НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Факультет електроніки

Кафедра акустичних та мультимедійних електронних систем

«На правах рукопису»
УДК 534.863.5

«До захисту допущено»
Завідувач кафедри

Найда С.А.
(ініціали, прізвище)
“ 10” 12 20 22р.

Магістерська дисертація

зі спеціальності (спеціалізації): 171. Електроніка
(код і назва спеціальності)

на тему: Використання мікроконтролера Arduino для розпізнавання ключових слів

Виконав (-ла): студент (-ка) II курсу, групи ДГ-11мп
(шифр групи)

Рижова Анна Романівна _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник проф. каф. АМЕС, д.т.н. Продеус А.М _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____доц. каф. АМЕС, к.т.н. Онікієнко Ю.О. _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент доц. каф. ЕІ, к.т.н. Шуляк О.П. _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертаціїнемає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Київ – 2022 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет електроніки

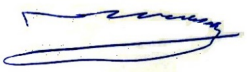
Кафедра Акустичних та мультимедійних електронних систем

Рівень вищої освіти – другий (магістерський) за освітньо-професійною програмою

Спеціальність 171 Електроніка

ЗАТВЕРДЖУЮ

Завідувач кафедри

 __ Найда С.А.

(підпис)

«_01_»_____09_____2022 р.

ЗАВДАННЯ

на магістерську дисертацію

студенту Рижовій Анні Романівні

1. Тема дисертації "Використання мікроконтролера Arduino для розпізнавання ключових слів", керівник роботи д.т.н., проф. кафедри АМЕС Продеус Аркадій Миколайович, затверджені наказом по університету від «8»__11__2022 р. № 4092-с

2. Строк подання студентом дисертації _____

3. Об'єкт дослідження: використання мікроконтролера для розпізнавання мовлення

4. Предмет дослідження (Вихідні дані – для магістерської дисертації за освітньо- професійною програмою): дослідити особливості використання мікроконтролера для розпізнавання мовлення.

5. Перелік графічного матеріалу:

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1,2,3	Онікієнко Ю.О., доц.каф.АМЕС	20.10.2021	30.09.2022

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Збір та вивчення джерел інформації для написання магістерської дисертації; складання бібліографії наукових джерел	30.10.2021	Виконано
2	Аналіз сучасного стану проблеми оцінювання розпізнавання мовлення	30.12.2021	Виконано
3	Аналіз використання обчислювальних ресурсів мікроконтролера для машинного навчання та розпізнавання голосу. Поставлення експерименту для визначення залежності часу розпізнавання ключового слова, об'єму використаної оперативної пам'яті та пам'яті програм в залежності від кількості мел-частотних кепстральних коефіцієнтів та типу згорткової нейронної мережі.	30.05.2022	Виконано
4	Обробка результатів експериментальних досліджень	30.09.2022	Виконано
5	Оформлення дисертації	15.11.2022	Виконано

Студент _____ Рижова А.Р.
(підпис)

Керівник роботи _____
(підпис)

Анотація

Використання нейронних мереж для розпізнавання інформації, зокрема голосу, розширює функціональні можливості вбудованих систем на мікроконтролерах. Але необхідно враховувати обмеження ресурсів мікроконтролера. Мета роботи – проаналізувати вплив параметрів обробки голосу та архітектури нейронної мережі на ступінь використання ресурсів мікроконтролера. Для цього створюється база даних зразків ключового слова, зразків інших слів і голосів, зразків шумів, оцінюється ймовірність розпізнавання ключового слова серед інших слів і шумів, залежність обсягу використовуваної пам'яті від мікроконтролера та встановлено час прийняття рішення від кількості коефіцієнтів MFC, а також встановлено залежність обсягу використаної пам’яті мікроконтролера та часу прийняття рішення від типу згорткової нейронної мережі.

Під час експерименту використовувалася плата Arduino Nano 33 BLE Sense. Модель нейронної мережі була побудована та навчалась на програмній платформі Edge Impulse. Для проведення експерименту було створено три групи даних з назвами «hello», «невідомо», «шум». Група «hello» містить 94 приклади слова «hello» англійською мовою, вимовленого жіночим голосом. Група «невідомі» містить 167 прикладів інших слів, які вимовляються як жіночими, так і чоловічими голосами. Група «шум» містить 166 зразків шуму і випадкових звуків. Згідно з рекомендацією Edge Impulse, 80% зразків з кожної з груп даних використовувалися для навчання моделі нейронної мережі, а 20% зразків використовувалися для тестування.

Аналіз результатів показує, що зі збільшенням кількості коефіцієнтів MFC і, відповідно, точності розпізнавання ключових слів, обсяг програмної пам’яті, зайнятої кодом, збільшується на 480 байт (менше 1%). Для мікроконтролера nRF52840 це не є значним збільшенням. Обсяг використовуваної оперативної пам'яті під час експерименту не змінився. Хоча час розрахунку точності визначення кодового слова збільшився лише на 14 мс (менше 5%) із збільшенням кількості коефіцієнтів MFC, процедура розрахунку

досить тривала (приблизно 0,3 с) у порівнянні з довжиною звукової вибірки. 1 с. Це може бути певним обмеженням при обробці звукового сигналу 32-розрядними мікроконтролерами. Для аналізу фраз або речень необхідно використовувати більш потужні мікроконтролери або мікропроцесори.

