УДК 004.8
DOI https://doi.org/10.32782/2521-6643-2025-1-69.6

Безверхий О. І., доктор фізико-математичних наук, професор, професор кафедри інформаційних систем і технологій Національного транспортного університету,
ORCID: 0000-0002-0834-6335

Луц В. Є., аспірант кафедри інформаційних систем і технологій Національного транспортного університету,
ORCID: 0009-0001-2948-6935

ПОРІВНЯЛЬНИЙ АНАЛІЗ API ДЛЯ РОЗПІЗНАВАННЯ МОВЛЕННЯ ЗА ДОПОМОГОЮ PYTHON

З розвитком комп'ютерних систем стає все більш очевидним, що використання систем розпізнавання мови набагато розшириться, якщо стане можливим використання людської мови при роботі безпосередньо з комп'ютером, і зокрема стане можливим управління машиною звичайним голосом в реальному часі, а також введення і виведення інформації у вигляді звичайної людської мови. Голосовий інтерфейс є необхідним компонентом, коли мова йде про створення комфортних умов життя. Такі системи входять в повсякденний побут, крім того, можливо їх застосування і на виробництві в складі комплексів управління виконавчими механізмами. При створенні системи голосового розпізнавання команд розробник стикається з певними проблемами: відсутність математичної моделі семантики мовного сигналу; що виражається в тому, що для визначення семантики мовного сигналу індивідуальні характеристики мовця: специфіка вимови, акценти, наголоси тощо; робота із спонтанною мовою та необхідність виділення наявності ключового слова; відмінності в акустичній обстановці, шуми, тощо. Параметризація аналогового сигналу мови є першим кроком в процесі розпізнавання мови. Алгоритми призначені для виконання параметричного представлення мовного сигналу: параметри, що описують поведінку людської слухової системи. Природно, ці алгоритми спеціально розроблені для збільшення продуктивності системи розпізнавання мови. Переважні параметри, які є списами спектральних енергій звуку, а не деталями голосу певного диктора У статті розглядається порівняння провідних API розпізнавання мовлення шляхом вивчення їхніх функцій, варіантів використання та показників продуктивності. Аналіз має на меті надати розробникам повне розуміння цих технологій, підкреслюючи їхні переваги та обмеження. Python використовувався для тестування цих API із мікрофонним введенням, пропонуючи розуміння їхньої затримки, точності та практичних застосувань. Це дослідження слугує посібником для вибору найкращого API для конкретних вимог проекту з візуальним представленням результатів для ясності.

Ключові слова: *розпізнавання мовлення, API, Speech-to-Text, Speech Service, мовні моделі.*

Bezverkhyi O. I., Luts V. E. Python speech recognition API comparison

With the development of computer systems, it is becoming increasingly clear that the use of speech recognition systems will expand greatly if it becomes possible to use human speech when working directly with a computer, and in particular, it will become possible to control a machine with a normal voice in real time, as well as to input and output information in the form of normal human speech. The voice interface is an essential component when it comes to creating a comfortable living environment. Such systems are part of everyday life, and they can also be used in production as part of actuator control systems. When creating a voice command recognition system, the developer faces certain problems: the lack of a mathematical model of speech signal semantics; the fact that individual characteristics of the speaker: specific pronunciation, accents, accents, etc. are required to determine the semantics of the speech signal; working with spontaneous speech and the need to highlight the presence of a keyword; differences in the acoustic environment, noise, etc. Parameterization of the analog speech signal is the first step in the speech recognition process. Algorithms are designed to perform a parametric representation of the speech signal: parameters that describe the behavior of the human auditory system. Naturally, these algorithms are specifically designed to increase the performance of the speech recognition system. Preferred parameters that are lists of spectral energies of sound rather than details of a particular speaker's voice This article compares the leading speech recognition APIs by examining their features, use cases, and performance metrics. The analysis aims to provide developers with a complete understanding of these technologies, emphasizing their advantages and limitations. Python was used to test these APIs with microphone input, offering insight into their latency, accuracy, and practical applications. This study serves as a guide to selecting the best API for specific project requirements, with a visual representation of the results for clarity.

