de 26 işaretten oluşan bir koleksiyondur. Bu işaretler 19 ASL el şeklinden oluşur. Örneğin, 'p' ve 'k' işaretleri aynı el şekline ancak farklı yönlere sahiptir. Yaygın bir yanlış algılama, ASL'nin yalnızca parmakla yazımdan oluşmasıdır. Bu yöntem (Rochester Metodu) kullanılmış olsa da, ASL değildir.

Parmakla yazım, sözcükleri bir dilden diğerine dahil etme işlemi olan bir ödünç alma türüdür. Parmakla yazım, yerel bir ASL karşılığı olmayan özel isimler ve teknik ifadeler için ASL'de kullanılır. İngilizce 'on'dan O-N ve İngilizce 'apartment'tan A-P-T gibi parmakla yazılan başka kredi terimleri de vardır; bunlar ya çok kısa İngilizce kelimelerdir ya da daha uzun İngilizce ifadelerin kısaltmalarıdır. Parmakla yazım, aksi takdirde imzalanacak bir kelimeyi vurgulamak için de kullanılabilir.

2.4.2.2 **Sözdizimi Yapısı**

ASL'de, kelime sırası genellikle bu temel kelime sırasını etkileyen çeşitli faktörlerle birlikte özne-fiil-nesne (SVO) idi. Temel SVO cümlelerinin imzalanmasında duraklama yoktur:

MOTHER – LOVE – CHILD

"The mother loves the child."

Bununla birlikte, ASL, bir cümlenin konusunun, topikalizasyon olarak bilinen bir fenomenin cümlenin önüne itilmesine izin verdiği için başka bir kelime sıralaması oluşabilir. İşte bir diyalog örneği:

1. Aatif: EXCUSE-me, ME NAME A-A-T-I-F. NAME YOU?

Excuse me, I'm Aatif, what is your name?

2. Fatih: ME F-A-T-I-H. NICE me-MEET-you.

I'm Fatih. It is nice to meet you.

3. Aatif: NICE me-MEET-you. YOU TAKE-UP ASL?

It is nice to meet you. Are you taking ASL?

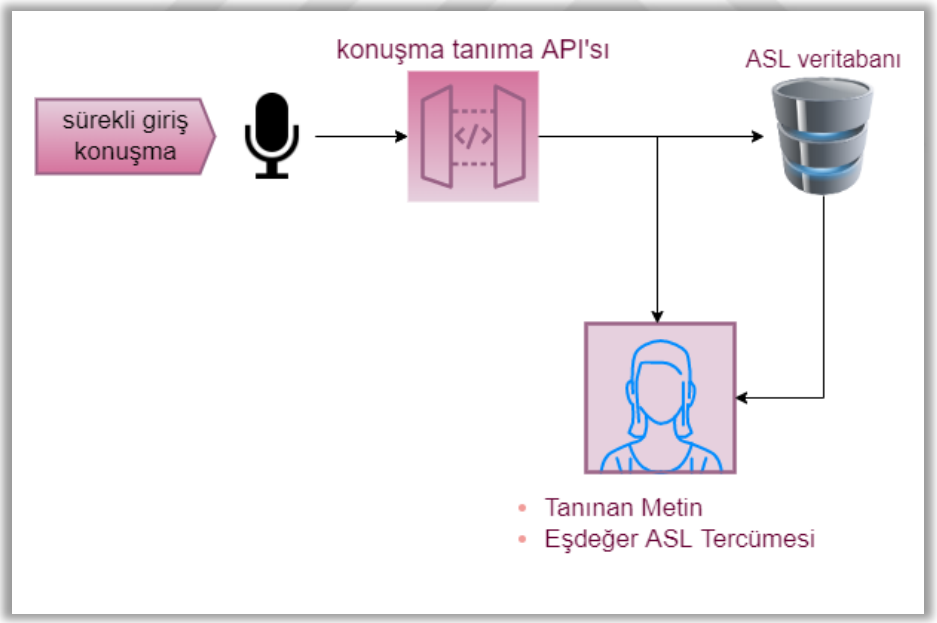
4. Fatih: YES, ME TAKE-UP ASL. YOU?

Yes, I am taking ASL. How about you?

5. Aatif: SAME-HERE. ME TAKE-UP ASL.

Same as you, I am taking ASL too.

Sistemin temel yapısı Şekil 3'te gösterilmektedir. Bir mikrofon, canlı olarak söylenen giriş sesini yakalamakta ve daha sonra bir konuşma tanıyıcı tarafından metne kopyalanmaktadır. Konuşma tanıyıcı motoru, bunu yapmak için Google Web Speech API'sini kullanmıştır. Tanınan metin, bir eşleşme aramak için kelime kelime bir ASL veritabanına girilecektir. Veritabanında, her temel kelime için bir video klip ile önceden kaydedilmiş belirli miktarda video işareti vardır. Bir eşleşme bulunursa, uygun ASL çevirisi, ASL sözdizimi yerine İmzalı İngilizce (SE) kılavuzu kullanılarak İngilizce'nin karşılığı olarak görüntülenecektir. Aksi takdirde, kelime parmaklar kullanılarak yazılacaktır. Sonunda, nihai sonuç olarak, hem tanınan metin hem de ASL çevirisi aynı anda gösterilecektir.



Şekil 0.3: STSLS'nin temel yapısı

Sonuç olarak, web uygulaması konuşmayı girdi olarak kabul eder, metne dönüştürür ve ardından hareketleri İşaret Dili'nde görüntüler.

Pycharm, sistemin ön ucunu oluşturmak için kullanılmaktadır.

PyAudio, bir mikrofon kullanarak konuşma girişini işlemek için kullanılmakatdır.

Web Speech API motoru konuşmayı tanımak için kullanılmaktadır.

Metin daha sonra doğal dil işleme (NLP) kullanılarak önceden işlenmektedir.

Son olarak, veritabanından sözlüklere dayalı makine çevirisi tamamlanmaktadır.

3. KONUŞMA TANIMA

3.1 Arka Plan

Çalışmada kullanılan konuşma tanıma ve konuşmadan metne (STT) özellikleri bu bölümde tartışılmaktadır. Bir bilgisayarın konuşulan sözcüklerden girdi almasına izin veren bir sistem, konuşma tanıma veya otomatik konuşma tanıma (ASR) olarak bilinir. Konuşma tanımanın kökleri 1950'lerin başlarındaki Bell Labs araştırmalarına dayanmaktadır. Erken sistemler yalnızca tek bir konuşmacı ve yaklaşık bir düzine kelimeden oluşan küçük bir kelime dağarcığı içermektedir. Günümüzde konuşma tanıma sistemleri, öncülerinden çok yol kat etmiştir. Birkaç dilde geniş kelime dağarcığına sahiptirler ve konuşmayı birden fazla konuşmacıdan ayırt edebilmektedir.

Gizli Markov Modeli (HMM), modern ses tanıma sistemlerinin çoğunluğu tarafından kullanılmıştır. Bu yöntem, bir konuşma sinyalinin, yeterince kısa bir zaman ölçeğinde (örneğin, on milisaniye) üzerinde çalışıldığında, makul bir şekilde durağan bir süreç, yani istatistiksel parametreleri zamanla değişmeyen bir süreç olarak temsil edilebileceği fikrine dayanmaktadır.

Konuşma verileri, standart bir HMM'de 10 milisaniyelik bölümlere ayrılmaktadır. Her bölümün güç spektrumu, sinyalin gücünün frekansın bir fonksiyonu olarak basitçe görselleştirilmesi olan, cepstral katsayılar adı verilen gerçek değerlerin bir vektörüne eşlenmektedir. Bu vektörün boyutu tipik olarak küçüktür—10 kadar küçükken, daha hassas sistemler 32 veya daha yüksek boyuta sahip olabilmektedir.

Bu vektörlerin bir dizisi HMM'nin nihai çıktısıdır.

Konuşmayı metne dönüştürmek için, bir dizi vektör, konuşmanın temel birimleri olan bir veya daha fazla ses birimiyle eşleştirilmektedir. Bir fonem sesi, konuşmacıdan konuşmacıya ve aynı konuşmacı tarafından bir heceden diğerine farklılık gösterdiğinden, bu hesaplama pratik gerektirmektedir. Belirlenen sesbirim dizisini oluşturan en olası sözcük (veya sözcükler), daha sonra belirli bir algoritma kullanılarak belirlenmektedir.

Tüm bu süreç, beklendiği gibi, hesaplama açısından pahalı olabilir. HMM tanımadan önce, sinir ağları, özellik değiştirme ve boyut azaltma teknikleri aracılığıyla ses sinyalini azaltmak için birçok mevcut ses tanıma sisteminde kullanılmaktadır. Ses Etkinliği Algılamaler (VAD) ayrıca bir ses sinyalini yalnızca konuşma içermesi muhtemel kısımlarla sınırlamak için kullanılmaktadır. Bu, tanıyıcının sinyalin gerekli olmayan kısımlarını inceleyerek zaman kaybetmesini önlmektedir.

3.2 Konuşma Tanıma Nasıl Çalışır

Sözcükleri sayısallaştırarak ve dijital sinyali cihazda kayıtlı belirli bir modelle eşleştirerek, konuşma tanıma teknolojisi bir bilgisayarın konuşulan kelimeleri algılamasını ve yorumlamasını sağlamaktadır. Konuşma, konuşma tanımanın ilk bileşenidir. Bir mikrofonla, gerçek ses bir elektrik sinyaline dönüştürülmekte ve daha sonra analogdan dijitale dönüştürücü ile dijital verilere dönüştürülmektedir. Ses sayısallaştırıldıktan sonra, onu metne dönüştürmek için çok sayıda model kullanılabilmektedir.

Konuşulan kelimelerin sonuçları, anlamlı konuşma şeklinde okunaklı harflerle görüntülenebilmektedir. Ardından, bazen konuşma tanıyıcı olarak da bilinen ve kullanıcının gerçek sözlü sözcük örneğini gerektiren ses tanıma aracı vardır. Numuneler, cihaza kaydedilecek sayısallaştırılmış kelimelere dönüştürülecektir. Sözleri jestlerle ilişkilendirmek için referans verisi olarak kullanılırlar. Çoğu konuşma tanıma sistemi hala hoparlörlere bağlıyken, bu, yalnızca bir veya iki kişinin konuşulan sözcüklerini ayırt edebilecekleri ve yalnızca tek tek tümceleri, yani tümceler arasındaki iletim döneminde yer alan sözcükleri tanıyabilecekleri anlamına geliyormuştur. Bölüm 4'te ele alınacak olan NLP'nin avantajları nedeniyle, incelenen sistem konuşmacıdan bağımsızdır ve konuşmacıdan bağımsız olarak konuşulan kelimeleri tanıyabilecektir.

3.3 Konuşma Tanıma Sistemi

Konuşma tanıma sistemlerinin amacı, işitme güçlüğü çeken kişilere yardımcı olmaktır. İşaret dilini girdi olarak metne veya sesi çıktı olarak dönüştüren işaret dilleri üzerine birçok proje geliştirilmiştir. Ancak, sesten işaret diline dönüştürme teknolojileri nadirdir. Hem işiten hem de sağır kişiler tarafından kullanılabilmektedir. Bu proje yeni bir teknoloji sunuyor: Python tabanlı bir sesten işaret diline çevirmendir. Burada ses girdi olarak kullanılır ve sistem web konuşma API'sini kullanarak bu kaydı arar, metni ekranda görüntüler ve ardından sağlanan giriş için işaret hareketleri oluşturmak üzere bir ASL oluşturucu kullanır. Daha sonra cümledeki tüm kelimeler, cümleleri temsil eden çok sayıda videonun bulunduğu veritabanındaki kelimelerle karşılaştırılır. Sözcükler bulunamazsa yerine eş anlamlı sözcükler kullanılır. Sistemin önceden tanımlanmış bir dizi hareketi vardır.