За результатами експериментальних досліджень можна стверджувати, що обчислювальних ресурсів 32-розрядних мікроконтролерів цілком достатньо для розпізнавання голосових команд з можливістю попередньої цифрової обробки звукового сигналу, зокрема використання низькочастотних кепстральних коефіцієнтів. Вибір числа коефіцієнтів суттєво не впливає на обсяг використовуваної FLASH і RAM пам'яті мікроконтролера nRF52840. Результати порівняння показують перевагу 2D мережі в точності визначення ключового слова як для 12, так і для 13 коефіцієнтів MFC. Використання одновимірної згорткової нейронної мережі для розпізнавання зразків голосу в проведеному експерименті забезпечує економію пам’яті приблизно на 5%. Якість розпізнавання ключового слова з числом коефіцієнтів MFC 12 становить приблизно 0,7. Для 17 коефіцієнтів MFC якість розпізнавання становить уже 0,97. Обсяг використовуваної оперативної пам'яті у випадку 2D мережі трохи зменшився. Час обробки вибірки голосу для обох типів мереж практично однаковий. Таким чином, одновимірні згорткові нейронні мережі мають певні переваги в додатках мікроконтролерів для обробки та розпізнавання голосу. Обмеженням розпізнавання голосу на мікроконтролері є досить великий час обробки звукового відліку (приблизно 0,3 с) при тривалості самого відліку 1 с, це можна пояснити досить низькою тактовою частотою 64 МГц. Збільшення тактової частоти зменшить час обчислення.

Ключові слова — мікроконтролери, мел-частотні кепстральні коефіцієнти, згорткові нейронні мережі, розпізнавання голосу

Annotation

The functional capabilities of embedded systems using microcontrollers are increased by the use of neural networks for information recognition, particularly speech recognition. However, it is important to consider the microcontroller's resource constraints. The goal of the work is to examine how the architecture of neural networks and voice processing parameters affect how much microcontroller resource is used. To achieve this, a database of samples of the keyword, samples of other words and voices, and samples of noise is created. The likelihood of recognizing the keyword among other words and noises is then assessed, and relationships between the amount of memory used by the microcontroller and the decision-making time on the number of MFC coefficients are established.

The Arduino Nano 33 BLE Sense development board was employed throughout the experiment. The Edge Impulse software platform was used to create and train the neural network model. Three groups of data with the designations "hello," "unknown," and "noise" were constructed in order to carry out the experiment. There are 94 instances of the English word "hello" pronounced by a female voice in the "hello" group. There are 167 instances of additional words in the "unknown" group that are pronounced by both male and female voices. There are 166 samples of noise and random sounds in the "noise" group. 80% of the samples from each of the data groups were used to train the neural network model, and 20% of the samples from each data group were utilized for testing, as suggested by Edge Impulse.

Analysis of the results shows that with an increase in the number of MFC coefficients and, accordingly, the accuracy of keyword recognition, the amount of program memory occupied by the code increases by 480 bytes (less than 1%). For the nRF52840 microcontroller, this is not a significant increase. The amount of RAM used during the experiment did not change. Although the calculation time of the accuracy of the code word definition increased by only 14 ms (less than 5%) with the increase in the number of MFC coefficients, the calculation procedure is quite long (approximately 0.3 s) compared to the sound sample length of 1 s. This can be a certain

limitation when processing a sound signal with 32-bit microcontrollers. To analyze phrases or sentences, it is necessary to use more powerful microcontrollers or microprocessors.

Based on the findings of experimental research, it can be concluded that 32-bit microcontrollers' computational capabilities are more than adequate for voice command recognition with the option of pre-digital sound signal processing, particularly the usage of low-frequency cepstral coefficients. The quantity of FLASH and RAM memory used by the nRF52840 microcontroller is unaffected by the choice of the coefficients' number. The comparison findings demonstrate the 2D network's superiority in terms of keyword definition precision for both 12 and 13 MFC coefficients. A one-dimensional convolutional neural network is used in the experiment to recognize voice samples, which results in a memory savings of about 5%. The effectiveness of keyword recognition with 12 MFC coefficients. When using 12 MFC coefficients, the quality of keyword recognition is roughly 0.7. The recognition quality for 17 MFC coefficients is already 0.97. In the case of the 2D network, less RAM is now being utilized. Both types of networks take essentially the same amount of time to process voice samples. As a result, 1D convolutional neural networks have some advantages in voice processing and recognition applications for microcontrollers. Voice recognition on the microcontroller is limited by the sufficiently low clock frequency of 64 MHz, which accounts for the sufficiently long processing time of the sound sample (about 0.3 s) with the sample duration itself being 1 s. The calculation time will be shortened by raising the clock frequency.

Keywords — microcontrollers, mel-frequency cepstral coefficients, convolutional neural networks, voice recognition

Зміст

ВСТУП

Розпізнавання голосу та мови з використанням вбудованих та мобільних систем дозволяє суттєво розширити область використання нейронних мереж та машинного навчання. В останні роки отримали розвиток системи розпізнавання на мікроконтролерах, що дозволило значно розширити функціональні можливості мікроконтролерних систем.