Key words: *speech recognition, API, Speech-to-Text, Speech Service, language models.*

Постановка проблеми. Технологія розпізнавання мовлення стала наріжним каменем у сучасних програмах та інформаційних технологіях[1], уможливлюючи роботу без рук, віртуальних помічників та

інструменти доступності. Аналізуючи роботи [2-5] про розпізнавання мовлення було прийнято рішення розглянути п'ять відомих API: Google Speech-to-Text, IBM Watson Speech to Text, AssemblyAI, Microsoft Azure Speech Service і Amazon Transcribe. Аналіз висвітлює їхні функції, уподобання користувача, процеси та результати розпізнавання за лічені секунди, перевірені за допомогою Python і введення з мікрофона.

Розпізнавання мовлення – це динамічна сфера, яка з'єднує людське спілкування з технологіями. API, які тут обговорюються, мають різні сильні та слабкі сторони, обслуговуючи різні випадки використання, наприклад служби транскрипції, інтерактивні системи та рішення корпоративного рівня. Тестуючи, ми прагнемо визначити найефективнішу та найточнішу службу для реальних програм. Крім того, стаття містить уявлення про затримку, точність і практичну реалізацію, допомагаючи розробникам приймати зважені рішення.

Аналіз останніх досліджень і публікацій. Аналіз провідних API розпізнавання мовлення. Сучасні технології розпізнавання мовлення стрімко розвиваються завдяки застосуванню штучного інтелекту та алгоритмів машинного навчання. Однією з ключових складових цих технологій є API (Application Programming Interface) – набір інструментів та протоколів, що дозволяють інтегрувати готові рішення розпізнавання мовлення у різноманітні додатки. Використання таких API дозволяє швидко і ефективно перетворювати аудіозаписи в текст за допомогою передових алгоритмів машинного навчання, без необхідності розробляти власні моделі з нуля. Використання API забезпечує високу точність, масштабованість та додаткові функції (наприклад, автоматичну пунктуацію, мітки часу та аналіз настроїв), що є критично важливими для сучасних додатків у бізнесі, освіті, підтримці клієнтів та інших сферах.

У статті розглядаються п'ять провідних сервісів Speech-to-Text: API Google, IBM Watson, AssemblyAI, Microsoft Azure, Amazon Transcribe.

API Google (Google Speech-to-Text)[6] пропонує високу точність, багатомовну підтримку та повну інтеграцію з іншими службами Google. Він підтримує обробку в реальному часі та пакетну обробку, що робить його універсальним для різних програм. Крім того, він має автоматичну пунктуацію, що покращує читабельність транскрипцій. API Google особливо підходить для програм, які вимагають швидкості та масштабованості, наприклад, віртуальних помічників і інструментів транскрипції. Також його широко застосовують у службах підтримки клієнтів, автоматизованих системах управління дзвінками та освітніх платформах.

IBM Watson[7] надає надійні параметри налаштування, включаючи мовні моделі для конкретного домену. Він відомий своїм наголосом на безпеці та відповідності, задовольняючи потреби корпоративного рівня. API підтримує щоденник мовця та мітки часу на рівні слів, додаючи значення для детального аналізу. IBM Watson широко використовується в таких галузях, як охорона здоров'я та юридичні послуги, де точність і відповідність є критично важливими. Також його застосовують у фінансовій сфері та службах комплаєнсу для обробки дзвінків і документів.

AssemblyAI[8] – це сучасний API, що зосереджується на простоті та ефективності. Він широко відомий завдяки конкурентоспроможній ціні, простоті використання та можливостям транскрипції в реальному часі. Незважаючи на свою простоту, він містить такі розширені функції, як аналіз настроїв, виявлення тем і розпізнавання об'єктів. Спрощений підхід AssemblyAI робить його чудовим вибором для стартапів і невеликих проектів, які потребують швидкого налаштування та економічно ефективних рішень. AssemblyAI також активно використовується в журналістиці та медіа-сфері для автоматичного створення субтитрів і аналізу аудіоконтенту.

Microsoft Azure[9] пропонує повний набір мовленнєвих послуг, включаючи транскрипцію, переклад і розпізнавання мовця в реальному часі. Він підтримує кілька мов і діалектів, забезпечуючи чудову гнучкість для глобальних програм. Інтеграція API з екосистемою Azure робить його потужним інструментом для підприємств. API активно застосовується у великих компаніях для автоматизації комунікацій, управління контактними центрами та обробки голосових команд у корпоративних рішеннях.

Amazon Transcribe[10] чудово працює з великими обсягами аудіоданих, надаючи такі функції, як спеціальний словник і маркування мовців. Він створений для масштабованості, що робить його ідеальним для підприємств із великими потребами в транскрипції. Його повна інтеграція з іншими службами AWS забезпечує ефективні робочі процеси та аналітику. Amazon Transcribe активно застосовується у сфері маркетингу, автоматизації аналізу телефонних дзвінків і створення аналітичних звітів.