Yüz ifadelerinin işaret dilinin kritik bir özelliğini taşıdığına yaygın olarak inanıldığından, bu araştırma öncelikle yüz ifadelerine odaklanmaktadır. Sonuç olarak, sistemin nihai çıktısı olarak gerçek bir kişinin işaret dili konuşmasına karar verilmiştir. Bu teknik, sağır ve dilsiz insanlar

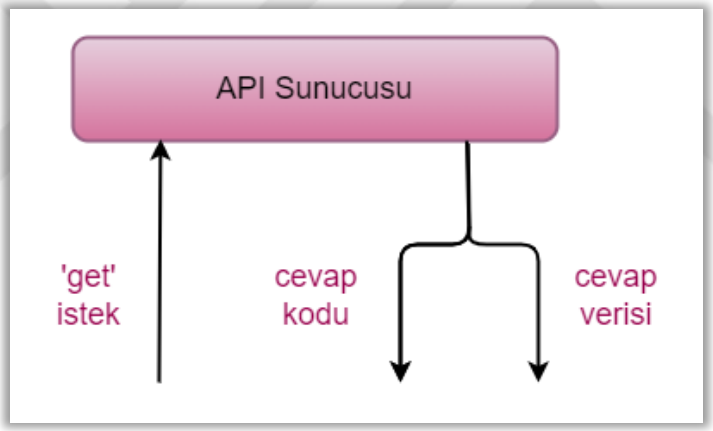
için video klip olmadan devlet web sitelerine erişmek veya çevrimiçi formları tercüman olmadan doldurmak gibi çeşitli durumlarda kullanılabilmektedir.

Ses kayıt bileşeni (bilgisayara bağlı bir mikrofon olacak), dijital sinyal işleme bileşeni (ses tanıma için python modülleri ile PyCharm ortamı olacak) ve metin tanıyıcı bileşen (formda) üç temel bileşendir. NLP birimi).

3.4 Web Konuşma API'sinde Konuşma Tanıma

Bir API veya Uygulama Programlama Arayüzü, kod kullanarak veri almak ve iletmek için kullanılabilen bir sunucudur. API'lerin en tipik uygulaması veri almaktır.

bir API'den veri almak için bir istek gönderilmelidir. İstekler internette yaygın olarak kullanılmaktadır. Örnek olarak, bir kullanıcı bir web sitesindeki belirli bir şablona tıkladığında, web tarayıcısı, kullanıcının ihtiyaçlarını içeren sayfanın içeriğiyle yanıt veren Dataquest web sunucusuna bir istek göndermektedir.



Şekil 0.4: API'nin çalışma şematiği

API istekleri aynı şekilde çalışır: Veriler için bir API sunucusuna bir istek gönderilmesi gerekir ve sunucu yanıt verir.

PyPI'de birkaç konuşma tanıma paketi vardır. apiai, Assemblyai, google-bulut-konuşma, pocketsphinx, konuşma tanıma, watson-geliştirici-bulut ve zekâ ve hepsi çok azdır.

Widget ve apiai gibi bu araçların çoğu, bir konuşmacının amacını belirlemek için doğal dil işleme gibi basit konuşma tanımanın çok ötesine geçen yerleşik işlevlere sahiptir. Web-speech-

API gibi diğerleri tamamen konuşmadan metne dönüştürmeye adanmıştır ve bu araştırmada kullanılacak paket budur.

Konuşma Tanıma kitaplığı, çok sayıda ana konuşma API'si için bir sarmalayıcı işlevi gördüğünden son derece esnektir. Yukarıdakilerden biri olan Google Web Voice API, konuşma tanıma kitaplığında sabit kodlanmış bir varsayılan API anahtarına sahiptir. Konuşma tanıma paketinin esnekliği ve kullanım kolaylığı, onu herhangi bir Python projesi için iyi bir seçim haline getirmektedir.



Şekil 0.5: Python'da konuşma tanıma işlemi

Web Speech API, konuşma verilerini web uygulamalarına dahil etmek için kullanılabilmektedir. Konuşma Sentezi (Metin-Konuşma) ve Konuşma Tanıma (Eşzamansız Konuşma Tanıma), Web Konuşma API'sı tarafından sağlanan ve erişilebilirlik ve kontrol yöntemleri için heyecan verici yeni olanaklar sunan iki bağımsız işlevsellik alanıdır.

3.5 Django'da konuşma tanıma

Konuşma tanıma, bir cihazın mikrofonu aracılığıyla konuşmanın alınmasını ve bir konuşma tanıma hizmeti tarafından bir dil bilgisi listesine karşı test edilmesini gerektirmektedir (temelde, belirli bir uygulamada sözcük dağarcığının tanınması gerekir). Bir sözcük veya tümcecik doğru bir şekilde tanımlandığında, şu şekilde döndürülür: sonuç olarak bir metin dizisi (veya sonuç kümesi) ve sonuç olarak başka işlemler yapılabilmektedir.

SpeechRecognition(), dilbilgisini, sonuçları vb. temsil etmek için birbiriyle yakından ilişkili çeşitli arabirimlerle birlikte Web Speech API'sinde bunun için ana denetleyici arabirimidir. Çoğu durumda, konuşma tanıma için cihazın varsayılan konuşma tanıma sistemi kullanılacaktır - çoğu modern işletim sistemi, sesli komutlar sağlamak için bir konuşma tanıma sistemine sahiptir. macOS Dikte, iOS Siri, Windows 10 Cortana, Android Konuşma vb.

Bir web sayfasındaki Konuşma Tanıma, Chrome gibi bazı tarayıcılarda sunucu tabanlı bir tanıma motoru gerektirir. Ses, tanınma için bir web hizmetine iletildiğinden, çevrimdışı olarak kullanılamazdır.

Prosedür, uygulamanın tanınmasını işlemek için kullanılacak bir ses tanıma nesnesinin oluşturulmasıyla başlamaktadır. Bunun için SpeechRecognition() yapıcısı kullanılmaktadır.

Bu SpeechRecognition nesnesi, onu özelleştirmek için kullanılabilecek birkaç özelliğe sahiptir.

• SpeechRecognition.continulous: Tanıma her başlatıldığında (yanlış) sürekli sonuçların mı (doğru) yoksa tek bir sonucun mu (yanlış) kaydedileceğini belirler.

• SpeechRecognition.lang: Tanıma dilini tanımlar. Bunun düzeltilmesi önerilir çünkü bu iyi bir uygulamadır.

• SpeechRecognition.interimResults: Bu özellik, konuşma tanıma sisteminin ara sonuçları mı yoksa nihai sonuçları mı döndürmesi gerektiğini belirtir. Bu küçük gösteri için, nihai sonuçlar tatmin edicidir.

• SpeechRecognition.maxAlternatives: Sonuç başına döndürülecek maksimum alternatif potansiyel eşleşme sayısını belirler. Bu, bir sonucun çok açık olmadığı ve kullanıcının seçebileceği bir seçenekler listesi göstermek istediğimiz gibi bazı durumlarda yararlı olabilmektedir.

Uygulamaya konuşma tanımayı eklemeden önce konsola aşağıdakileri girerek başlanır:

```
function record(){
        var recognition = new webkitSpeechRecognition();
        recognition.lang='en-IN';

        recognition.onresult = function(event){
                console.log(event)
                document.getElementById('speechToText').value = event.results[0][0].transcript;
        }
        recognition.start();
```

Şekil 0.6: Konuşma Tanıma API'si oluşturma

Bu üç kod satırını açıklığa kavuşturmak için Basitçe, bir Konuşma Tanıma API'sinin ("WebKit" ön ekine sahip satıcı) oluşturulduğunu ve ona sesten metne hizmetten aldığı sonuçları rapor etmesini ve dinlemeye başlamasını söylediğimizi beyan edilebilir.

Mikrofona bir şey söylendiğinde ve ardından konuşmayı bitirdiğimizde, konsolda bir Konuşma Tanıma Olayı kaydedilmelidir.

SpeechRecognitionEvent, bir konuşma tanıma olayını temsil eden bir nesnedir. SpeechRecognitionResult nesnelerinin bir listesi olan Results özelliği en önemli özelliktir. Dinlemeyi bırakmadan önce sadece bir şey belirttik, bu nedenle sadece bir sonuç nesnesi var. Bu sonucu incelediğimizde, bir dizi SpeechRecognitionAlternative nesnesi alırız; bunlardan ilki, konuşulanların dökümünü ve 0 ile 1 arasındaki bir güven değerini içerir. Tanıma hizmeti varsayılan olarak yalnızca bir seçenek döndürür, ancak biz daha fazlasını almayı seçebilir, bu da kullanıcıların söylediklerine en yakın seçeneği seçmelerine izin verirsek faydalı olabilir.

Chrome, sesi alıp deşifre için Google'ın sunucularına göndereceğinden, SpeechRecognition'ın, kullanıcıların onu kullanabilmesi için çevrimiçi olmasını gerektiren bir sunucu tarafı API olduğunu ima etmektedir.

(document.getElementById), kullanılacağı bazı öğelere referansı olacaktır.

Son olarak, ekrana dokunulduğunda veya tıklandığında konuşma tanıma hizmetinin başlaması için bir tıklama işleyicisi eklenir. Bu, SpeechRecognition.start yöntemi () kullanılarak gerçekleştirilir.

Konuşma tanıma başlatıldıktan sonra sonuçları almak için kullanılabilecek çeşitli olay işleyicileri vardır.

Başarılı bir sonuç alındığında tetiklenen sonuç olayı, burada kullandığımız en tipik olaydır: (Ses tanıma hizmeti bir sonuç verdiğinde tetiklenir - bir kelime veya kelime öbeği olumlu olarak tanınmış ve uygulamaya iletilmiştir.)

```
document.getElementById('speechToText').value = event.results[0][0].transcript;
```

Şekil 0.7: Get komutunda metin işleme

İkinci satırın adım adım açıklaması aşağıdaki gibi olacaktır: SpeechRecognitionEvent.results özelliği tarafından SpeechRecognitionResult nesnelerini içeren bir SpeechRecognitionResult List nesnesi döndürülür. Bir alıcısı olduğundan, ona bir dizi gibi erişilebilir - ilk [0], 0 konumunda SpeechRecognitionResult değerini döndürür. Tanınan tek tek sözcükler, her SpeechRecognitionResult nesnesinde SpeechRecognitionAlternative nesneleri olarak depolanır. Bunların da alıcıları vardır, bu nedenle diziler gibi erişilebilirler; ikinci [0], sonuç olarak 0 konumunda SpeechRecognitionAlternative'i döndürür. Daha sonra, algılanan bireysel

sonucu içeren bir diziyi çıkarmak için transkript özelliğini kullanılır, ses tanıma kutusunu bu diziye ayarlarız ve kullanıcı arayüzünde tanınan rengi bir tanı mesajı olarak bildirilir.

4. YAZILIM MIMARISI

4.1 Arka Plan

Bu bölüm, kayıtları insan hareketleriyle eşleştiren makine öğrenimi tabanlı sistem için yazılım geliştirmeden geçmektedir. Bilgisayarlara açıkça programlanmadan hareket etmeyi öğretme bilimi, makine öğrenimi olarak bilinmektedir. Makine öğrenimi, son birkaç on yılda bize kendi kendini süren bir araba, etkili web araması, konuşma tanıma ve insan genomunun önemli ölçüde daha iyi anlaşılmasını sağlamıştır. Makine öğrenimi artık o kadar yaygın ki, insanlar her gün fark etmeden yüzlerce kez onunla etkileşime girebilmektedir.