Процес розпізнавання будується на основі аналізу енергетичного спектру звукового сигналу і може бути виконано різними методами аналізу спектру сигналу в залежності від подальшої обробки голосової інформації. В системах надійного визначення мовця на основі сталої фрази найбільш популярним є використання мел-частотних кепстральних коефіцієнтів - MFCC (Mel Frequency Cepstral Coefficients) та лінійних прогностичних кепстральних коефіцієнтів - LPCC (Linear Predic-tive Cepstral Coefficients) коефіцієнтів. MFCC — це коефіцієнти, обчислені на нелінійній частотній шкалі на основі відомого слухового сприйняття людини, тоді як LPCC — це коефіцієнти, які представляють людську слухову систему на основі лінійного передбачення. LPCC у порівнянні з MFCC забезпечує дещо більшу точність при автентифікації мовця. Однак MFCC потребує значно менше часу для прийняття рішення [1], що є дуже важливою властивістю саме для вбудованих систем з обмеженими обчислювальними ресурсами. Також метод MFCC достатньо просто імплементується у вигляді програмного коду і постійно удосконалюється [2, 3].

Для класифікації голосу або мови у вбудованих системах використовують гаусівську модель суміші (Gauss Mixture Model) [4], приховану модель Маркова (Hidden Markov Model) [5], штучні нейронні мережі (Artificial Neural Network) [6] та інші. Нейронні мережі завдяки гнучкості архітектури та простоті програмної реалізації дуже популярні саме у вбудованих системах, а тому постійно удосконалюються [7].

Вбудовані системи для розпізнавання мови умовно можна розділити на три групи: системи з використанням FPGA (Field-Programmable Gate Array), системи на основі процесора або мікропроцесора і системи на мікроконтролерах. FPGA системи мають високу швидкість обробки інформації, але обмежені в зміні налаштувань [8]. Мікропроцесорні системи мають великий об'єм пам'яті і, як правило, операційну систему. Реалізовують такі системи розпізнавання як на потужних і дорогих платах типу Nvidia Jetson Tegra K1 [9], так і на більш дешевих платах Raspberry PI [10, 11] Відповідно максимальну швидкість розпізнавання забезпечують спеціалізовані плати вартістю від декількох сотень, або навіть тисяч доларів. Мікрокомп'ютери Raspberry PI з власною операційною системою дешевше, але з хорошим функціоналом також коштують до декількох сотень доларів.

Особливий інтерес для використання в IT-індустрії та системах Інтернету речей представляють вбудовані системи розпізнавання мови на мікроконтролерах, як правило 32-х бітних. Головними перевагами таких систем є розвинена периферія та невелика вартість. Недоліками є обмежений об'єм пам'яті та нижчі робочі частоти ядра мікроконтролера, що зменшує швидкість обробки даних у порівнянні з системами на мікропроцесорах. Але для застосувань, де використання розпізнавання мови є однією з функцій системи, використання мікроконтролерів є цілком прийнятне.

В роботі [12] запропоновано систему для голосового виклику медичного працівника побудовану на мікроконтролері STM32. Рішення в системі приймається на порівнянні MFC коефіцієнтів отриманих з голосового сигналу з сигналами, які зберігаються в пам'яті мікроконтролера. Розпізнавання голосу реалізовано у складі домофонної системи на 32-х бітному мікроконтролері, де верифікація голосу виконується за допомогою гаусівської моделі суміші [13]. В системі для розпізнавання голосу, побудованій на мікроконтролері STM32, класифікація голосу, виконується з використанням DTW (Dynamic Time Warping) перетворення після отримання значень LPCC [14].

В останні роки зростає популярність систем розпізнавання на мікроконтролерах, в яких для класифікації використовуються нейронні мережі, зокрема згорткові (Convolutional Neural Network). Це в першу чергу пояснюється наявністю безкоштовних програмних платформ (фреймворків) з широким функціоналом для створення та навчання нейронних мереж. Прикладами таких платформ є Tensorflow [15] та Keras [16], які використовуються не тільки для розпізнавання мови, а й зображень та інформації з сенсорів [17, 18].

Таким чином, використання нейронних мереж для розпізнавання інформації і, зокрема, голосу розширює функціональні можливості вбудованих систем на мікроконтролерах при необхідності врахування обмеженості їх ресурсів, що використовуються. Метою даної роботи є аналіз впливу параметрів обробки голосу та архітектури нейронної мережі на об’єм використаних ресурсів мікроконтролера. Для здійснення аналізу необхідно:

- Створити базу даних з зразків ключового слова, зразків інших слів та голосів та зразків шуму. Оцінити вірогідність розпізнавання ключового слова серед інших слів та шумів;
- Встановити залежності об’єму задіяної пам’яті мікроконтролера та часу прийняття рішення від кількості MFC коефіцієнтів;
- Встановити залежності об’єму задіяної пам’яті мікроконтролера та часу прийняття рішення від типу згорткової нейронної мережі.

РОЗДІЛ 1. Аналіз сучасного стану питання

Завдяки сучасним технологіям комп'ютерне програмне забезпечення тепер може розуміти мовлення. Це програмне забезпечення може слухати мову і інтерпретувати це в цифровану версію, яка читає та аналізує.

Це відбувається завдяки штучному інтелекту і машинному навчанню. Великі обсяги даних використовуються для створення алгоритму, який можна розробити з часом. Потім штучний інтелект вивчає ці дані та визначає закономірності. Він переглядає попередні введення та фіксує те, що говориться.