Постановка завдання. Мета статті: Провести порівняльний аналіз провідних API розпізнавання мовлення шляхом вивчення їхніх функцій, варіантів використання та показників продуктивності. Аналіз має на меті надати розробникам повне розуміння цих технологій, підкреслюючи їхні переваги та обмеження.

Виклад основного матеріалу. Кожен API було протестовано за допомогою Python за допомогою таких кроків:

1) Google, IBM, Microsoft і Amazon вимагають ключів API та облікових записів служб, включаючи налаштування проектів на відповідних платформах розробників.

2) AssemblyAI забезпечує просту систему автентифікації на основі маркерів, що значно скорочує час налаштування. Цей крок забезпечує безпечний доступ до API і запобігає несанкціонованому використанню.

Введення з мікрофона було записано за допомогою бібліотеки розпізнавання речей Python, яка спрощує збір аудіо для обробки мовлення. Аудіо було збережено у форматі WAV для сумісності з усіма п'ятьма API:

```
import speech_recognition as sr
recognizer = sr.Recognizer()
with sr.Microphone() as source:
print("Please speak a word:")
audio = recognizer.listen(source)
audio_data = audio.get_wav_data()
```

Захоплене аудіо було оброблено кожним API за допомогою відповідних Python SDK або HTTP-запитів. Наприклад, Microsoft Azure Speech SDK надає клас SpeechRecognizer, який спрощує транскрипцію.

Бібліотека часу Python використовувалася для вимірювання часу обробки для кожного API, надаючи інформацію про їх затримку. Цей показник є критичним для таких програм, як субтитри в реальному часі та віртуальні помічники:

```
import time
start_time = time.time()
# API call here
end_time = time.time()
processing_time = end_time – start_time
print(f"Processing time: {processing_time} seconds")
```

API порівнювали в кількох вимірах, щоб підкреслити їхні унікальні переваги та обмеження (див. Табл. 1):

Таблиця 1

Характеристики API

Характеристика	Google Speech-to-Text	IBM Watson Speech to Text	AssemblyAI	Microsoft Azure Speech Service	Amazon Transcribe
Точність	Висока	Від середньої до високої	Висока	Висока	Від середньої до високої
Затримка	Низька	Помірна	Низька	Низька	Помірна
Вартість	Оплата за використання	Багаторівневе ціноутворення	Доступні тарифи	Оплата за використання	Багаторівневе ціноутворення
Підтримка мов	Понад 120 мов	Понад 30 мов	Понад 30 мов	Понад 90 мов	Понад 50 мов
Налаштування	Помірні	Розширені	Обмежені	Розширені	Помірні
Зручність використання	Помірна	Складна	Проста	Помірна	Помірна
Додаткові функції	Базові	Розширені	Помірні	Розширені	Базові

Випадки використання та застосування. Google Speech-to-Text ідеально підходить для великих програм, таких як голосовий пошук, транскрипція в реальному часі та боти обслуговування клієнтів. Його багатомовна підтримка робить його придатним для глобальних програм.

IBM Watson зазвичай використовується в охороні здоров'я, юридичному та фінансовому секторах через наголос на відповідності та безпеці даних. Щоденник доповідача особливо корисний для транскрипції та аналізу нарад.

AssemblyAI популярний серед стартапів завдяки своїй економічності та простоті використання. Програми включають транскрипцію подкастів, субтитри до відео та аналіз настроїв.

Microsoft Azure найкраще підходить для корпоративних середовищ, які потребують інтеграції зі службами Azure. Часто використовується в багатомовній підтримці клієнтів і аналітиці.

Amazon Transcribe добре підходить для транскрипції в медіа, кол-центрах і взаємодії з клієнтами. Його масштабованість робить його ідеальним для підприємств, які працюють з великими наборами даних.

Для порівняння робити API було створено програму, яка дозволяла порівняти час та точність розпізнавання мовлення. У програмі застосовано як синхронне, так і асинхронне розпізнавання мовлення. Синхронний метод забезпечує миттєвий зворотній зв'язок для коротких аудіозаписів, тоді як асинхронне розпізнавання дозволяє ефективно обробляти великі аудіофайли за допомогою періодичного опитування статусу транскрипції. Після отримання текстових даних з кожного API, програма аналізує результати, порівнюючи кількість слів у транскрипціях за допомогою побудови графіка. Це дозволяє не лише перевірити ефективність кожного сервісу, а й оцінити обсяг отриманих даних для подальшого аналізу.