İnsanlar internette arama yapmak için Google veya Bing'i kullandıklarında, bu kadar etkili çalışmasının nedenlerinden biri, bir öğrenme algoritmasının web sitelerini nasıl sıralayacağını öğrenmiş olmasıdır. Kullanıcılar artık bir öğrenme algoritması sayesinde gelen kutularını her kontrol ettiklerinde bir sürü spam e-postayı araştırmak zorunda değiller. Makine öğrenimi nedeniyle, Facebook veya Apple'ın resim yazma aracı yaygın olarak kullanılmakta ve arkadaşların fotoğraflarını tanımlamaktadır. Sonuç olarak, makine öğrenimi bilgisayarlar için yeni bir yetenek olarak yaratılmış ve artık hem çok çeşitli endüstrileri hem de bilim alanlarını etkilir.

Bu çalışmada kullanılan yazılım, sistemi çalıştırmak için herhangi bir uygulama yüklemeye gerek kalmadan Windows işletim sistemi üzerinde herhangi bir tarayıcı ile çalışacak şekilde tasarlanmıştır, makine öğrenmesinin ana çekirdeği Django ortamında Doğal Dil İşleme ve web konuşma API'si olacaktır.

4.2 Yazılım Sistem Bileşenleri

1. Üzerinde python yüklü windows işletim sistemi

2. Pycharm Entegre geliştirme ortamı (IDE)

3.Django çerçevesi

4.SQL işlemçi

5.Krom tarayıcı

4.3 Python'da Doğal Dil İşleme

İnsan tarafından okunabilen metin içeren yapılandırılmamış veriler, çalışılabilen verilerin büyük bir bölümünü oluşturmaktadır. Programlı olarak analiz edebilmemiz için önce verileri önceden işlememiz gerekmektedir.

Doğal dil işleme (NLP), doğal insan dilini bilgisayar algoritmalarıyla anlaşılır ve işlenebilir hale getirmeye odaklanan bir makine öğrenimi (AI) dalıdır. Teknoloji açısından, NLP'nin birincil amacı, bilgisayarları büyük miktarda doğal dil verisini analiz edecek ve yorumlayacak şekilde programlamaktır. NLTK veya Natural Language Toolkit, NLP'ye dayalı bir Python paketidir ve bu tezin temel aldığı temel pakettir.

Genel olarak bir makinenin dilimizi anlaması kolay değil, ancak NLP yardımıyla bu başarılabilir. Daha da NLP'nin nasıl çalıştığının bir özeti aşağıdadır:

Konuşmayı kayıt ettiği bilgisayara girdi olarak gönderilir.

NLP sistemi, konuşmanın bağlamını ve kişinin niyetini anlamak için metni temel bölümlerine ayırır.

4.3.1 NLP Uygulamaları:

Makine Çevirisi.

Konuşma Tanıma.

Duygu Analizi.

Soru Cevaplama.

Metnin Özetlenmesi.

Sohbet robotu.

Akıllı Sistemler.

Metin Sınıflandırmaları.

Karakter Tanıma.

Yazım Denetimi.

İstenmeyen Posta Algılama.

Otomatik tamamlama.

Adlandırılmış Varlık Tanıma.

Tahmini Yazma.

4.3.2 Kural tabanlı NLP ve İstatistiksel NLP:

Doğal Dil İşleme iki kategoriye ayrılır:

4.3.2.1 **Kural tabanlı NLP:**

İşleme görevleri için sağduyulu düşünmeyi kullanmaktadır. Örnek olarak, sağduyulu düşünmenin diğer örnekleri arasında aşırı soğuk ölüme neden olabilmekte ve sıcak kahve insanların cildini yakabilmektedir. Ancak bu işlem uzun zaman alabilmekte ve el becerisi gerektirmektedir.

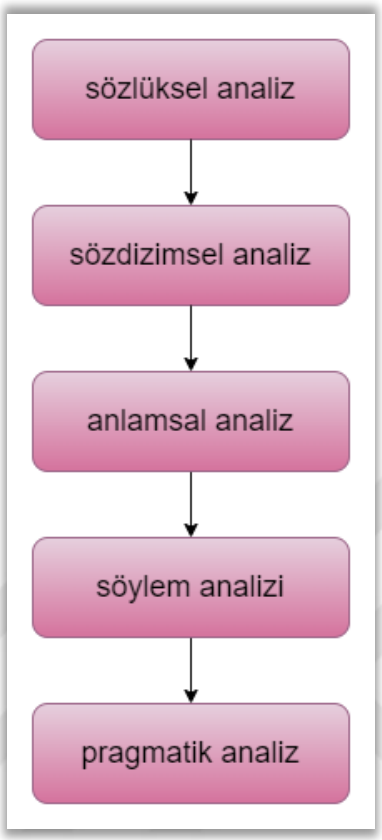
4.3.2.2 **İstatistiksel NLP:**

Sonuç çıkarmak için büyük miktarda veri kullanmaktadır. NLP modellerini istatistiksel NLP'de eğitmek için makine öğrenme teknikleri kullanılmaktadır. Eğitilen model, önemli miktarda veri üzerinde başarılı bir eğitimden sonra mükemmel kesinti sonuçları üretecektir.

Tablo 0.1: Kural Tabanlı NLP ve İstatistiksel NLP:

	Kural Tabanlı NLP	**İstatistiksel NLP**
Artıları	esnek	ölçeklenmesi kolay
	hata ayıklamak kolay	kendi kendine öğren
	fazla eğitim gerektirmez	hızlı gelişme
	dili anlamak	yüksek kapsama
	yüksek doğruluk	
Eksileri	yetenekli geliştirici gerektirir	büyük miktarda veri gerektirir
	yavaş ayrıştırma	hata ayıklamak zor
	orta kapsam	yüksek kapsama

4.3.3 Doğal Dil İşlemenin Bileşenleri:



Şekil 0.8: Doğal Dil İşlemenin (NLP) Bileşenleri.

a. Sözcük Analizi:

Sözcük analizini kullanarak büyük bir metin yığınını paragraflara, ifadelere ve kelimelere ayırırız. Kelimelerin yapısını tanımayı ve analiz etmeyi gerektirir.

b. Sözdizimsel Analiz:

Sözdizimsel analiz, bir cümledeki kelimelerin gramerini analiz etmek ve aralarındaki ilişkiyi gösterecek şekilde düzenlemek anlamına gelmektedir. Örneğin, "Dükkan eve gider" satırı geçmezdir.

c. Semantik Analiz:

Semantik analiz, kelimelerin kesin anlamını belirler ve metnin anlamını değerlendirmektedir. "Sıcak dondurma" gibi cümleler kabul edilemezdir.

d. Açıklama Entegrasyonu:

Açıklama entegre edilirken metnin bağlamı dikkate alınmaktadır. Bitirmeden önce cümlenin anlamını değerlendirir. Örnek olarak, "Google'da çalışıyor". Bu cümleden önceki cümlede "o" kelimesi geçmelidir.

e. Pragmatik Analiz:

Pragmatik analiz, genel dil iletişimi ve yorumu ile ilgilidir. Çeşitli durumlarda kelimelerin uygun kullanımını belirlemekle ilgilidir.

4.3.4 NLP kütüphaneleri:

a. Doğal Dil Araç Seti (NLTK):

NLTK Python çerçevesi çoğunlukla eğitim ve araştırma için kullanılmaktadır. Üretim ortamlarında yaygın olarak kullanılmazdır. Ancak basitliği nedeniyle ilginç programlar geliştirmek için kullanılabilmektedir.

NLTK Özellikleri arasında Tokenizasyon, Konuşma Konuşma bölümleri (POS) etiketleme, Adlandırılmış Varlık Tanıma (NER) ve sınıflandırma bulunmaktadır. NLTK Duygu analizi, chatbot paketlerine bağlıdır.

Kullanım durumları, Öneri sistemleri, Duyarlılık analizi ve Sohbet robotları oluşturmadır.

Tablo 0.2: NLTK çerçevesini kullanmanın artıları ve eksileri.

Artıları	**Eksileri**
en iyi bilinen ve tamamen NLP	öğrenmesi ve kullanması zor
en fazla sayıda dili destekleyin	kelimenin bağlamı göz ardı edilir
	yavaş
	sinir ağı modu yok

b. SpaCy:

spaCy, doğal dil işleme için hızlı ve üretime hazır bir açık kaynaklı Python modülüdür. spaCy, üretime hazır yazılım sağlamaya odaklanmaktadır.

spaCy özellikleri Tokenizasyon, Konuşma Parçası etiketleme (POS), Adlandırılmış Varlık Tanıma (NER), Sınıflandırma, Duygu analizi, Bağımlılık ayrıştırma ve Kelime vektörleridir.

Kullanım durumları Otomatik tamamlama ve otomatik düzeltme, İncelemeleri analiz etme ve Özetleme'dir.

Tablo 0.3: spaCy çerçevesinin artıları ve eksileri.

Artıları	Eksileri
hızlı	lehçe esnek
Öğrenmesi ve kullanması kolay	
eğitim için sinir ağını kullanın	

c. Gensim:

Gensim, konu modelleme ve benzerlik tanımlaması için Python tabanlı bir NLP çerçevesidir. Genel amaçlı bir NLP kütüphanesi değildir, ancak kendisine verilen işlerde mükemmeldir.

Özellikleri Gizli anlamsal analiz, Negatif olmayan matris çarpanlara ayırma ve TF-IDF'dir.

Kullanım durumları, Belgeleri vektörlere dönüştürme, Metin benzerliğini bulma ve Metin özetlemedir.

Tablo 0.4: Gensim çerçevesinin artıları ve eksileri.

Artıları	Eksileri
sezgisel arayüz	denetimsiz metin modelleri için tasarlanmıştır
ölçeklenebilir	diğer kütüphanelerle kullanılmalıdır
uygulanan algoritmalar	

d. Model:

Model, basit bir sözdizimine sahip bir Python NLP çerçevesidir. Hem bilimsel hem de bilimsel olmayan uygulamalar için kullanılabilecek çok yönlü bir araçtır. Öğrenenler için gerçekten faydalı.

Özellikleri arasında Tokenizasyon, Konuşmanın Parçası etiketleme, adlandırılmış varlık tanıma, Ayrıştırma ve Duygu analizi bulunmaktadır.

Kullanım durumları Yazım düzeltme, Arama motoru optimizasyonu ve Duygu analizidir.

Tablo 0.5: Model çerçevesinin artıları ve eksileri.

Artıları	Eksileri
veri madenciliği	belirli NLP görevi ile optimize edilmemiş
ağ analizi ve görselleştirme	

e. TextBlob (**MetinBloğu):**

TextBlob, metinsel verilerle uğraşmak için bir Python kütüphanesidir.

Özellikleri arasında Konuşma Bölümü etiketleme, İsim tümcesi çıkarma, Duygu analizi, Sınıflandırma, Dil çevirisi, Ayrıştırma ve Wordnet entegrasyonu bulunmaktadır.

Kullanım durumları Duygu Analizi, Yazım Düzeltme ve Çeviri ve Dil Tespitidir.

Tablo 0.6: TextBlob kitaplığının artıları ve eksileri.

Artıları	Eksileri
kullanımı kolay	yavaş
NLTK için sezgisel arayüz	sinir ağı modeli yok
dil çevirisi ve algılama sağlar	entegre kelime vektörleri yok

Bu proje için, diğer kütüphanelere kıyasla mükemmelliği nedeniyle NLTK kütüphanesinin kullanılmasına karar verilmiştir

4.3.5 Doğal Dil Araç Seti

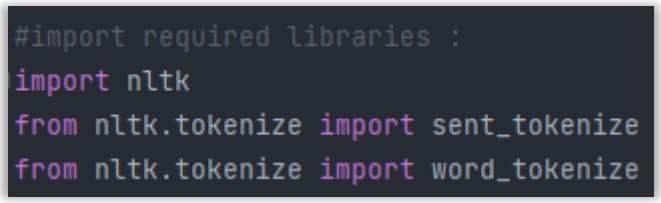
NLTK, insan dili verileriyle çalışmak için python tabanlı bir betik dilidir. Sınıflandırma, kaynak oluşturma, belirteçleştirme, etiketleme, ayrıştırma ve anlamsal akıl yürütme için bir dizi metin işleme kitaplığı, endüstriyel güçlü NLP kitaplıkları için paketler ve aktif bir tartışma panosu yanı sıra WordNet gibi 50'den fazla şirket ve sözlük kaynağına kullanımı kolay API'ler içermektedir.

