DOI: https://doi.org/10.36910/6775-2524-0560-2023-53-26
УДК 004.415.3
Пех Петро Атонович, к.т.н., доцент
https://orcid.org/0000-0002-6327-3319
Бортник Катерина Яківна, к.т.н., доцент
https://orcid.org/0000-0001-5282-099X
Шепелюк Дмитро Леонідович, магістрант
Шепелюк Леонід Дмитрович, магістрант
Луцький національний технічний університет, м. Луцьк, Україна

ОБГРУНТУВАННЯ ВИБОРУ МОВНОЇ МОДЕЛІ ДЛЯ РОЗРОБЛЕННЯ ГОЛОСОВОГО АСИСТЕНТА МОБІЛЬНОГО ДОДАТКУ

Пех П.А., Бортник К.Я., Шепелюк Д.Л., Шепелюк Л.Д. Обґрунтування вибору мовної моделі для розроблення голосового асистента мобільного додатку. В статті запропоновано мобільний додаток на базі вибраної мовної моделі для голосового управління розумним домом і наведені результати досліджень якості розпізнавання голосу за допомогою цього додатку.
Ключові слова: Розумний дім, систем розпізнавання мовлення, голосовий асистент, нейронна мережа

Pekh P., Bortnyk K., Shepelyuk D., Shepelyuk L. Rationale for choosing a language model for the development of a mobile application's voice assistant. The article proposes a mobile application based on a neural network for voice control of a smart home and gives the results of research on the quality of voice recognition using this application.
Keywords: Smart home, speech recognition systems, voice assistant, neural network.

Постановка задачі. У теперішній час голосові асистенти стали невід'ємною частиною комп'ютерно-інформаційних технологій, пропонуючи широкий спектр функцій від голосового пошуку до управління розумним домом [1]. Однак існуючі голосові асистенти не є ідеальними, і їхні недоліки можуть негативно впливати на роботу пристроїв – у тому числі пристроїв розумного дому [2].

Серед найбільш популярних голосових асистентів на сьогоднішній день варто відзначити такі, як Google Assistant, Amazon Alexa, Apple Siri та Microsoft Cortana [3]. Ці голосові асистенти відомі своєю високою якістю розпізнавання мовлення та широким спектром функцій, однак вони мають і певні недоліки, а саме:

- Недостатня точність розпізнавання мовлення. Багато голосових асистентів, зокрема Google Assistant та Siri, можуть допускати помилки у розпізнаванні мовлення, особливо у разі наявності акцентів, специфічних особливостей мовлення або в умовах підвищеного шуму [3].
- Залежність від Інтернету. Amazon Alexa та Google Assistant, наприклад, вимагають для роботи стабільного підключення до Інтернету, що стає серйозною перешкодою в умовах нестабільного зв'язку або повільного Інтернету [3].
- Загрози конфіденційності. Усі популярні голосові асистенти зберігають та обробляють аудіодані користувачів на спеціальних серверах, що може породжувати питання щодо конфіденційності та безпеки особистої інформації [2].

У зв'язку з наведеним вище, вважаємо актуальним завданням створення мобільного додатку на базі нейронних мереж для голосового управління розумним домом, і проведення дослідження процесу, як саме додаток впливає на якість розпізнавання голосу.

Метою дослідження було створити мобільний додаток на базі нейронної мережі для голосового управління розумним домом і провести дослідження якості розпізнавання голосу на базі цього додатку.

Новизна дослідження полягає у тому, що для побудови мобільного додатку використовуються сучасні нейронні мережі, що, на нашу думку, певною мірою забезпечує ефективність його роботи.

Основна частина. Для управління пристроями розумного дому потрібно мати систему, яка складається з таких елементів:

- Голосові асистенти, такі як Amazon Alexa[5], Google Assistant[8] або Apple Siri[4], які можуть служити інтерфейсом для взаємодії та видачі команд пристроям розумного дому за допомогою голосу.
- Центральний контролер та мережу, які встановлюються у будинку і координують взаємодію між різними пристроями системи.

- Мобільний додаток для керування пристроями розумного дому віддалено, навіть якщо ви не знаходитесь вдома.

Якщо пристрої розташовані недалеко від приміщення (теплиця, басейн та інше), або знаходяться поза зоною досяжності Wi-Fi, можна використовувати Bluetooth для з'єднання та керування ними. Багато сучасних пристроїв розумного дому підтримують Bluetooth-протокол. Але для цього потрібно розробити мобільний додаток з голосовим асистентом для управління пристроями, який зможе працювати offline, використовуючи Bluetooth.