Розпізнавання голосу означає, що мобільний пристрій, розумні колонки або комп'ютер можуть слухати те, що говориться. Ця розширена функціональність може стати в нагоді, коли потрібна допомога по дому. Його також можна використовувати для диктування нотаток, коли немає часу або фізичних засобів, щоб записати їх.

Багато компаній також використовують його для покращення обслуговування клієнтів. Абоненти можуть відповісти на певні запитання та бути направленими до потрібної людини для вирішення їхньої проблеми. Це технологія розпізнавання голосу RingCentral. Вона покращує частоту вирішення першого дзвінка та гарантує, що агентам не доведеться переадресовувати дзвінки в інші відділи. Це чудово для клієнтів, які швидко та ефективно вирішують свої проблеми, і чудово для бізнесу, щоб підвищити продуктивність і прийняти більше дзвінків [19].

У деяких системах розпізнавання голосу використовується біометрична технологія. Ця технологія сьогодні стає дуже популярною в цілях безпеки та для електронних проектів серед студентів інженерних спеціальностей. Завдяки цьому людей легко ідентифікувати, і ймовірність крадіжки та шахрайства зменшується. Іншими методами біометричної ідентифікації є сканування райдужної оболонки ока/очі, відбитки пальців, сканування обличчя, відбитки рук, відбитки голосу, почерк тощо.

За допомогою біометричної системи розпізнавання голосу можна розпізнати унікальні голосові характеристики людини. Ця система безпеки має широкий спектр застосування та використання, як для виробників банкоматів, виробників автомобілів, так і в системах безпеки мобільного телефону для запобігання будь-якої крадіжки чи шахрайства. Він також має багато застосувань у вбудованих програмах.

Система розпізнавання голосу — це здатність пристрою розуміти голосові інструкції. Фактично це тип вбудованої системи. При використанні з комп'ютером використовується АЦП, який перетворює різні аналогові голосові сигнали в цифрові імпульси або цифрові сигнали, які легко сприймаються комп'ютером. На жорсткому диску вже збережені форми мови. Голосовий сигнал декодується та перевіряється на збережені форми. Іноді через наявність інших голосів і шумів результат не є точним.

Щоб перетворити мову або вимовлені слова в команду комп'ютера, комп'ютер виконує кілька складних кроків. Аналого-цифровий перетворювач перетворює голосовий сигнал у цифровий сигнал для комп'ютера. АЦП відцифровує звукову хвилю через часті проміжки часу, виконуючи деякі точні вимірювання. Цей дискретизований або цифрований звук потім фільтрується для видалення шуму. Це також робиться для розділення звуку в різних діапазонах частот. Завдяки цьому звук також нормалізується. Різні люди мають різну швидкість розмови, тому звук налаштовується таким чином, щоб він відповідав швидкості звукового шаблону, збереженого в пам'яті системи.

Конструкція апаратного забезпечення найпростішої системи безпеки розпізнавання голосу включає три основні елементи:

- Схема мікрофона.
- Схема мікроконтролера.
- LCD-дисплей.

Схема мікрофона підключається до АЦП контролера. У пам'яті мікроконтролера зберігається набір слів і фраз. Як тільки слово промовляється в мікрофон, АЦП перетворює його в цифрові сигнали, які проходять через цифрові фільтри, і нарешті РК-дисплей, підключений до мікроконтролера, відображає вимовлені слова.

Існують такі датчики розпізнавання мовлення:

1. Ультразвукові датчики:

Ультразвукова обробка схожа на радарну. Надвисокочастотний акустичний сигнал кидається на рухомий об'єкт, створені відбиття реєструються приймачем. Ефект Допплера визначає частоту відбитого тону, рівняння для нього можна виразити так:

$$f = f0(1+ v c), \qquad (1)$$

де f0 = частота випромінюваного тону

f = частота відбитого тону

v = швидкість поверхні, що відбиває, до випромінювача

c = швидкість звуку

Таким чином, можна зробити висновок, що якщо поверхня, що відбиває, віддаляється від випромінювача, тон записаної частоти буде нижчим, і навпаки. Відбитий сигнал складатиметься із суми синусоїд, що мають різну силу та частоту. У випадку, коли людина розмовляє, рух артикулятора під час промови викликатиме відображення. Для розрізнення звуків мовлення потенційно можуть бути використані шаблони частоти часу.

2. Фізіологічний сенсор

Це прилад, розроблений у Армійській науково-дослідній лабораторії. Цей датчик фізично з’єднується з пацієнтом і записує медичну інформацію, таку як серцебиття та дихання пацієнта. Цей датчик одягається на горло.

Це корисно в місцях, де занадто багато шуму. Слова, вимовлені людиною в мікрофон, порівнюються з даними, отриманими від фізіологічного датчика, прикріпленого на шиї людини, і потім легко визначаються слова або речення.

Раніше за допомогою цього датчика було важко розпізнати мову через спотворення мови. Але пізніше Науковий центр Роквелла розробив розпізнавач мовлення на основі прихованої марківської моделі для використання з фізіологічним датчиком.

Існує чотири типи систем розпізнавання голосу:

1. Ізольована система розпізнавання голосу. Ця система вимагає короткого переходу між вимовленими словами.
2. Система безперервного розпізнавання голосу. Як випливає з назви, ця система не потребує переходу між словами.
3. Система розпізнавання, що залежить від людини. Ця система розпізнає мову лише однієї людини.
4. Незалежна система розпізнавання мовлення: Ця система може ідентифікувати мову будь-якої людини [20].