NLTK, programlama ilkelerinin yanı sıra hesaplamalı dilbilimdeki konuların yanı sıra kapsamlı API belgelerini tanıtan uygulamalı bir kılavuz sayesinde dilbilimciler, mühendisler, öğrenciler, eğitimciler, akademisyenler ve endüstri kullanıcıları için idealdir. Windows, Mac

OS X ve Linux için NLTK mevcuttur. Hepsinden iyisi, NLTK topluluk odaklı, ücretsiz, açık kaynaklı bir projedir [18].

4.3.6 NLTK'nın Uygulanması:

NLP'deki çeşitli veri işleme senaryoları için bazı kitaplıkların içe aktarılması gerekmektedir. Bu bağlamda Doğal Dil İşleme için NLTK kullanılacaktır. Metni çeşitli şekillerde işlemek için kullanılacaktır.



Şekil 0.9: Gerekli kitaplıkları pycharm'da içe aktarma.

NLTK'da birçok özellik vardır ve aşağıda tezde kullanılanlar bulunmaktadır.

4.3.6.1 **Tokenleştirme**

Metni, simgeleştirerek kolayca kelime veya cümleye bölenebilir. Bu, bağlam dışına alındığında bile oldukça mantıklı ve anlamlı olan daha küçük metin parçalarıyla çalışmamızı sağlayacaktır. Yapılandırılmamış verileri daha kolay analiz edilebilen yapılandırılmış verilere dönüştürmeye yönelik ilk adımdır.

Metni analiz ederken, kelimeleri kelime belirteci ile ve cümleleri cümle belirteci ile belirtilecektir. Her iki tokenleştirme biçiminin faydaları şunlardır:

Kelime ile simgeleştirme: Doğal dilin atomları kelimelerdir. Hâlâ kendi başına anlaşılabilen en küçük anlam birimidirler. Metni kelime kelime belirtmek, sık görünen kelimeleri tespit etmemizi sağlar. Örnek olarak bir dizi iş ilanına bakacak olursak, "Python" kelimesinin sık sık geçtiğini fark edilebilir. Bu, Python bilgisi için önemli bir talebi gösterebilir, ancak kesin olarak öğrenmek için biraz daha gitmemiz gerekir.

Cümleye göre simgeleştirme: Bu kelimelerin birbirleriyle nasıl ilişkili olduğunu analiz edebilir ve cümle ile simgeleştirdiğimizde daha fazla bağlam görenebilir. İşe alım müdürü Python'u sevmediği için "Python" kelimesi bir sürü kötü kelimeyle mi çevrili? Yazılım geliştirme alanından ziyade herpetoloji alanından, beklediğimizden tamamen farklı bir Python türü ile çalışıyor olabileceği gösteren daha fazla sayıda ifade var mı?

Kelime ve cümle ile belirteç oluşturmak için NLTK'nın gerekli bileşenlerini aşağıdaki gibi içe aktarılır:

a. Cümle belirleme:

Metni sent_tokenize() ile simgeleştirerek, metni cümle olarak elde edebiliriz.

```
from nltk.tokenize import sent_tokenize

text = "Hello everyone. Welcome to STSLS. i'm explaining Sentence tokenizing"
sent_tokenize(text)
print(sent_tokenize(text))
```

```
['Hello everyone.', 'Welcome to STSLS.', "i'm explaining Sentence tokenizing"]
```

Şekil 0.10: Metni cümle olarak belirtmek için sent_tokenize( ) kullanımı.

Şekil 4.3, sent_tokenize'ın nasıl çalıştığını gösterir; sent_tokenize işlevi, daha önce eğitilmiş olan nltk.tokenize.punkt modülünden bir PunktSentenceTokenizer örneğini alır ve böylece bir cümlenin sonunu ve başlangıcını belirtmek için hangi karakterlerin ve noktalama işaretlerinin kullanılacağını tam olarak bilir. Çalışacak büyük miktarda veriye sahip olduğumuzda, bunu yapmak daha verimlidir.

İngilizce dışında, farklı turşu dosyaları kullanarak birkaç dilden cümleleri belirtebiliriz.

```
import nltk.data
turkish_tokenizer = nltk.data.load('nltk:tokenizers/punkt/turkish.pickle')
text = 'text = merhaba arkadaşlar. kolay gelsin.'
turkish_tokenizer.tokenize(text)
print(turkish_tokenizer.tokenize(text))
```

```
['merhaba arkadaşlar.', 'kolay gelsin.']
```

Şekil 0.11: diğer dillerden cümleleri tokenize edin

b. Kelime belirleme:

Metni word_tokenize() ile simgeleştirerek, metin kelimeler olarak temsil edilebilir.

```
from nltk.tokenize import word_tokenize

text = "Hello everyone. Welcome to STSLS. i'm explaining word tokenizing"
word_tokenize(text)
print(word_tokenize(text))
```

```
['Hello', 'everyone', '.', 'Welcome', 'to', 'STSLS',
'.', 'i', "'m", 'explaining', 'Sentence', 'tokenizing']
```

Şekil 0.12: Metni kelimeler olarak belirtmek için word_tokenize() kullanımı.

Şekil 4.5, word_tokenize'ın nasıl çalıştığını gösterir; tokenize() işlev sözcüğü, bir TreebankWordTokenizer örneğinde tokenize() öğesini çağıran bir sarmalayıcı işlevidir. ve ayrıca, verilerin tam metninin kelimelerle temsil edildiğini fark edilebilir.

c. Durdurma Sözlerini Filtreleme

Durdurma sözcükleri, okunmasını istemediğimiz ifadelerdir, bu nedenle işlenirken bunları metnimizden filtrenir. 'in', 'is' ve 'an' gibi durdurma sözcükleri, tek başlarına bir metne fazla bir anlam ifade etmedikleri için, sık sık durak sözcükleri olarak kullanılır.

Durdurma sözcüklerini filtrelemek için NLTK'nın gerekli bölümünü aşağıdaki gibi içe aktarılır

```
from nltk.corpus import stopwords
#list of stopwords:
stopwords = stopwords.words("english")
# print(stopwords)
print(text)
```

Şekil 0.13: Durdurulan kelimeler listesini içe aktarma.

```
for w in (words,tagged):
    if w not in stop_words:
        clean_words.append(w)
```

Şekil 0.14: Durdurulan kelimeleri kaldırma.

4.3.6.2 **Köklenme:**

İngilizce'de ve diğer birçok dilde, sözcükleri normalleştirmek için kök çıkarma kullanılır, çünkü tek bir sözcük bağlama bağlı olarak birçok farklı biçim alabilir. Bağlama bağlı olarak, "çalışma" kelimesinin "çalışmalar", "çalışıyor", "çalışılmış" ve diğerleri dahil olmak üzere birkaç farklı biçimi olabilir. Kelimeleri simgeleştirildiğinde, temel anlamları aynı olmasına rağmen, bir tercüman bunları ayrı kelimeler olarak ele alır. Ayrıca, NLP bilginin anlamını anlamakla ilgili olduğu için, bu sorunun üstesinden gelmek için köklendirmeyi kullanılır.

Belirli bir kelimeyi kök kelimesine keserek, kökten ayırma, onu normalleştirilir. Örnek olarak, "çalışmalar", "çalışıyor", "çalışılmış" kelimelerinin tümü, tek bir simgeye atıfta bulunarak "çalışma" olarak basitleştirilecektir.

Bu tezde sistemin geliştirilmesinde kullanılmadığı için buharlamanın uygulanması ve özellikleri buraya dahil edilmemiştir; bunun yerine Lemmatizasyon adlı karşılaştırılabilir bir özellik kullanılmıştır.

4.3.6.3 **Lemmatizasyon:**

Lemmatization, bir kelimenin temel "kökünü" benzer şekilde elde etmeye çalışır. Onu ayıran şey, orijinali kesmek yerine sözlük kelimesini aramasıdır. Köklendirilirken kelimenin bağlamı ihmal edilir. Sonuç olarak, lemmatizasyondan daha az doğru olmasına rağmen sonuçları daha hızlı üretir.

Doğruluk projenin ana amacı değilse, kaynak yapmak iyi bir seçenektir. Doğruluk kritikse, Lemmatizasyon en iyi çözümdür (Lemmatizasyon, köklenme ile karşılaştırıldığında daha düşük bir işlem hızına sahiptir).

Lemmatizasyon, Part Of Speech (POS) değerlerini hesaba katar. Ayrıca, lemmatizasyon, farklı POS değerleri için farklı çıktılar üretebilir. POS için genellikle dört seçeneğimiz vardır.

Lemmatizatörün Python Uygulaması:

a. Bir lemmatizatörün nasıl çalıştığını gösteren temel bir örnek

Aşağıdaki örnekte, "fiil" PoS etiketini kullanacağız ve orijinal kelimeyi kısaltmak yerine, lemmatizasyon kurallarını uyguladığımızda sözlük kelimeleri alınır:

```
from nltk import WordNetLemmatizer

lemma = WordNetLemmatizer()
word_list = ["Study","Studying","Studies","Studied"]

for w in word_list:
    print(lemma.lemmatize(w ,pos="v"))
```

```
Study
Studying
Studies
Studied
```

Şekil 0.15: NLTK çerçevesi ile basit lemmatizasyon örneği.

b. Varsayılan PoS değerine sahip lemmatizatör

Lemmatizasyonda, PoS'nin varsayılan değeri bir isimdir (n). Aşağıdaki örnekte sözlük kelimeleri oluşturduğunu görenebilir:

```
from nltk import WordNetLemmatizer

lemma = WordNetLemmatizer()
word_list = ["studies","leaves","decreases","plays"]

for w in word_list:
    print(lemma.lemmatize(w))
```

```
study
leaf
decrease
play
```

:

Şekil 0.16: Varsayılan değerler oluşturmak için lemmatizasyonu kullanma.

c. Lemmatizatörün gücünü gösteren başka bir örnek

```
from nltk import WordNetLemmatizer

lemma = WordNetLemmatizer()
word_list = ["am","is","are","was","were"]

for w in word_list:
    print(lemma.lemmatize(w ,pos="v"))
```

```
be
be
be
be
be
```

Şekil 0.17: “am”, “are”, “is”, “was”, “were” kelimelerinin lemmatizasyonu

d. Farklı POS değerlerine sahip olan lemmatizatör

```
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()
print(lemmatizer.lemmatize('studying', pos="v"))
print(lemmatizer.lemmatize('studying', pos="n"))
print(lemmatizer.lemmatize('studying', pos="a"))
print(lemmatizer.lemmatize('studying', pos="r"))
```

```
study
studying
studying
studying
```

Şekil 0.18: Şekil 17: Farklı Konuşma Bölümü değerleriyle lemmatizasyon.

4.3.6.4 **PoS Etiketleme**

Konuşma bölümlerinin etiketlenmesi (PoS), sözdizimsel ve anlamsal analiz için önemlidir. Sonuç olarak, "(can you help me with the can?) can konusunda bana yardım eder misin?" gibi cümlelerde. "can" kelimesinin çeşitli anlamsal anlamları vardır. İlk "can" soru yapmak için kullanılır. Şişe, cümlenin sonundaki ikinci "can" ile temsil edilir. İlk "can" bir fiile atıfta bulunurken, ikincisi bir isme atıfta bulunur. Belirli bir anlam verirseniz, bilgisayar kelimeyi hem anlamsal hem de sözdizimsel analizde doğru şekilde işleyecektir.