Завдання дослідження полягає в тому, щоб дослідити, які мовні моделі найкраще підходять для створення голосового асистента для управління пристроями розумного дому, розробити мобільний додаток з використанням різних мовних моделей та дослідити якість розпізнавання в режимі ofline.

Центральним елементом системи управління пристроями розумного дому є мобільний додаток. На рисунку 1 наведено спрощена функціональна схема системи голосового управління розумним домом.

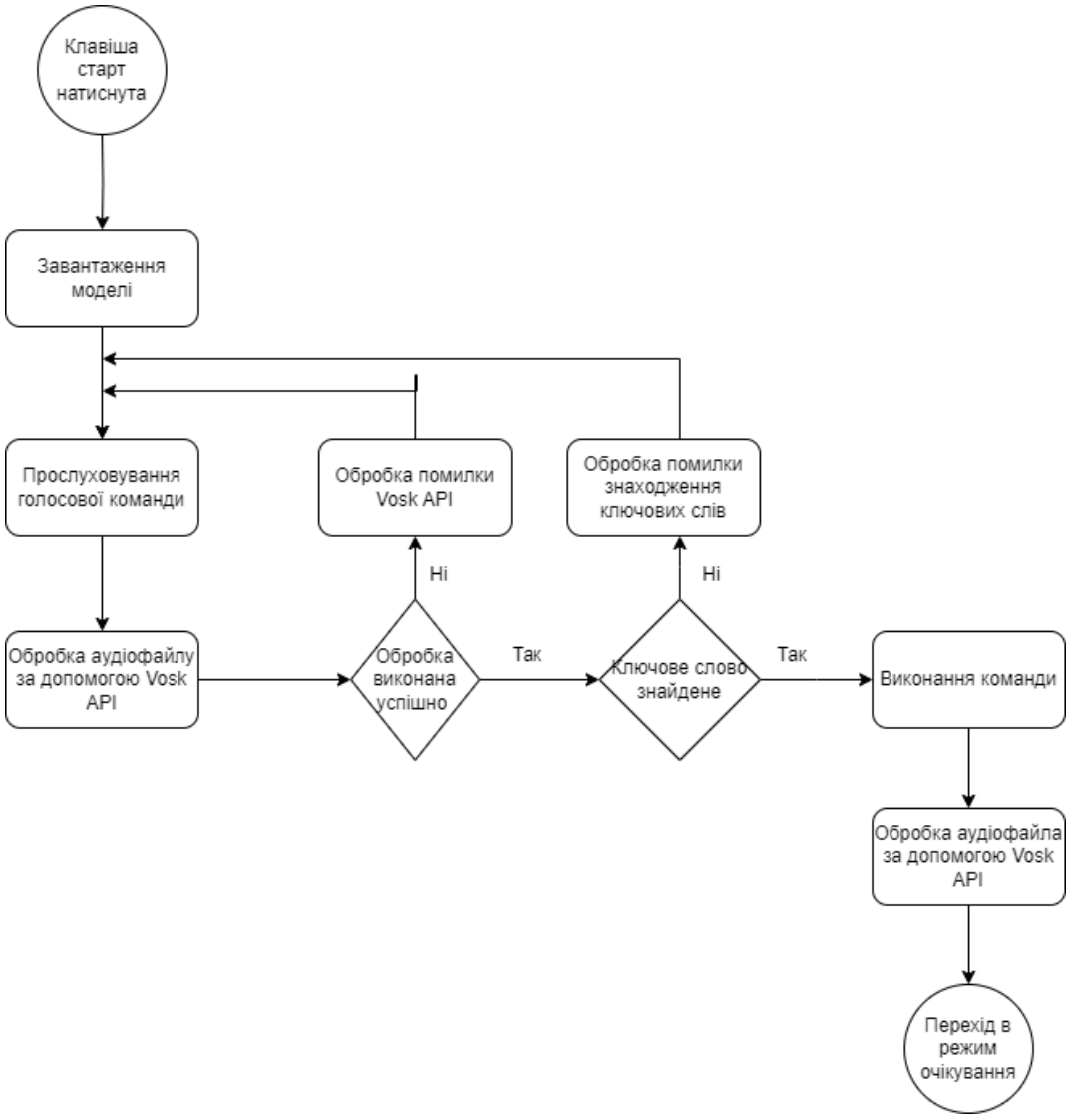

Рис.1. Спрощена функціональна схема системи голосового управління розумним домом

Аналіз та обгрунтування вибору мовної моделі для мобільного додатку. Основою системи розпізнавання голосових повідомлень є мовні моделі, розроблені із застосуванням згорткових або рекурентних нейронних мереж та машинного навчання. Наводимо деякі з них:

- DeepSpeech [6]. Відкрите програмне забезпечення для розпізнавання голосу від компанії Mozilla, використовує глибокі нейронні мережі (RNN) для розпізнавання мовлення. Модель DeepSpeech навчається за допомогою технік машинного навчання на великих обсягах аудіоданих.
- Wav2Vec 2.0 [10]. Модель для автоматичного розпізнавання мовлення, розроблена командою вчених від Facebook AI Research (FAIR). Ця модель використовує трансформери та призначена для ефективного розпізнавання мовлення на основі аудіоданих.
- Google Cloud Speech-to-Text [8]. Сервіс від Google для розпізнавання мовлення, яка надає API для використання в різних додатках.
- Kaldi [14]. Набір інструментів для розпізнавання мовлення, дозволяє робити розпізнавання мовлення для різних мов та в різних умовах. Це може бути використано для мов, які не є англійською, і для специфічних вимог в різних областях застосування. Однією з особливостей Kaldi є підтримка традиційних підходів до розпізнавання мовлення, а також впровадження техніки глибокого навчання для покращення якості розпізнавання.
- CMU Sphinx (PocketSphinx) [9]. Відкрите програмне забезпечення для розпізнавання мовлення від Carnegie Mellon University

Тестування проводилося на демо-версії розробника, або на основі рекомендованих шаблонів застосування.

Програма тестувалась чотирма користувачами. Для кожної моделі користувач промовляв вісім різних команди в умовах відносної тиші (за наявності незначного побутового шуму – музики, розмови, вуличного шуму), на відстанях: 30-40 см та 2 м.

Результати експериментальних досліджень (табл.2) були обчислені за формулою

$$K = \frac{\sum_j^m \sum_i^n a_{ij} * b_I}{k} * 100\% \quad (1)$$

де aij – j-та спроба і-го користувача;
bi – ідентифікатор розпізнавання;
bi = [1– вдале розпізнавання 0 – невдале;
m – кількість спроб ;
n – кількість користувачів;
k – кількість експериментів.

Таблиця 1 – Результати дослідження голосових моделей

Назва мовної моделі (сервісу)	Підтримка укр. мови	Вартість	Підтримка offline розпізнавання	Джерело Примітки	Якість розпізнавання % в умовах відносної тиші	
					Відстань 20 см	Відстань 200 см
1.vosk-model-uk-v3	так	Безоплатно	так	343Мб для серверних застосунків	96.13	90.24
2.vosk-model-small-uk-v3-nano	так	Безоплатно	так	73Мб для мобільних застосунків	96.35	91.46
3.Google S. R. - Android	так	Безоплатно	ні	вбудовано в смартфон	96.57	89.69

4.Google S. R. - Python	так	Безоплатно (60 хвилин)	ні	https://www.google.com/intl/en/chrome/demos/speech.html	95.32	90.54
5.Google Cloud Demo	–	0.96-2.16$	ні	Потребує платного аккаунту	не тестувалась	
6.Yandex SpeechKit Demo	–	платний	ні	-	не тестувалась	
7.Web Speech - Chrome PC	так	Безоплатно	ні	Демо: https://www.google.com/intl/en/chrome/demos/speech.html	94.76	88.0
8.Web Speech - Android WV	так	Безоплатно	так	https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API	96.71	90.24
9.Sphinx - Python	ні	Безоплатно	так	https://www.sphinx-doc.org/en/master/ англомовна підтримка	97.64 для англ.	91.17 для англ.
10.Amazon Transcribe	ні	1.44$	ні	https://docs.aws.amazon.com/transcribe/latest/dg/custom-language-models.html Відсутня демо-модель	Не тестувалась	
11.Facebook wav2vec-U	так	Безоплатно	ні	https://www.cockatoo.com/	98.1	92.23

За результатами дослідження голосових моделей було зроблено висновок, що для offline розпізнавання голосових команд найкраще підійде модель vosk-model-small-uk-v3-nano, яка має такі переваги:

– невеликий об'єм 73Мб, що забезпечує економне використання оперативної пам'яті, а це важливо для мобільних пристроїв з обмеженими ресурсами;

– можливість роботи в режимі offline, що дозволяє підвищити швидкість розпізнавання та немає недоліків пов'язаних з передачею даних;

– розпізнавання української мови на рівні кращих моделей.