1.1Сучасні методи розпізнавання мовлення

1.1.1.Приховані марковські моделі

Сучасні системи розпізнавання мови загального призначення базуються на прихованих моделях Маркова. Це статистичні моделі, які виводять послідовність символів або величин. Приховані марковські моделі використовуються для розпізнавання мовлення, тому що мовний сигнал можна розглядати як частково стаціонарний сигнал або короткочасний стаціонарний сигнал. У короткому часовому масштабі (наприклад, 10 мілісекунд) мова може бути апроксимована як стаціонарний процес. Мовлення можна розглядати як марковську модель для багатьох стохастичних цілей.

Інша причина популярності прихованих марковських моделей полягає в тому, що їх можна навчити автоматично, а також вони прості та зручні для використання. У розпізнаванні мовлення прихована модель Маркова виводить послідовність n-вимірних дійсних векторів (де n є малим цілим числом, наприклад 10), виводячи один із них кожні 10 мілісекунд. Вектори складатимуться з кепстральних коефіцієнтів, які отримують за допомогою перетворення Фур'є короткого часового вікна мовлення та декореляції спектра за допомогою косинусного перетворення, потім беруть перші (найзначніші) коефіцієнти. Кожне слово або (для більш загальних систем розпізнавання мовлення) кожна фонема матиме різний вихідний розподіл.

Прихована модель Маркова для послідовності слів або фонем створюється шляхом об'єднання окремих навчених прихованих моделей Маркова для окремих слів і фонем.

Вище описано основні елементи найпоширенішого підходу до розпізнавання мовлення на основі прихованих марківських моделі. Сучасні системи розпізнавання мовлення використовують різні комбінації низки стандартних методів, щоб покращити результати порівняно з базовим підходом, описаним вище. Типова система з великим словниковим запасом потребує контекстної залежності для фонем (тому фонеми з різним лівим і правим контекстом мають різні реалізації як стани ПММ).

Використовуватиметься кепстральна нормалізація для нормалізації для іншого мовця та умов запису; для подальшої нормалізації мовця може використовуватись нормалізація довжини голосового тракту для нормалізації між чоловіками та жінками та лінійну регресію максимальної ймовірності для більш загальної адаптації мовця. Функції матимуть так звані дельта- та дельта-дельта-коефіцієнти для фіксації динаміки мовлення, а також можуть використовувати гетероскедастичний лінійний дискримінантний аналіз.

Багато систем використовують так звані дискримінаційні методи навчання, які обходяться без суто статистичного підходу до оцінки параметрів ПММ і натомість оптимізують певну міру навчання, пов’язану з класифікацією. Прикладами є максимальна взаємна інформація, мінімальна помилка класифікації і мінімальна телефонна помилка.

Декодування мовлення (термін для того, що відбувається, коли системі представлено нове висловлювання та має обчислити найбільш ймовірне вихідне речення), ймовірно, використовуватиме алгоритм Вітербі, щоб знайти найкращий шлях, і тут є вибір між динамічним створенням комбінована прихована марковська модель, яка включає в себе як інформацію про акустичну, так і про мовну моделі та попередньо об’єднує її статично (підхід перетворювача кінцевого стану, або FST).

Можливе вдосконалення декодування полягає в тому, щоб зберегти набір хороших кандидатів замість того, щоб просто зберегти найкращого кандидата, і використовувати кращу функцію підрахунку балів (повторне оцінювання), щоб оцінити хороших кандидатів, щоб можна було вибрати найкращого відповідно до цього вдосконаленого балу . Набір кандидатів можна зберігати або як список (підхід списку N-найкращих), або як підмножину моделей (решітку).

Переоцінка зазвичай виконується, намагаючись мінімізувати ризик Байєса [21] (або його наближення): замість того, щоб брати вихідне речення з максимальною ймовірністю, намагаємося взяти речення, яке мінімізує очікування даної функції втрат щодо усі можливі транскрипції (тобто беремо речення, яке мінімізує середню відстань до інших можливих речень, зважених за їхньою оціненою ймовірністю). Функція втрат зазвичай є відстанню Левенштейна, хоча це можуть бути різні відстані для конкретних завдань; набір можливих транскрипцій, звичайно, скорочується, щоб підтримувати зручність. Було розроблено ефективні алгоритми для повторного оцінювання решіток, представлених у вигляді зважених перетворювачів кінцевого стану з відстанями

редагування, представлених у вигляді перетворювача кінцевого стану, що перевіряє певні припущення [22].

1.1.2 Розпізнавання мовлення на основі динамічного викривлення часу (DTW).

Динамічне викривлення часу — це підхід, який історично використовувався для розпізнавання мовлення, але зараз значною мірою витіснений більш успішним підходом на основі ПММ.

Динамічне викривлення часу — це алгоритм для вимірювання подібності між двома послідовностями, які можуть відрізнятися за часом або швидкістю. Наприклад, буде виявлено схожість у моделях ходьби, навіть якщо на одному відео людина йшла повільно, а на іншому – швидше, або навіть якщо під час одного спостереження були прискорення та уповільнення. Динамічне викривлення часу застосовувався до відео, аудіо та графіки – справді, будь-які дані, які можна перетворити на лінійне представлення, можна аналізувати за допомогою динамічного викривлення часу.