Aşağıdaki tablo, ilgili örnekleriyle birlikte Konuşma Bölümü (PoS) etiketlerinin bir listesini gösterir:

Tablo 0.7: Konuşma bölümü etiketlerinin listesi

Part-of-speech tag	Description
UNKNOWN	bilinmeyen kelime
DT	belirleyici
QT	niceleyici
CD	asıl sayı
NN	isim, tekil
NNS	isim, çoğul
NNP	Özel isim, tekil
NNPS	Özel isim, çoğul
EX	Varoluşsal bir parti vardı cümlesindeki gibi.
PRP	Kişi zamiri (PP)
PRP$	İyelik zamiri (PP$)
POS	iyelik bitişi
RBS	zarf, üstün
RBR	zarf, karşılaştırmalı
RB	Zarf
JJS	sıfat, üstün
JJR	sıfat, karşılaştırmalı
JJ	Sıfat
MD	modeli
VB	Fiil, temel form
VBP	Fiil, şimdiki zaman, üçüncü tekil şahıs
VBZ	Fiil, geçmiş zaman
VBD	Fiil, geçmiş ortaç
VBN	Fiil, gerund veya mevcut ortaç
WDT	Wh-belirleyici, örnek olarak, cümlede hangi kitabı daha çok beğeniyorsunuz?
WP	Wh-zamiri, örnek olarak, hangi ve o, göreli zamirler olarak kullanıldığında

WP$	Kim gibi iyelik zamiri
WRB	Wh-zarf, cümledeki gibi, benim için yemek hazırladığında hoşuma gidiyor
TO	edat
IN	Edat veya bağımlı bağlaç
CC	Koordinasyon bağlantısı
UH	ünlem
RP	parçacık
SYM	sembol
$	para birimi işareti
"	Çift veya tek tırnak işareti
(	Parantez, köşeli ayraç, köşeli ayraç veya köşeli ayraç açma
)	Parantez, köşeli ayraç, köşeli ayraç veya köşeli ayraç kapatma
,	Virgül
.	Cümle sonu noktalama işaretleri (. ! ?)
:	Cümle ortasında noktalama (: ; ... -- -)

PoS'nin Python Uygulaması:

PoS etiketlemeyi gösteren basit bir örnek.

```
#PoS tagging :
tag = nltk.pos_tag(["Studying","Study"])
print (tag)
```

```
[('Studying', 'VBG'), ('Study', 'NN')]
```

Şekil 0.19: PoS etiketleme örneği.

PoS etiketlemenin kullanımını gösteren tam bir örnek.

```
#PoS tagging example :

sentence = "A very beautiful young lady is walking on the beach"

#Tokenizing words :
tokenized_words = word_tokenize(sentence)

for words in tokenized_words:
    tagged_words = nltk.pos_tag(tokenized_words)

tagged_words
```

```
[('A', 'DT'),
 ('very', 'RB'),
 ('beautiful', 'JJ'),
 ('young', 'JJ'),
 ('lady', 'NN'),
 ('is', 'VBZ'),
 ('walking', 'VBG'),
 ('on', 'IN'),
 ('the', 'DT'),
 ('beach', 'NN')]
```

Şekil 0.20: PoS etiketlemesini gösteren tam Python örneği.

4.4 Django çerçevesi

Bir çerçeve, birçok farklı bileşen ve paketten oluşur. Ayrıca, karmaşık bir web uygulaması geliştirdiğimiz için zamandan tasarruf etmek için önceden bazı hazırlıklar yapılmalıdır.

Django, model-görünüm-şablon (MVT) mimari modelini izleyen Python tabanlı ücretsiz ve açık kaynaklı bir web çerçevesidir. Django Yazılım Vakfı (DSF) bunu sürdürür.

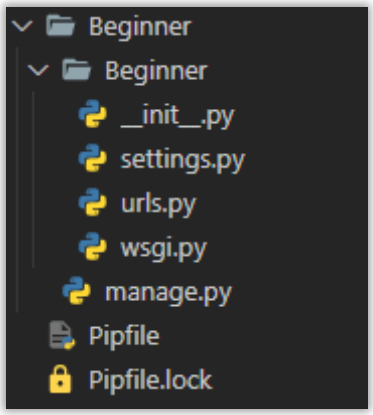
Django'nun temel amacı, karmaşık, veritabanına dayalı web siteleri oluşturmayı kolaylaştırmaktır.

Public Broadcasting Service, Mozilla, Instagram, The Washington Times, Bitbucket, Disqus ve Nextdoor, Django kullanan tanınmış sitelerden sadece birkaçıdır.

Her web sitesinin bir ön ucu ve bir arka ucu vardır. Web sitesinin ön ucu HTML, CSS ve JavaScript kullanılarak oluşturulacaktır. Temel olarak, sayfayı tasarlamak için HTML kullanılacak ve tasarımı geliştirmek ve standartlaştırmak için CSS kullanılacaktır. Web sitesini etkileşimli hale getirmek için JavaScript kullanılacaktır.

Python'da Django, web sitesinin dinamik bölümü ve web sitesinin arka ucunda gerçekleştirilecek belirli işlemler için ana anahtardır.

Python Manage.py STSLS komutunu kullanarak Django'da yeni bir uygulama oluşturduktan sonra aşağıdaki pencere açılacaktır



Şekil 0.21: pycharm'da Django ortamı

__init .py: Bu dosya yorumlayıcıya projenin Python ile yazıldığını bildirir.

settings.py: SECRET KEY, ALLOWED HOSTS, INSTALLED APPS, MIDDLEWARES, 'STATIC FILES CONFIG've diğer proje yapılandırmalarının tümü bu dosyada bulunur.

urls.py: Kök düzeyindeki URL ayarları bu dosyada bulunur. Projeyi çalıştırıldığında bu dosyadaki tüm URL'ler incelenip eşleştirilir.

wsgi.py: Bu dosya, dağıtım sonrasında web sunucuları için projenin giriş noktası görevi görür. Bu dosya, web sunucusu ve Django projesi arasında bir bağlantı kurar.

Manage.py: Kök dizini bu python dosyasını içerir. Bu, Django-admin ile aynıdır. Django yöneticisine benzer şekilde çalışır, ancak yalnızca mevcut proje için. Bu komut ile python yönetmeye benzer şekilde tüm projeyi yönetebiliriz. py run server - Bu dosyayı, sunucu çalıştır, süper kullanıcı oluştur vb. komutları ileterek çalıştırılabilir.

Django kullanmanın ana nedenleri şunlardır:

Hızlıdır: Bu, basitçe, eğer bir şey inşa etmek istiyorsak, konfigürasyonlar üzerinde fazla zaman harcamamamız gerektiği anlamına gelir, böylece Django, uygulamayı daha hızlı inşa etmemize yardımcı olacaktır.

Bileşenler: Django'da paketlenmiş olarak gelen ve diğer çerçevelerde sunulmayan bir avantaj olan bileşenlerin sayısı

Güvenlik: Django, kullanıcı verileri için yüksek düzeyde güvenlik sağlar

Ölçeklenebilirlik: Django, birden çok kullanıcıya ve daha fazla özelliğe sahip olma seçeneği sunar

4.4.1 Django'nun Mimarisi

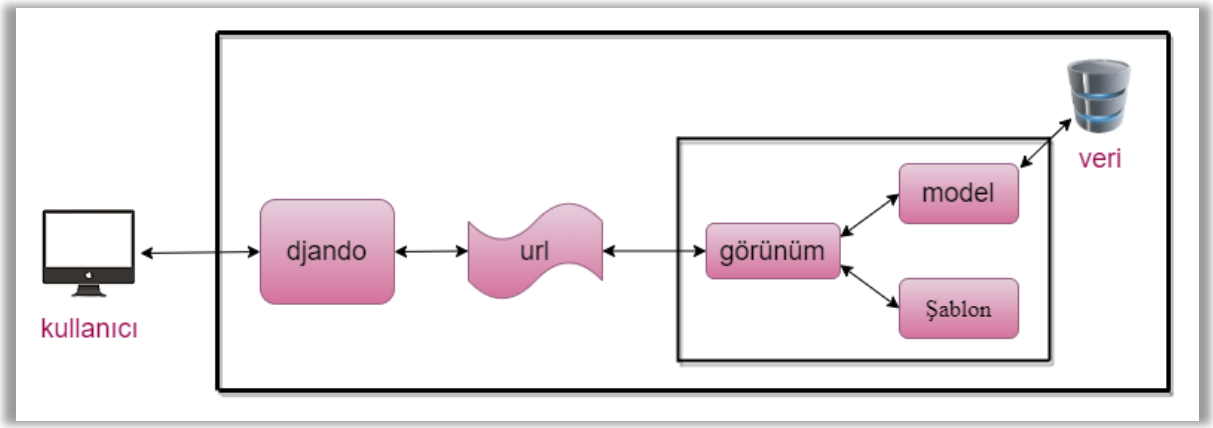
Her web uygulamasında, (MVC) model-görünüm-denetleyici adı verilen şeye sahibi, veri için bir modeli olduğu için endişelerle işbirliği yapmak için, ekranda görülen HTML formatı için bir görüşüsü var ve sonra bu işlemleri ve hepsini kontrol edecek bir kontrolör, iyi bir web uygulaması oluşturmaya yardımcı olacaktır.

Django, aynı MVC mimarisini takip eden ancak bazı değişikliklerle birlikte mimari (MVT) model-görünüm-şablonunu takip eder.

Model: Bu, mimarinin veritabanı katmanıdır. Veritabanındaki tablolara model adı verilir. Django, SQL kullanmak zorunda kalmadan Python'da veritabanı komutları oluşturmamızı sağlayan ORM (Object Relational Mapping) Protokolünü takip eder.

Şablon: Bu, mimarinin görüntüleme katmanıdır. Çevrimiçi arayüzümüzde oluşturulan (görüntülenen) HTML dosyaları burada saklanır. Her görünüm (işlev/sınıf) tarafından bir şablon veya bir HTTP yanıtı gönderilir. Django şablon dili de şablona dahildir (DTL)

Görünüm: Bu, mimarinin mantıksal katmanıdır. Burası projemizin mantığını yazdığımız yer. Sitemizin farklı URL Modellerine (proje) eşlenen işlevler/sınıflar oluşturuyoruz. Bir Django projesinde görünümler, model ve şablonlar arasında köprü oluşturur.



Şekil 0.22: Django'da MTV çerçeve mimarisi

Bir kullanıcı, http://localhost:8000/login adresinde projenin spesifik URL'sine bir istek gönderecektir. Örnek olarak Django, bu URL'yi, onun için mantığı uygulayan görünüm (işlev) ile ilişkilendirir.

Görünüm, ağ modellerinden (veritabanı) verileri alır ve oluşturulan web arayüzüne bir şablon gönderir.

4.4.2 Django'daki Modeller ve Geçişler:

Veritabanı tabloları model olarak bilinir. Modeller için Django, ORM'yi (Nesne İlişkisel Eşleme) kullanır. Django'nun ORM'si, veritabanımızı sorgulamak ve yönetmek ve sonuçları Python tabanlı bir şekilde elde etmek için SQL yazmak için yalnızca Python tabanlı bir yöntemdir.

Her model, Django'dan devralınan bir alt sınıftır. db.models model grubu. Her Django Model özniteliği, veritabanı tablosunda bir sütun belirtir.

Bir model tanımladığımızda, bu yalnızca biz onu taşıyana kadar veritabanımızda henüz yansıtılmamış olan python kodudur. Sonuç olarak, geçiş yapmak için python Manage.py'yi çalıştırılmak gerekir.