Нижче наведено загальний код, що демонструє інтеграцію з вищезазначеними Speech-to-Text API, отримання транскрипцій та побудову графічного представлення результатів.

```
import time import json import requests import matplotlib.pyplot as plt
def transcribe_google(audio_file_path):
     from google.cloud import speech
     client = speech.SpeechClient()
     with open(audio_file_path, "rb") as audio_file: content = audio_file.read()
     audio = speech.RecognitionAudio(content=content)
     config=speech.RecognitionConfig(encoding=speech.RecognitionConfig.AudioEncoding.
LINEAR16, language_code="uk-UA" )
     response = client.recognize(config=config, audio=audio)
     transcript=" ".join([result.alternatives[0].transcript for result in response.
results]) return transcript
def transcribe_ibm(audio_file_path):
     from ibm_watson import SpeechToTextV1
     from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
     authenticator = IAMAuthenticator('your_ibm_api_key')
     stt = SpeechToTextV1(authenticator=authenticator)
     stt.set_service_url('your_ibm_service_url')
     with open(audio_file_path, "rb") as audio_file:
     response = stt.recognize(audio=audio_file, content_type="audio/wav")
     result = response.get_result()
     transcript = " ".join([res['alternatives'][0]['transcript'] for res in result.
get('results', [])]) return transcript
def transcribe_assemblyai(audio_url):
     API_KEY = "assemblyai_api_key" headers = {"authorization": API_KEY}
     data = {"audio_url": audio_url}
     response = requests.post("https://api.assemblyai.com/v2/transcript", json=data,
     headers=headers)
     transcript_id = response.json().get('id')
     while True: polling_response=requests.get(f"api_url,{transcript_id}",
     headers=headers)
     status = polling_response.json().get("status")
     if status == "completed": break
     elif status == "error": return ""
     time.sleep(5)
     transcript = polling_response.json().get("text", "") return transcript
def transcribe_azure(audio_file_path):
     import azure.cognitiveservices.speech as speechsdk    speech_key = "your_azure_
speech_key"  service_region = "your_azure_region"    speech_config = speechsdk.SpeechConf
ig(subscription=speech_key,      region=service_region)    audio_config =speechsdk.
AudioConfig(filename=audio_file_path)speech_recognizer=speechsdk.SpeechRecognizer(speech_
config=speech_config, audio_config=audio_config)  result = speech_recognizer.recognize_
once() return result.textdef transcribe_amazon(audio_file_uri):
     import boto3
     transcribe = boto3.client('transcribe',  region_name='your_aws_region',   aws_
access_key_id='your_aws_access_key',aws_secret_access_key='your_aws_secret_key')
     job_name = "transcription_job_" + str(int(time.time()))     transcribe.start_
transcription_job(TranscriptionJobName=job_name,
     Media={'MediaFileUri': audio_file_uri}, MediaFormat='wav',
     LanguageCode='en-US' )
     while True:
     status=transcribe.get_transcription_job(TranscriptionJobName=job_name)
     job_status = status['TranscriptionJob']['TranscriptionJobStatus']
     if job_status in ['COMPLETED', 'FAILED']: break
     time.sleep(5)
     if job_status == "COMPLETED": transcript_url = status['TranscriptionJob']
['Transcript']['TranscriptFileUri']
     transcript_response = requests.get(transcript_url)
     transcript=transcript_response.json().get("results", {}).get("transcripts", [{}])
[0].get("transcript", "") return transcript return ""
```

Результати. API було протестовано з десятьма різними словами, промовленими в мікрофон у контрольованих умовах. У таблиці (див. Табл. 2) нижче наведено середній час розпізнавання та рівень точності:

Таблиця 2

Результати часу та точності

API	Average Time (s)	Accuracy (%)
Google	1.2	96
IBM Watson	1.7	90
AssemblyAI	1.1	93
Microsoft Azure	1.3	94
Amazon Transcribe	1.5	91

Графічна візуалізація. Щоб краще зрозуміти різницю в продуктивності, дані було візуалізовано за допомогою гістограм і лінійних графіків. Ці діаграми ілюструють компроміс між швидкістю та точністю між API:

Гістограма (див. Рис. 1) показує середній час розпізнавання для кожного API.