Добре відомим додатком було автоматичне розпізнавання мовлення, щоб справлятися з різними швидкостями мовлення. Загалом, це метод, який дозволяє комп’ютеру знаходити оптимальну відповідність між двома заданими послідовностями (наприклад, часовими рядами) з певними обмеженнями. Тобто послідовності «викривляються» нелінійно, щоб відповідати одна одній. Цей метод вирівнювання послідовності часто використовується в контексті прихованих марковських моделей.

1.1.3 Нейронні мережі

Нейронні мережі з’явилися як привабливий підхід до акустичного моделювання наприкінці 1980-х років. Відтоді нейронні мережі використовувалися в багатьох аспектах розпізнавання мовлення, таких як класифікація фонем [23], класифікація фонем за допомогою багатоцільових

еволюційних алгоритмів [24] розпізнавання ізольованих слів [25] аудіовізуальне розпізнавання мовлення, аудіовізуальне розпізнавання мовця.

Нейронні мережі роблять менше явних припущень щодо статистичних властивостей функцій, ніж ПММ, і мають кілька якостей, що робить їх привабливими моделями розпізнавання для розпізнавання мови. Коли нейронні мережі використовуються для оцінки ймовірностей сегмента ознак мовлення, вони дозволяють проводити дискримінаційне навчання природним і ефективним способом. Однак, незважаючи на їх ефективність у класифікації короткочасних одиниць, таких як окремі фонеми та ізольовані слова[26], ранні нейронні мережі рідко були успішними для завдань безперервного розпізнавання через їх обмежену здатність моделювати тимчасові залежності.

Одним із підходів до цього обмеження було використання нейронних мереж як етапу попередньої обробки, трансформації ознак або зменшення розмірності [27] перед розпізнаванням на основі ПММ. Однак нещодавно мережі з довготривалою короткостроковою пам'яттю (LSTM) і пов’язані з ним рекурентні нейронні мережі (RNN)[28][29] і нейронні мережі із затримкою часу (TDNN)[30] продемонстрували покращену продуктивність у цій галузі.

1.1.4 Глибокі прямі та рекурентні нейронні мережі

Також досліджуються глибокі нейронні мережі та автокодери з усуненням шуму [31]. Глибока нейронна мережа прямого зв’язку (DNN - deep neural network) — це штучна нейронна мережа з кількома прихованими шарами одиниць між вхідним і вихідним рівнями. Подібно до дрібних нейронних мереж, DNN можуть моделювати складні нелінійні зв’язки. Архітектури DNN генерують композиційні моделі, де додаткові рівні дозволяють створювати функції з нижчих рівнів, надаючи величезну здатність до навчання, а отже, потенціал моделювання складних шаблонів мовних даних [32].

Успіх DNN у розпізнаванні мовлення великого словника відбувся в 2010 році промисловими дослідниками у співпраці з академічними дослідниками, де були прийняті великі вихідні рівні DNN на основі контекстно-залежних станів HMM, побудованих за допомогою дерев рішень [33][34] [35].

Одним із фундаментальних принципів глибокого навчання є відмова від ручної розробки функцій і використання необроблених функцій. Цей принцип вперше був успішно досліджений в архітектурі глибокого автокодувальника на «необроблених» спектрограмах або функціях лінійного банку фільтрів [36], показуючи його перевагу над функціями Mel-Cepstral, які містять кілька етапів фіксованого перетворення спектрограм. Нещодавно було показано, що справжні «необроблені» особливості мови, форми хвилі, дають чудові результати розпізнавання мовлення у великому масштабі.[37]

1.1.5 Наскрізне автоматичне розпізнавання мовлення

З 2014 року спостерігався великий дослідницький інтерес до «наскрізного» автоматичного розпізнавання мовлення. Традиційні фонетичні підходи (тобто всі моделі, засновані на HMM) вимагали окремих компонентів і навчання моделі вимови, акустичної та мовної моделі. Наскрізні моделі спільно вивчають усі компоненти розпізнавача мови. Це важливо, оскільки спрощує процес навчання та розгортання. Наприклад, модель мови n-gram необхідна для всіх систем на основі ПММ, а типова модель мови n-gram часто займає кілька гігабайт пам’яті, що робить її непрактичною для розгортання на мобільних пристроях [38]. Отже, сучасні комерційні системи автоматичного розпізнавання мовлення від Google і Apple (станом на 2017 рік) розгортаються в хмарі та потребують підключення до мережі, а не локального пристрою.

Перша спроба наскрізного автоматичного розпізнавання мовлення була з системами на основі часової класифікації зв’язків (CTC), представленими Алексом Грейвсом з Google DeepMind і Навдіпом Джайтлі з Університету Торонто в 2014 році [39]. Модель складалася з рекурентних нейронних мереж і

CTC-шару. Спільно модель RNN-CTC вивчає вимову та акустичну модель разом, однак вона не здатна вивчати мову через припущення умовної незалежності, подібні до ПММ. Отже, моделі CTC можуть безпосередньо навчитися відображати акустику мовлення англійським символам, але моделі допускають багато поширених орфографічних помилок і повинні покладатися на окрему мовну модель для очищення транскриптів. Пізніше Baidu розширив роботу з надзвичайно великими наборами даних і продемонстрував певний комерційний успіх китайською мандаринською та англійською мовами [40].