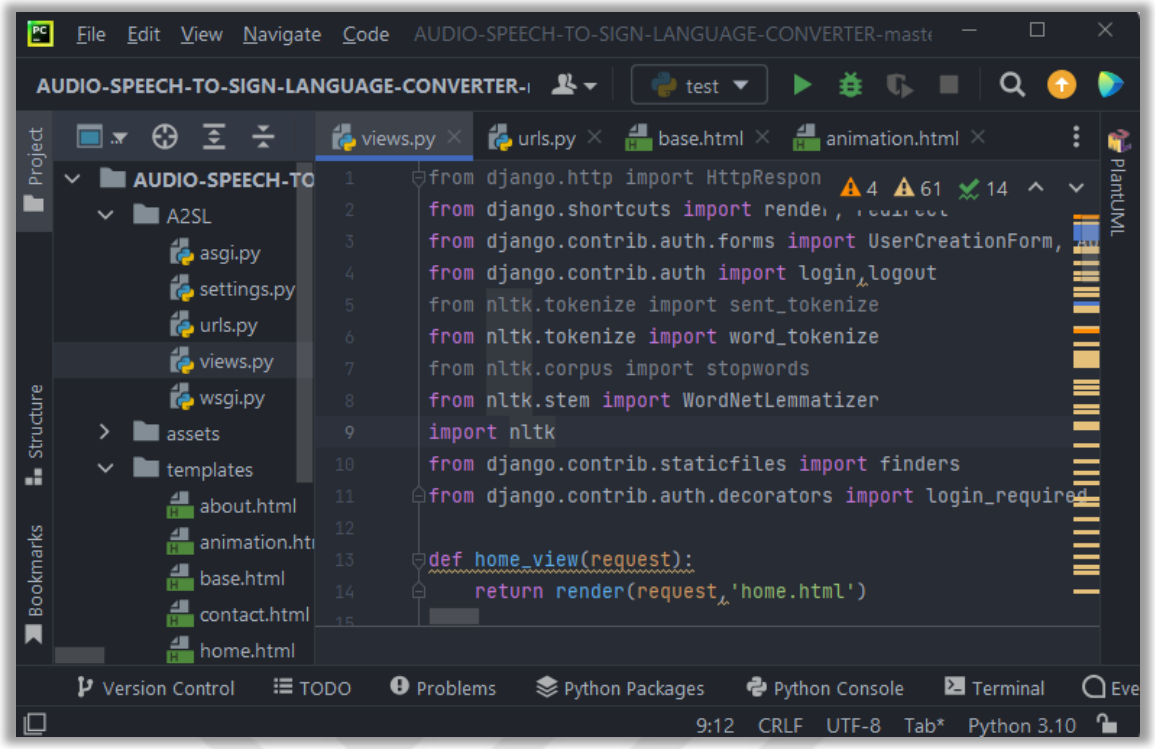
Modellerimizin tanımlandığı uygulama bu projenin kalbidir. Uygulama geçişleri klasörümde uygulama geçişleri adlı bir klasör oluşturur.

Her make geçişinde, veritabanına yansıtıldığı değişikliklerle ilgili bilgileri içeren 0001 initial.py adlı bir dosya oluşturulur.

Veritabanında veri oluşturulabilmesi ve değiştirilebilmesi için, geçişler yaptıktan sonra, bu değişiklikleri veritabanına fiilen yansıtılmak gerekir. Bu çalıştırma için Python Manage.py geçişi yapılır. En son geçişler yürütülecek ve değişiklikleri bu komut tarafından yansıtılacaktır.

4.5 Pycharm IDE

PyCharm, bilgisayar programlamada Python programlama dili için özel olarak kullanılan entegre bir geliştirme ortamıdır (IDE). Hem Django ile web geliştirmeyi hem de Anaconda ile veri bilimini destekler ve kod analizi, grafik hata ayıklayıcı, entegre birim test cihazı, sürüm kontrol sistemleri (VCS'ler) ile entegrasyon ve tüm bu özellikleri sunar.

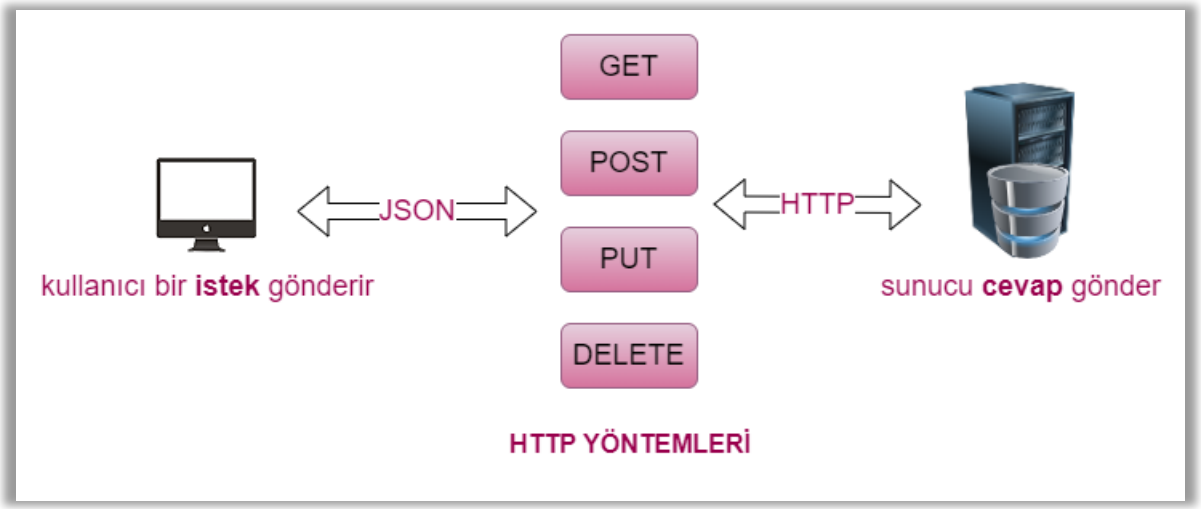


Şekil 0.23: Pycharm IDE

Sözlü mesajı gönderdiğimizde, istek işlemin gerçekleştiği sunucuya gönderilir ve sonucu alırız.

4.6 HTTP İstek Yöntemleri

Bir istemci/uygulama ve bir sunucu, HTTP isteklerini kullanarak birbirleriyle iletişim kurar. Sunucu, istemciden bir HTTP isteği alır ve bir yanıt döndürmeden önce onu işler. İsteğin durumu yanıta dahil edilecektir.



Şekil 0.24: HTTP istek yöntemleri

GET

GET yöntemi, belirtilen kaynağın bir temsilini ister. GET istekleri tarafından yalnızca belirli bir sunucu kaynağından alınan veriler alınmak gerekir.

XML, HTML, Metin ve JSON dahil olmak üzere verileri alabileceğimiz çeşitli formlar vardır ve genellikle sunucudan json verilerini almak için bir GET isteği kullanılır.

POST

POST, başka bir iyi bilinen HTTP istek tekniğidir. POST istekleri, kaynakları oluşturmak veya güncellemek için bir sunucuya veri iletmek için web iletişiminde kullanılır. Kullanıcı tarafından oluşturulan verileri bir sunucuya göndermek, sıklıkla HTTP POST tekniğinin kullanılmasını içerir. Örnek oalark, bir hareketin videosu yayınlandığında.

```
<form action="" method="post" align="left">
{% csrf_token %}
            <input type="text" name="sen" class="mytext" id="speechToText" placeholder="">
            <button type="button" name="button" class="mic" onclick="record()"><img src="{%
                
            <input type="submit" name="submit" class="submit">
```

Şekil 0.25: STSLS'de POST yönteminin kullanılması

Bu özel web sitesinde GET yöntemi kullanılabilir, ancak varsayılan olarak sunucuya veri gönderildiğinde adres çubuğunda güvenli olmayan bir şekilde görünebilir ve POST kullanacağımızdan kaçınmak için ikinci olarak web sitesi büyük miktarda ile uğraşacaktır. veri ve tarayıcının adres çubuğunda görünmesi iyi bir fikir değil.

Her birine belirli bir amaç atandığında başka çeşitli HTTP istek yöntemleri vardır ve bunlar:

PUT (hedef kaynağın tüm mevcut temsillerini istek yüküyle değiştirmek için)

DELETE (belirtilen kaynağı silmek için)

HEAD (bir GET isteğine benzer bir yanıt istemek için, ancak yanıt gövdesi olmadan)

CONNECT (hedef kaynak tarafından tanımlanan sunucuya bir tünel oluşturmak için)

OPTIONS (hedef kaynak için iletişim seçeneklerini açıklamak için)

TRACE (hedef kaynağın yolu boyunca mesajın geri döngü testi uygulamak için)

PATCH (bir kaynağa kısmi değişiklikler uygulamak için).

Web sitesi ön uç oluşturma dili

HTML: Köprü Metni İşaretleme Dili

4.7 Web sitesi ön uç oluşturma dilleri

4.7.1 HTML: HyperText Markup Language (Köprü Metni İşaretleme Dili)

Web'in en temel bileşeni HTML'dir (Köprü Metni İşaretleme Dili). Web içeriğinin amacını ve organizasyonunu açıklar. Bir web sayfasının görünümü/sunumu (CSS) ve işlevselliği/davranışı (JS) tipik olarak HTML (JavaScript) dışındaki teknolojiler kullanılarak tanımlanır. Tek bir web sitesi içinde veya web siteleri arasında çevrimiçi sayfaları bir araya getiren bağlantılara "hiper metin" denir. Web'in önemli bir bileşeni bağlantılardır. Çevrimiçi içerik yayınlayarak ve diğer kişilerin web sayfalarına bağlayarak World Wide Web'e aktif olarak katılabilirsiniz.

Bir Web tarayıcısında görüntülenmek üzere metne, resimlere ve diğer içeriğe açıklama eklemek için HTML, "markup." Special "elements" like head>, title>, body>, header>, footer>, article> ve diğerleri gibi özel "markup." HTML işaretlemesine dahil edilmiştir.

4.7.2 CSS: Cascading Style Sheets (Basamaklı Stil Sayfaları)

HTML veya XML'de oluşturulan bir sayfanın sunumu, Basamaklı Stil Sayfaları (CSS) olarak bilinen stil sayfası dili kullanılarak tanımlanır (SVG, MathML veya XHTML gibi XML lehçeleri dahil). CSS, konuşma, kağıt, ekranlar ve diğer ortamlar dahil olmak üzere çeşitli ortamlarda öğelerin nasıl gösterilmesi gerektiğini belirtir.

4.7.3 JavaScript (JS):

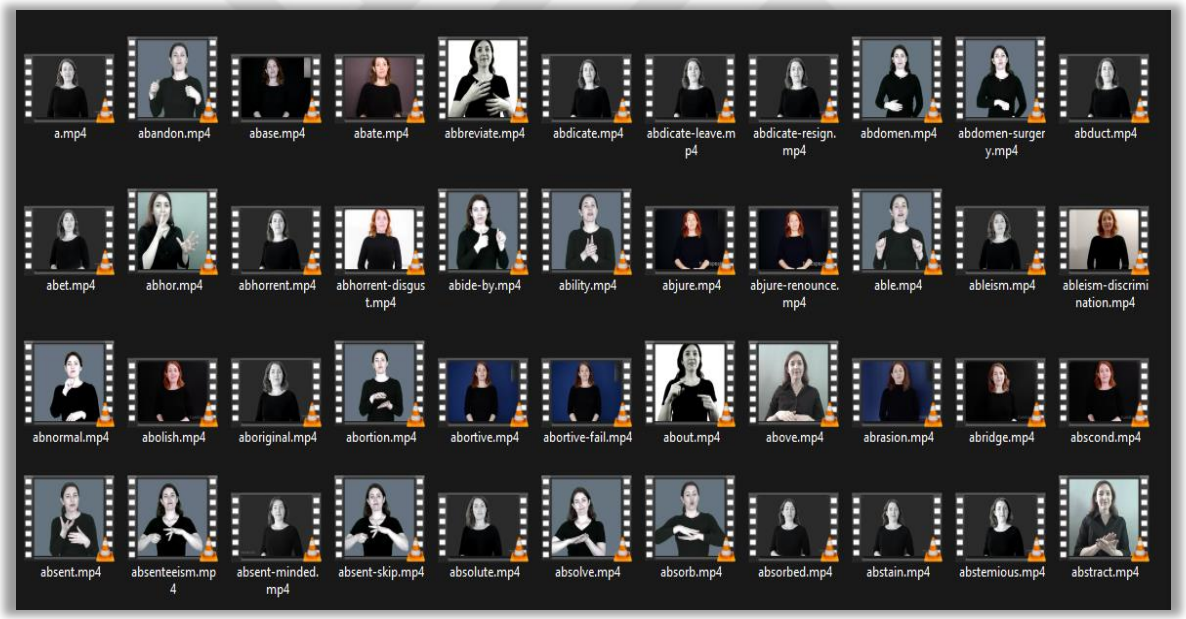
Birinci sınıf işlevlere sahip bir programlama dili olan JS, küçüktür, yorumlanır veya tam zamanında derlenir. Node.js, Apache CouchDB ve Adobe Acrobat gibi tarayıcı olmayan birçok ortam, Web sayfaları için en iyi komut dosyası dili olarak tanınmasına rağmen onu kullanır. JavaScript, nesne yönelimli, bildirimsel (işlevsel programlama gibi) ve zorunlu programlama paradigmalarını destekleyen dinamik, prototip tabanlı bir dildir.