Алгоритм розробки голосового ассистента.

Підготовка датасета. Датасет являє собою текстовий файл слідуючої структури (рис. 2). Файл може містити до 10 000 рядків. Його легко модифікувати вручну за допомогою текстового редактора, або вбудованою функцією.

```
#Набір ключових слів, що служать ключем для запуску голосового асистента
TRIGGERS = {'маруся','мару','муся' }
'''Тренувальна модель для нейронної мережі'''
data_set = {
#'фраза': 'ім'я_функції_щобуде_її_опрацьовувати, коментар_асистента',
'яка погода': 'weather зараз подивлюся',
'як там на вулиці': 'weather погода чудова',
'скільки градусів': 'weather подивися у вікно, я скажу що на сайті',
'запусти браузер': 'browser запускаю браузер',
'відкрий браузер': 'browser запускаю браузер',
'закрийся': 'offBot відключаюсь',
'відключись': 'offBot відключаюсь',
'як справи':'passive працюю, не переживай',
'що робиш':'passive жду команди',
'привіт':'passive і тобі не хворіти',
'ти тут':'passive тихенько працюю',
'}
```

Рис.2. Структура датасету

Навчання моделі.

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
import words #підготовлений датасет
```

Метод CountVectorizer з бібліотеки scikit-learn використовується для побудови векторів слів на основі текстових даних та класифікації цих даних за допомогою логістичної регресії (LogisticRegression). Алгоритм навчання наступний:

1. Створення об'єкту CountVectorizer, який буде відповідальний за токенізацію текстових даних та побудову векторів лічильників слів.

```
vectorizer = CountVectorizer()
```

2. Створення векторів слів із використанням методу fit_transform для отримання векторів слів з текстових даних, які містяться в словнику words.data_set.keys().

```
vectors = vectorizer.fit_transform(list(words.data_set.keys()))
```

3.Створення класифікатора логістичної регресії для класифікації текстових даних.

```
clf = LogisticRegression()
```

4. Навчання класифікатора. Використання методу fit для навчання логістичної регресії на отриманих векторах слів та відповідних класах, які містяться в dataset.

```
clf.fit(vectors, list(words.data_set.values()))
```

Аналізатор голосової команди. Функція розпізнавання отримує на вході датасет, підготовлений мовний вектор та натреновану модель виділяє з датасету ім'я функції та запускає опрацювання голосової команди.

```
from skills import *
import queue
q = queue.Queue()
device = sd.default.device  # мікрофон та динаміки за замовчуванням
samplerate = int(sd.query_devices(device[0], 'input')['default_samplerate'])
''' Аналіз розпізнаної команди    '''
def recognize(data, vectorizer, clf):
    # перевіряє звернення до бота
    trg = words.TRIGGERS.intersection(data.split())
    if not trg: return
    data.replace(list(trg)[0], '')
    # порівнює текстовий вектор з подібними варіантами
    text_vector = vectorizer.transform([data]).toarray()[0]
    answer = clf.predict([text_vector])[0]
    # визначає ім'я функції (команди) з dataset
    func_name = answer.split()[0]
    # озвучує коментар бота
    voice.speaker(answer.replace(func_name, ''))
    # запуск функции из власної бібліотеки skills
    exec(func_name + '()')
```

Рис. 3. Фрагмент коду розпізнавання голосової команди

Фонове прослуховування. Голосовий асистент працює в фоновому режимі. Отримавши від мікрофону аудіопотік опрацьовує його за допомогою мовної моделі та передає текст на опрацювання функції