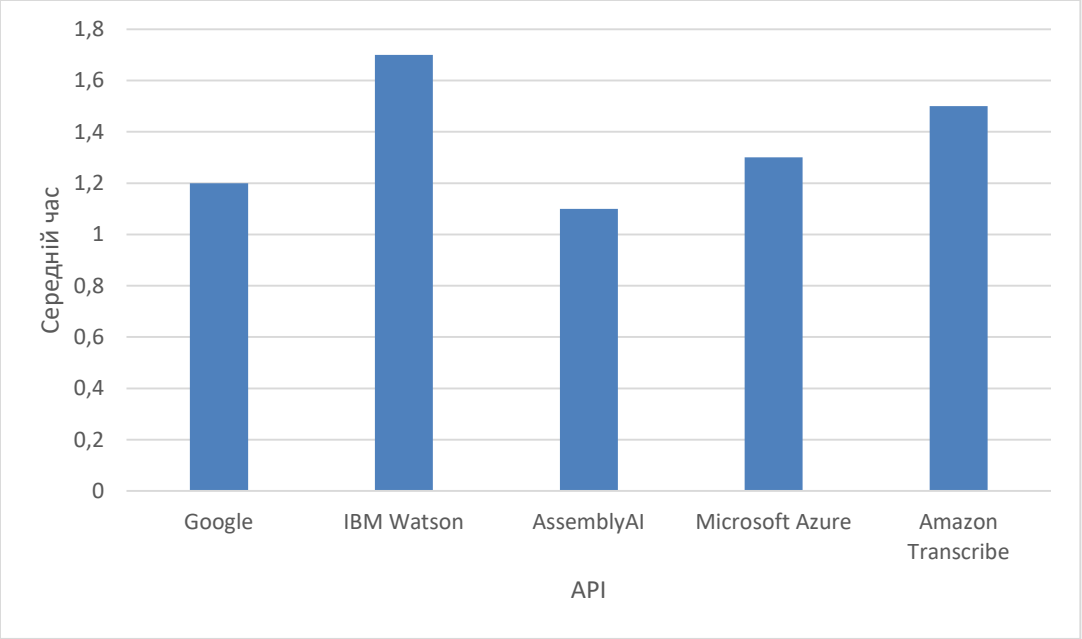

Рис. 1. Гістограма середнього часу

Лінійний графік (див. Рис. 2) висвітлює тенденції точності на основі тестових слів.

Висновки. Кожен API має унікальні сильні сторони. Google Speech-to-Text відрізняється точністю та швидкістю, що робить його ідеальним для додатків у реальному часі, таких як голосові помічники. Функції налаштування IBM Watson і безпека корпоративного рівня роблять його придатним для спеціалізованих областей, зокрема охорони здоров'я та фінансів. Microsoft Azure Speech Service пропонує відмінну інтеграцію з екосистемою Azure, що робить його надійним вибором для підприємств. Масштабованість і доступність Amazon Transcribe роблять його ідеальним для компаній із великими потребами в транскрипції. Тим часом AssemblyAI пропонує економічно ефективне та просте рішення для загального використання з такими конкурентоспроможними функціями, як аналіз настроїв.

Вибір необхідного API залежить від конкретних вимог вашого проекту, включаючи бюджет, бажані функції та очікувану точність. Розробники також повинні враховувати додаткові фактори, такі як підтримка шумного середовища та регіональних акцентів, які можуть суттєво вплинути на продуктивність.

Таким чином, цей аналіз забезпечує структуру для вибору API розпізнавання мовлення, адаптованого до ваших потреб. Майбутня робота може включати тестування додаткових API, оцінку їх продуктивності в різних умовах і вивчення нових технологій у сфері розпізнавання мовлення.