4.8 Hareket Kitaplığı Geliştirme

Bu proje, yüz ifadelerine odaklanmaktadır, çünkü yüz ifadelerinin işaret dilinin önemli bir bölümünü ilettiği iyi bilinmektedir, bu nedenle işaretleri gerçekleştiren gerçek bir insanın, kelimenin veya cümle ile eşlenen sistemin son jest çıktısını temsil etmesine karar verilmiştir. işaret dilidir.

Bu insanlaştırılmış veri tabanı https://www.handspeak.com/ adresinden toplanmıştır. Üniversite öğrencileri ve öğrenciler, dil ve kültür meraklıları, tercümanlar, evde eğitim görenler, ebeveynler ve dil öğrenme, uygulama ve kendi kendine çalışma için dünyanın dört bir yanındaki profesyoneller için açık kaynaklı bir işaret dili ve Sağır kültür kaynağıdır.

Şekil. hareket kitaplığı



Şekil 0.26: ASL hareketleri kütüphanesi

5. TARTIŞMA VE SONUÇ

5.1 STSLS'de ASL çevirisinin gösterilmesi

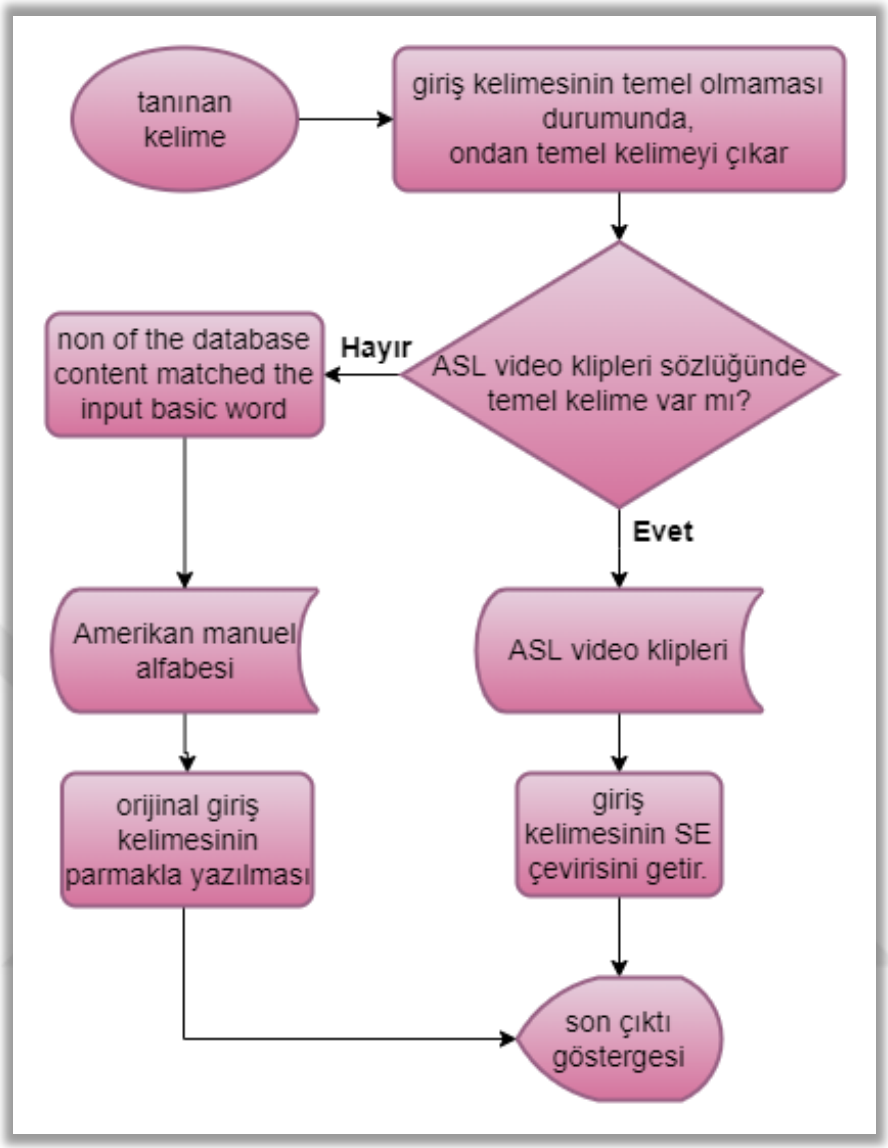
Yazılımımızdaki ASL'yi manipüle etmek için, [bölüm 2]'de açıklandığı gibi Signed English (SE) modelini takip edilmiştir. Basit bir şekilde, konuşma tanıma motorunun tanınan çıktı metni ASL veritabanına girilecektir. ASL veritabanı aşağıdakileri içerir:

_ Her bir video klibin aşağıdakileri temsil ettiği 14,200 ASL önceden kaydedilmiş video klip

İngilizce kelime dağarcığının tek bir temel kelimesinin karşılık gelen işareti.

Amerikan Manuel Alfabesi tamamen elle oluşturulmuştur. Şekil 25'da gösterildiği gibi, tanınan metni oluşturan varsa tek tek sözcükler ve alfabeler ayrı ayrı ele alınacaktır. Önce kelimenin temel kelime olup olmadığına karar verilecek; temel olmayan ifadeler zarfları, çoğulları vb. içerir. Ardından, kelime basitse, önce önceden kaydedilmiş ASL kısa kliplerinin veritabanıyla eşleştirilecektir. Bir eşleşme yapılmışsa, çıktı eşleşen klip olacaktır. Başka bir durumda, kelime Amerikan Manuel Alfabe veritabanı uygulanarak manuel olarak yazılacaktır. İkincisi, eğer terim temel değilse, temel kelime ondan çıkarılacak ve ASL veritabanının veritabanı ile karşılaştırılacaktır.

Bir eşdeğeri varsa, gösterilecektir, ancak İmzalı İngilizce kılavuz, bir önek veya bir sonek gibi uygun bir işaretleyici tarafından takip edilmesi gerektiğini belirtir. Basitçe söylemek gerekirse, bu temel kelime ASL veritabanı sözlüğünden çıkarılmışsa, çıkarmadan önceki orijinal kelime tamamen parmakla yazılacaktır. Aşağıdaki Şekil 26'da gösterilen nihai sonucun önceden kaydedilmiş video, işaretleyiciler ve parmakla yazımın bir kombinasyonunu içerebileceğini unutmamak gerekmektedir.



Şekil 0.27: STSLS'de ASL çevirisinin gösterimi

5.2 Doğruluk Ölçümleri

Kelime hata oranı (WER), model doğruluğunu belirlemek için endüstri standardıdır. WER, tanıma sırasında tanımlanan yanlış sözcüklerin sayısını sayar ve toplamı, insan etiketli transkriptte (N) sağlanan toplam sözcük sayısına böler [19].

Kelime hata oranı şu şekilde hesaplanabilir:

$$WER = \frac{S + D + I}{N} = \frac{S + D + I}{S + D + C}$$

Insertion (I): Ekleme; Hipotez transkriptine yanlış eklenen kelimeler

Deletion (D): Silme; Hipotez transkriptinde tespit edilmeyen kelimeler

Substitution (S): İkame; Referans ve hipotez arasında ikame edilen kelimeler

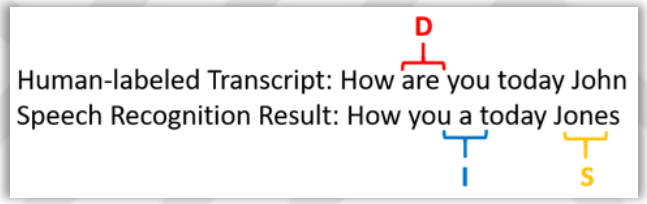
Correct words (C): Doğru sözcükler; Doğru sözcük sayısıdır.

N: referanstaki kelime sayısıdır (N=S+D+C)

ve amaç bir konuşma tanıma sisteminin performansını raporlamak olduğunda, bazen bunun yerine kelime doğruluğu (WAcc) kullanılır:

$$WAcc = 1 - WER = \frac{N - S - D - I}{N} = \frac{C - I}{N}$$

Aşağıda, insan etiketli transkript ile karşılaştırıldığında kelimelerin ne kadar yanlış tanımlandığını gösteren bir örnek:



Şekil 0.28: Yanlış tanımlanmış kelimelerin basit bir örneği.

Konuşma tanıma sonucu aşağıdaki gibi hata verdi:

Ekleme (I): "a" kelimesi eklendi

Silme (D): "are" kelimesi silindi

İkame (S): "John" yerine "Jones" kelimesi kullanılmıştır.

Bir önceki örnekteki kelime hata oranı %60'tır.

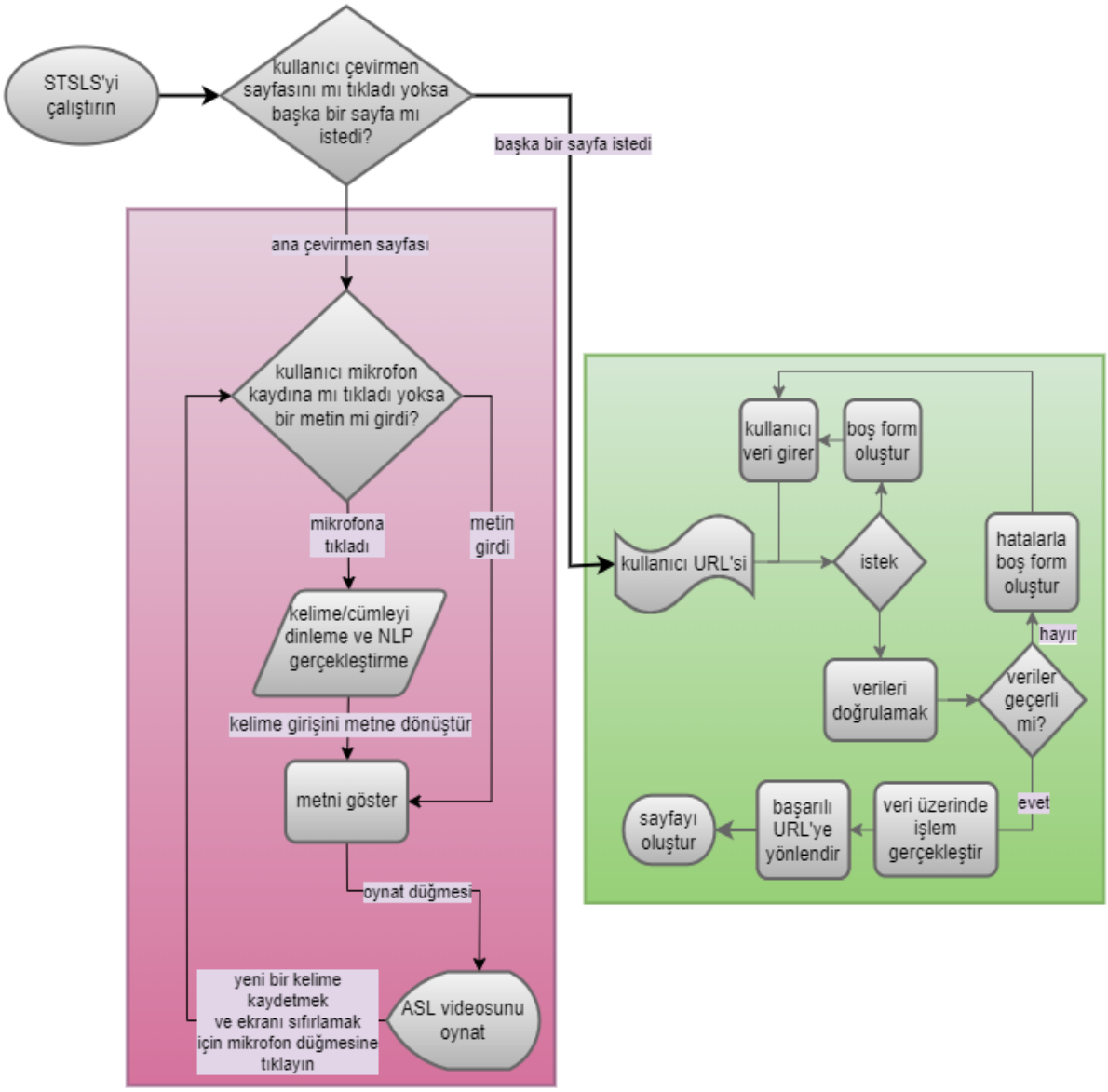
Yukarıdaki yöntemi kullanarak, dört adayın her biri farklı bir İngilizce dili telaffuz kullanarak ve alfabe/sayı/kelime dahil olmak üzere bir cümle denemesi için belirli bir konuşma parçası konuşarak 3 kere test edilmiştir