```
with sd.RawInputStream(samplerate=samplerate, blocksize = 16000,
                       device=device[0], dtype='int16',
                       channels=1, callback=callback):
    rec = vosk.KaldiRecognizer(model, samplerate)
    while True:
        data = q.get()
        if rec.AcceptWaveform(data):
            data = json.loads(rec.Result())['text']
            c(data, vectorizer, clf) #
        # else: print(rec.PartialResult())
```

Рис. 4. Фрагмент коду для організації фонового прослуховування аудіопотоку та опрацювання голосових повідомлень

Тестування голосового асистента

Програма тестувалась за умовами що і голосові моделі. Результати експериментальних досліджень були обчислені за формулою 1

Таблиця 2. Результати тестування

Відстань, см	Якість розпізнавання, % в умовах тиші	Відсоток розпізнавання,% за наявності побутових шумів
20	94,25	90,75
200	85,50	76,00

Висновки:

Створення власного голосового асистента дозволяє управляти пристроями безпосередньо на локальному рівні без використання мережі Інтернет, що усуває залежність від нестабільного Інтернет-з'єднання. Створюючи свого асистента, користувач має можливість персоналізувати функціонал під свої потреби та вимоги, що дозволяє отримати максимальну користь від системи управління розумним домом.

Розробка власного датасету для навчання моделі зменшує об'єм використання оперативної пам'яті додатком, дозволяє вести "живу" розмову з асистентом, не застосовуючи фіксованого порядоку слів у команді.

Результати даного дослідження можуть бути використані під час проектування та розроблення голосових систем керування розумними будинками або в інших системах інтернету речей.

Список бібліографічного опису

1. Голосове управління: як технологія управління розумним будинком https://www.smarthouse.ua/ua/golosovoe_upravlenie.html (дата звернення 23.09.2023)
2. Робота з дому: голосовий помічник може стати причиною витоку даних. Malware Protection & Internet Security | ESET. URL: https://www.eset.com/ua/about/newsroom/blog/data-protection/rabota-iz-doma-golosovoy-pomoshchnik-mozhet-stat-prichinoy-utechki-dannykh/ (дата звернення: 18.10.2023).
3. Що таке Siri. URL: https://itech.co.ua/novyny/rozpovidaiemo-shcho-take-sirii-iak-vona-pratsiuie/ (дата звернення 24.04.2023).

References

1. AI Comparison: Siri vs. Cortana vs. Google Assistant vs. Alexa. Business News Daily. URL: https://www.businessnewsdaily.com/10315-siri-cortana-google-assistant-amazon-alexa-face-off.html (date of access: 18.10.2023).
2. Alexa Voice Service. URL: https://developer.amazon.com/en-US/docs/alexa/alexa-voice-service/get-started-with-alexa-voice-service.html (дата звернення 22.04.2023).
3. DeepSpeech Playbook. deepspeech-playbook. URL: https://mozilla.github.io/deepspeech-playbook/ (дата звернення: 17.11.2023).

4. Exploring wav2vec 2.0 on speaker verification and language identification | Semantic Scholar. Semantic Scholar | AI-Powered Research Tool. URL: https://www.semanticscholar.org/reader/9a9d374d1dad72a0349c3a64be93660151274f41 (дата звернення: 17.11.2023).
5. Speech-to-Text: Automatic Speech Recognition | Google Cloud. Google Cloud. URL: https://cloud.google.com/speech-to-text (дата звернення: 17.11.2023).
6. CMUSphinx Open Source Speech Recognition. CMUSphinx Open Source Speech Recognition. URL: https://cmusphinx.github.io/ (дата звернення: 17.11.2023).
7. Google Cloud Speech API URL: https://fotc.com/blog/speech-to-text-what-is/(дата звернення 23.09.2023).
8. Offline Speech Recognition with Vosk. URL: https://blog.anuran.works/offline-speech-recognition-with-vosk (дата звернення 27.09.2023).
9. Kaldi: Modules. Kaldi ASR. URL: https://kaldi-asr.org/doc/modules.html (date of access: 18.10.2023).
10. Normalization of Ukrainian letters, numerals, and measures for natural language processing. In: Digital Scholarship in the Humanities. Published online: 29 December 2022. DOI: 10.1093/llc/fqac090.
URL: https://academic.oup.com/dsh/advance-article-abstract/doi/10.1093/llc/fqac090/6965035. (дата звернення 23.09.2023)
11. EXPLORING WAV2VEC 2.0 ON SPEAKER VERIFICATION AND LANGUAGE IDENTIFICATION Zhiyun Fan, Meng Li, Shiyu Zhou, Bo Xu Institute of Automation, Chinese Academy of Sciences, China School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China URL:https://arxiv.org/abs/2012.06185 (дата звернення 23.09.2023)