У 2016 році Оксфордський університет представив LipNet [41], першу наскрізну модель читання з губ на рівні речення, що використовує просторово-часові згортки в поєднанні з архітектурою RNN-CTC, що перевершує продуктивність людського рівня в обмеженому наборі граматичних даних [42]. Масштабна архітектура CNN-RNN-CTC була представлена в 2018 році компанією Google DeepMind, яка досягла в 6 разів кращої продуктивності, ніж люди-експерти [43].

Альтернативним підходом до моделей на основі CTC є моделі на основі уваги. Моделі ASR на основі уваги були введені одночасно Чаном та ін. Університету Карнегі-Меллона та Google Brain і Bahdanau et al. Монреальського університету в 2016 році [44][45]. Модель під назвою «Listen, Attend and Spell» (LAS) буквально «слухає» акустичний сигнал, звертає «увагу» на різні частини сигналу та «промовляє» розшифровку по одному символу за раз. На відміну від моделей на основі CTC, моделі на основі уваги не мають припущень про умовну незалежність і можуть безпосередньо вивчати всі компоненти розпізнавача мовлення, включаючи вимову, акустику та мовну модель. Це означає, що під час розгортання немає необхідності носити мовну модель, що робить її дуже зручною для програм з обмеженою пам’яттю. До кінця 2016 року моделі на основі уваги досягли значного успіху, включаючи перевершення моделей CTC (з або без моделі зовнішньої мови) [46]. Починаючи з оригінальної моделі LAS, були запропоновані різні розширення.

Latent Sequence Decompositions (LSD) було запропоновано Університетом Карнегі-Меллона, Массачусетським технологічним інститутом і Google Brain для прямого випромінювання одиниць підслів, які є більш природними, ніж англійські символи;[47] Оксфордський університет і Google DeepMind розширили LAS до «Дивитися, слухати, бути присутнім» для читання з губ, що перевершує продуктивність людського рівня [48].

1.2 Використання розпізнавання голосу

На сьогоднішній день 72% людей, які користуються пристроями голосового пошуку, стверджують, що вони стали частиною їх повсякденних справ. Технології швидко розвиваються, і іноді наступна старі розробки затьмарюються новою розробкою. Але чим більше людей почуваються комфортно розмовляти по телефону, тим більше ця тенденція набирає популярності.

Розпізнавання голосу використовується не лише для особистого користування. У міру того, як індустрії та підприємства підключаються, тенденція використання розпізнавання голосу – лише питання часу, коли ця кількість зросте. Все більше компаній використовують системи розпізнавання голосу, щоб допомогти їм з ефективністю та точністю в обслуговуванні клієнтів.

Ось деякі з основних застосувань розпізнавання голосу:

- Диктування

Технологію розпізнавання мовлення можна використовувати різними способами. Зараз багато галузей використовують розпізнавання голосу, щоб допомогти в повсякденних процесах. Наприклад, юридична галузь отримала велику користь від розпізнавання голосу. Юристи використовують його для

диктування важливих зустрічей, які потім можуть записувати в документи. Це не тільки економить їхній час, але й забезпечує точний запис усієї інформації.

Це також допомагає у звичайних повсякденних заняттях. У багатьох із нас є смартфони, які також мають віртуального помічника, і можна продиктувати свій список покупок, щоденні завдання та майже все, що потрібно зробити нотаткою. Це легше, а часто й продуктивніше, ніж писати це самостійно.

- Використання у зворотньому порядку

Розпізнавання голосу також можна використовувати в зворотному порядку, тобто замість перетворення мови в текст можливо перекладати текст у мову. Деякі платформи, такі як Dragon Professional від Nuance, пропонують цю функцію. Багато людей, які мають проблеми з промовою та зором. Люди з обмеженими можливостями чи вадами мови, вважають його корисним. З цієї причини його також можна використовувати в освітньому секторі.

- Покупки за допомогою голосової команди

Понад 55% клієнтів придбали продукт на веб-сайті електронної комерції за допомогою розпізнавання мовлення. І, оскільки більше людей знайомляться з технологією розпізнавання голосу, це число може зрости.

- n-car системи

Зазвичай ручне введення керування, наприклад за допомогою керування пальцем на кермі, вмикає систему розпізнавання мовлення, і про це водієві повідомляється звуковою підказкою. Після звукової підказки система має «вікно прослуховування», протягом якого вона може прийняти мовний ввід для розпізнавання.

Прості голосові команди можна використовувати для ініціювання телефонних дзвінків, вибору радіостанцій або відтворення музики з сумісного смартфона, MP3-плеєра або музичного флеш-накопичувача. Можливості розпізнавання голосу залежать від марки та моделі автомобіля. Деякі з найновіших моделей автомобілів пропонують розпізнавання мовлення

природною мовою замість фіксованого набору команд, що дозволяє водієві використовувати повні речення та загальні фрази. Таким чином, у таких системах користувачеві немає потреби запам’ятовувати набір фіксованих командних слів.

- Охорона здоров'я

У секторі охорони здоров’я розпізнавання мовлення може бути реалізовано у передньому або задньому кінці процесу медичної документації. Внутрішнє розпізнавання мовлення – це те, що постачальник диктує в механізм розпізнавання мовлення, розпізнані слова відображаються під час промовлення, а диктатор відповідає за редагування та підписання документа. Внутрішнє або відкладене розпізнавання мовлення полягає в тому, що постачальник диктує в систему цифрового диктування, голос направляється через машину розпізнавання мовлення, а розпізнаний чернетка документа направляється разом із оригінальним голосовим файлом до редактора, де чернетка редагується і звіт завершено. Відкладене розпізнавання мовлення зараз широко використовується в галузі.