Tablo 0.8: Accuracy test of STSLS

Aday	Kelime /alfabe/sayı içeren bir cümle ile doğruluğu test etme I am test sample ”X” and I am “X” years old			Genel doğruluk
	1. kayıt doğruluğu	2. kayıt doğruluğu	3. kayıt doğruluğu	
A	%97	%98	%99	%98
B	%98	%99	%100	%99
C	%96	%97	%95	%96
D	%98	%99	%96	%97.6

Tablo 25.1'de gösterilen sonuçlara dayanarak, tüm testler için genel doğruluk, %96 ila %99 arasında değişen oldukça yüksek doğruluklardır. Her kullanıcı için test edilen denemelerde en az 3 deneme başarılı olmuştur. Web konuşma API'sinde işlem kolaylığı nedeniyle alfabe/sayı daha yüksek doğruluğa sahiptir. Bazı kelimelerin doğruluğu, telaffuz güçlükleri nedeniyle daha düşüktür, ancak anadili İngilizce olan bir kişi bununla ilgili bir sorunla karşılaşmaz ve bu nedenle doğruluk daha iyi olurdur.

Genel STSLS yapısının akış şemaları ve tüm ayrıntılı sistemin adım prosedürü aşağıdaki Şekil 27'de gösterilmektedir.



Şekil 0.29: Genel sistemin ön ucu, Pycharm'da Django kullanılarak tasarlanmıştır.

Sistem web sitesinin farklı seçeneklerinin GUI'si aşağıda gösterilmiştir:

5.3 Hesap Oluşturma Ekranı



Şekil 0.30: Hesap Oluşturma Ekranı

"Sign Up"a baslatıldığında bu ekran gelecektir. Burada, kullanıcı sadece STSLS hesabı için Adını, e-postasını ve şifresini girecek ve anında yeni bir hesap oluşturulacaktır. Her oturum başlangıcında kullanıcı, kullanıcı adını ve parolasını uygulamanın kendisinde saklamayı seçebilir ve bir sonraki seferde, kullanıcı aynı oturumu değiştirmediyse devam eder.

5.4 Oturum Açma Ekranı



Şekil 0.31: Oturum Açma Ekranı

STSLS'yi başlatıldığında görüneceği ilk ekran, oturum açma ekranıdır. Burada kullanıcı STSLS hesabına giriş yapabilir, yeni bir hesap oluşturabilir ve ardından sistemi doğru şekilde kullanmaya başlamak için Başlat düğmesine basabilir.

5.5 Ana STSLS Ekranı



Şekil 0.32: Ana STSLS Ekranı

Bu, ASL çeviri ana ekranıdır. Mikrofon simgesine basıldığında web sitesinin sol tarafı kayıt işlemini başlatır. Neyin çevrilmesini istediğimizi söyledikten sonra, web konuşma API yorumlayıcısı sesi bir dizeye dönüştürecek ve tamamen insanlaştırılmış bir karakter ifadeyi heceleyecektir. daha sonra çeviriyi tekrar oynatabilir veya mikrofon veya metin yazarak yeni kelimeler girilebilir.

6. SONUÇ VE ÖNERİLEN

6.1 Sonuç:

Bu tezde sunulan web tabanlı sistem, canlı modda sözlü İngilizce konuşmadan eşzamanlı olarak yazılı metin ve video imzalı İngilizce üretmiştir. Web ses API'sı, özünde konuşma tanıma motoru olarak kullanılmıştır. Tez, NLPTK parametrelerinin inşa ve işletim süreçlerini, DJANDO çerçevesinin geliştirilmesini, doğruluk testlerini ve STSLS'nin genel işleyişini tartışmıştır. TSLS tarafından gerçekleştirilen tüm prosedür, mikrofon kullanılırken başlayarak kısaca özetlenebilmiş, konuşma sinyali kaydedilmiş, dijital verilere dönüştürülmüş ve daha sonra her bir giriş cümlesi için belirli bir dizinin çıkarıldığı çevrimiçi web konuşma API işlemcisine uygulanmıştır. Çıktı daha sonra hareket kitaplığına karşı yerleştirilmiştir. Ayrıca motorun çıktı metninin video imzalı İngilizce'ye nasıl dönüştürüleceği de gösterilmiştir. Bu, ASL kılavuz alfabesi, işaretleyiciler ve önceden kaydedilmiş belirli sayıda ASL videosu STSLS işaret dili veritabanına dahil edilerek ve bunların hepsini İmzalı İngilizce (SE) kılavuzuna uygun olarak aynı anda uygulayarak gerçekleştirilmiştir. Her video, ASL'de bir temel kelimeyi temsil etmiştir.

Sistemin diğer yeteneklerinin başka gösterileri de varmıştır. Bunlar, çok sayıda kullanıcının oturum açma ve kapatma seçeneği, konuşmayı gerçek zamanlı olarak metne dönüştürme yeteneği, ASL filmler için ayrı bir kontrol paneli ve kelimeleri tanımak için kullanılan tekniklerin bir gösterimini içermiştit.

Doğruluk açısından, sistemin doğruluğunun makale boyunca gösterilmiştir.

Diğer benzer sistemlere kıyasla oldukça iyi görünmüştür. STSLS'nin araştırmaların dikkatini bu alana çekeceğini ve sağır ve sağır olmayan topluluklar arasındaki boşluğu doldurmaya yardımcı olacağını umuyoruz.

6.2 Öneriler

Django'dan web sitesi url'si terminal etki alanına kaydedilebilir, bu da Projeyi yerel veritabanından bulut veritabanına aktarmak anlamına gelir ve bu, her yerden her cihazdan erişim yeteneği verir.

Kelime kütüphanesi, İngilizce dilinde kullanılan diğer tüm kelimeleri, teknik kelimeleri ve diğer bilimsel alanlarda mevcut kelimeleri içerecek şekilde genişletilebilir.

Mevcut olan çeşitli NLP algoritmaları kullanılarak konuşmadan metne dönüştürme daha doğru hale getirilebilir ve metin işleme optimize edilebilir.

Web sitesi kullanıcı arayüzü daha da geliştirilebilir ve yeni işlevler eklenebilir.

Sistemi çapraz platformlar haline getirmek ve sistemin kullanılabilirliğini artırmak için kullanılabilecek net veya android uygulaması gibi çeşitli ön uç seçenekleri mevcuttur.

Kaynakça

[1] «Khalid El-Darymli, Othman O. Khalifa, and Hassan Enemosah, "Speech to Sign Language Interpreter System (SSLIS)", the IEEE International Conference of Computer and Communication Enginnering (ICCCE'06), in Proceedings of, Kuala Lumpur, Malaysia, 2006.».

[2] «V2S: Voice to Sign Language Translation System for Malaysian Deaf People Oi Mean Foong, Tang Jung Low, and Wai Wan La».

[3] «Elmahgiubi, M., Ennajar, M., Drawil, N., Elbuni, M.S.: Sign language translator and gesture recognition. In: Proceedings of the Global Summit on Computer & Information Technology. pp. 1-6 (2015).».

[4] «Zue, V., Cole, R., Ward, W.: Speech Recognition. Cambridge University Press, New York».

[5] «SIGN LANGUAGE CONVERTER by Taner Arsan and Oğuz Ülgen.».

[6] «Amit Kumar Shinde and Ramesh Khagalkar "sign language to text and vice versa recognization using computer vision in Marathi" International journal of computer Application (0975-8887) National conference on advanced on computing (NCAC 2015).».

[7] «Deaf Mute Communication Interpreter Anbarasi Rajamohan, Hemavathy R., Dhanalakshmi M. (Department of B.M.E., Sri Sivasubramania Nadar College of Engineering, Chennai, Tamil Nadu, India.) International Journal of Scientific Engineering and Technology (ISSN».

[8] «Vajjarapu Lavanya, Akulapravin, And M.S., Madhan Mohan "Hand Gesture Recognition And Voice Conversion System Using Sign Language Transcription System" ISSN: 2230-7109 (Online) | ISSN: 2230-9543 (Print) IJECT Vol. 5, Oct - Dec 2014.».

[9] «Salian, S., Dokare, I., Serai, D., Suresh, A., Ganorkar, P.: Proposed system for sign language recognition. In: Proceedings of the International Conference on Computation of Power, Energy Information and Communication. pp. 058-062 (2017).».

[10] «Nikam, A.S., Ambekar, A.G., Sign language recognition using image-based hand gesture recognition techniques. In: Proceedings of the Online International Conference on Green Engineering and Technologies. pp. 1-5 (2016).».

[11] «Jin, C.M., Omar, Z., Jaward, M.H.: A mobile application of American sign language translation via image processing algorithms. In Proceedings of the IEEE Region 10 Symposium. pp. 104-109 (2016).».

[12] «Madhuri, Y., Anitha, G., Anburajan, M.: Vision-based sign language translation device. In: Proceedings of the International Conference on Information Communication and Embedded Systems. pp. 565-568 (2013).».

[13] «The World Federation of the Deaf.,» [Çevrimiçi]. Available: http://www.hearinglossweb.com/res/hlorg/wfd.htm.

[14] «Cornucopia of Disability Information: Disability Statistics,» [Çevrimiçi]. Available: http://codi.buffalo.edu/graph_based/demographics/statistics.htm.

[15] [Çevrimiçi]. Available: https://www.handspeak.com/learn/index.php?id=278 .

[16] [Çevrimiçi]. Available: https://www.handspeak.com/learn/index.php?id=361.

[17] «Duyusal Eldiven ve Sinir Ağı Kullanarak Amerikan İşaret Dili Tanıma kitabından (s.4) Firas Abdulrazzaq Raheem, Aralık 2019, Irak tarafın.».

[18] «Bird, Steven, Edward Loper and Ewan Klein (2009), Natural Language Processing with Python. O'Reilly Media Inc.».

[19] «Klakow, Dietrich; Jochen Peters (September 2002). "Testing the correlation of word error rate and perplexity". Speech Communication. 38 (1–2): 19–28. doi:10.1016/S0167-6393(01)00041-3. ISSN 0167-6393.».