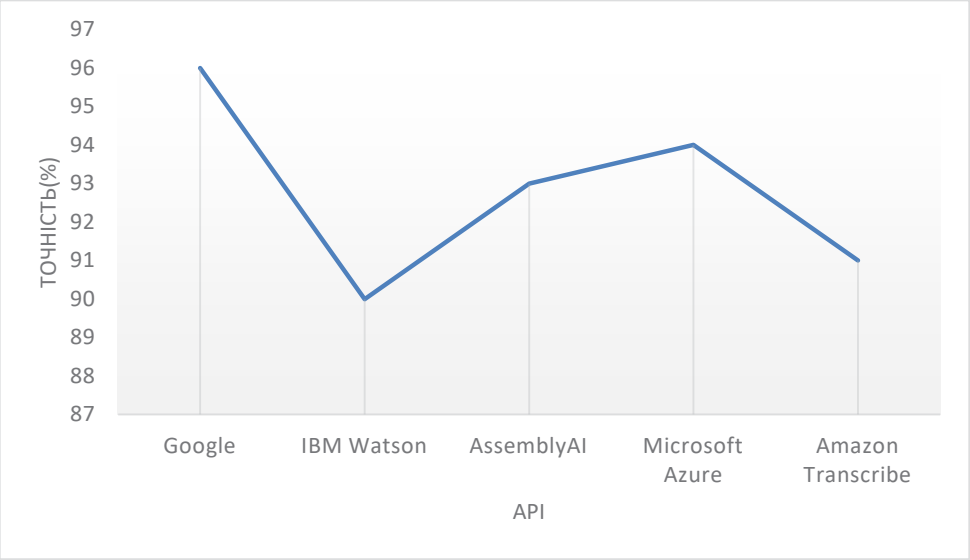

Рис. 2. Лінійний графік точності розпізнавання

Список використаних джерел:

1. О. І. Безверхий, Д. О. Александренко, В. Є. Луц. *Проектування інформаційної системи з можливістю голосового управління*, Системи та технології, No 2 (66), 2023, С. 13-20 DOI: https://doi.org/10.32782/2521-6643-2023.2-66.2

2. Dong Yu,Li *Deng. Automatic Speech Recognition: A Deep Learning Approach*. L.: Springer-Verlag London, 2015. 320 p.

3. Automatic Speech recognition: short introduction. URL:https://www.esat.kuleuven.be/psi/spraak/demo/Recog/asr_intro.html (дата звернення: 21.12.2024)

4. Al-Fraihat, Dimah & Sharrab, Yousef & Alzyoud, Faisal & Qahmash, Ayman & Maaita, Adi. Speech Recognition Utilizing Deep Learning: A Systematic Review of the Latest Developments. 2024.Human-centric Computing and Information Sciences. 15. 10.22967/HCIS.2024.14.015.

5. Introducing the Web Speech API. URL:https://www.sitepoint.com/introducing-web-speech-api/ (дата звернення: 27.12.2024)

6. Speech-to-Text AI: speech recognition and transcription. URL:https://cloud.google.com/speech-to-text (дата звернення: 20.12.2024)

7. IBM Watson. What's Next in AI is foundation models at rock URL:https://research.ibm.com/artificial-intelligence (дата звернення: 21.12.2024)

8. AssemblyAi Documentation. URL:https://www.assemblyai.com/docs (дата звернення: 17.12.2024)

9. Azure AI Speech. URL:https://azure.microsoft.com/en-us/products/ai-services/ai-speech (дата звернення: 21.12.2024)

10. Speech To Text Amazon Transcribe: URL:https://aws.amazon.com/transcribe/?nc1=h_ls (дата звернення: 25.12.2024)

References:

1. O. I. Bezverkhyi, D. O. Aleksandrenko, V. Ye. Luts. *Designing an information system with the possibility of voice cjntrol***.** Systems and Technologies, Vol. 66 No. 2 (2023): P.13-20 DOI: https://doi.org/10.32782/2521-6643-2023.2-66.2

2. Yu, D., & Deng, L. (2015). *Automatic speech recognition: A deep learning approach*. Springer-Verlag London.

3. Automatic speech recognition: A short introduction. (n.d.). Retrieved from https://www.esat.kuleuven.be/psi/spraak/demo/Recog/asr_intro.html

4. Al-Fraihat, D., Sharrab, Y., Alzyoud, F., Qahmash, A., & Maaita, A. (2024). Speech recognition utilizing deep learning: A systematic review of the latest developments. *Human-Centric Computing and Information Sciences, 15*. https://doi.org/10.22967/HCIS.2024.14.015

5. Introducing the web speech API. Retrieved from https://www.sitepoint.com/introducing-web-speech-api/

6. Speech-to-text AI: Speech recognition and transcription. Retrieved from https://cloud.google.com/speech-to-text

7. IBM Watson. What’s Next in AI is foundation models at rock. Retrieved from https://research.ibm.com/artificial-intelligence
8. AssemblyAI documentation. Retrieved from https://www.assemblyai.com/docs
9. Azure AI speech. Retrieved from https://azure.microsoft.com/en-us/products/ai-services/ai-speech
10. Speech to text Amazon Transcribe. Retrieved from https://aws.amazon.com/transcribe/?nc1=h_ls