- Терапевтичне використання

Тривале використання програмного забезпечення для розпізнавання мовлення в поєднанні з текстовими процесорами показало переваги для зміцнення короткочасної пам’яті у пацієнтів з АВМ мозку, які пройшли резекцію. Необхідно провести подальші дослідження, щоб визначити когнітивні переваги для осіб, у яких АВМ лікували за допомогою радіологічних методів.

- Військова справа

Високоефективний винищувач

В останнє десятиліття значні зусилля були спрямовані на випробування та оцінку розпізнавання мови в літаках-винищувачах. Особливої уваги заслуговує програма США з розпізнавання мовлення для літаків Advanced Fighter

Technology Integration (AFTI)/F-16 (F-16 VISTA), програма у Франції для літаків Mirage та інші програми у Великобританії, що стосуються різноманітних авіаційних платформ. У цих програмах розпізнавання мовлення успішно використовувалося в літаках-винищувачах із застосуваннями, включаючи налаштування радіочастот, керування системою автопілота, встановлення координат точки керма та параметрів випуску зброї та керування дисплеєм польоту.

Працюючи зі шведськими пілотами, які літали в кабіні JAS-39 Gripen, Енглунд (2004) виявив, що розпізнавання погіршується зі збільшенням перевантажень. У звіті також зроблено висновок, що адаптація значно покращила результати в усіх випадках і що введення моделей для дихання показало значне покращення показників розпізнавання. Всупереч тому, що можна було очікувати, жодних наслідків ламаної англійської мови спікерів виявлено не було. Було очевидно, що спонтанна мова викликала проблеми для впізнавача, як і можна було очікувати. Тому можна очікувати, що обмежений словниковий запас і, перш за все, правильний синтаксис значно підвищить точність розпізнавання [49].

Винищувач Eurofighter Typhoon, який зараз перебуває на озброєнні Королівських ВПС Великої Британії, використовує систему, що залежить від динаміків, і кожен пілот повинен створити шаблон. Система не використовується для критично важливих для безпеки або зброї завдань, таких як випуск зброї або опускання шасі, але використовується для широкого спектру інших функцій кабіни. Голосові команди підтверджуються візуальним і/або звуковим зворотним зв'язком. Система розглядається як головна конструктивна особливість у зменшенні робочого навантаження на пілота[50], і навіть дозволяє пілоту призначати цілі своєму літаку за допомогою двох простих голосових команд або будь-якому зі своїх супутніх лише п'ятьма командами [51].

Також розробляються і тестуються незалежні від гучномовців системи для F35 Lightning II (JSF) і початкового навчально-тренувального винищувача Alenia Aermacchi M-346 Master. Ці системи показали точність слів, що перевищує 98% [52].

- Гелікоптери

Проблеми досягнення високої точності розпізнавання в умовах стресу та шуму є особливо актуальними в середовищі вертольотів, а також у середовищі реактивних винищувачів. Проблема акустичного шуму насправді є більш серйозною в середовищі гелікоптера не лише через високий рівень шуму, але також через те, що пілот гелікоптера, як правило, не носить маску, яка б зменшила акустичний шум у мікрофоні. За останнє десятиліття було проведено значні програми випробувань і оцінки систем розпізнавання мовлення на вертольотах, зокрема Дослідницько-розробною діяльністю армії США (AVRADA) і Королівським аерокосмічним закладом (RAE) у Великобританії. Робота у Франції включала розпізнавання мови в вертольоті Puma. У Канаді також було багато корисної роботи. Результати були обнадійливими, і голосові програми включали: керування радіостанціями зв'язку, налаштування навігаційних систем та керування автоматизованою системою передачі цілей.

Як і у винищувачах, головною проблемою голосу на вертольотах є вплив на ефективність пілота. Повідомляється про обнадійливі результати випробувань AVRADA, хоча вони є лише демонстрацією здійсненності в тестовому середовищі. Ще багато чого потрібно зробити як у розпізнаванні мовлення, так і в загальних мовних технологіях, щоб постійно досягати підвищення продуктивності в робочих налаштуваннях.

- Підготовка авіадиспетчерів

Навчання диспетчерів повітряного руху є чудовим додатком для систем розпізнавання мовлення. Зараз багато систем навчання вимагають, щоб особа діяла як «псевдопілот», беручи участь у голосовому діалозі з диспетчером-

стажером, який імітує діалог, який диспетчер мав би вести з пілотами в реальній ситуації. Технології розпізнавання та синтезу мовлення можуть позбавити людину необхідності виконувати роль псевдопілота, таким чином скорочуючи навчання та допоміжний персонал.

- Телефонія та інші домени

Автоматичне розпізнавання мовлення зараз є звичним явищем у сфері телефонії та стає все більш поширеним у сфері комп'ютерних ігор та симуляції. У системах телефонії ASR зараз переважно використовується в контакт-центрах шляхом інтеграції з системами IVR. Незважаючи на високий рівень інтеграції з обробкою текстів у загальних персональних комп'ютерах, у сфері виробництва документів ASR не побачив очікуваного зростання використання.

Покращення швидкості мобільного процесора зробило розпізнавання мовлення практичним у смартфонах. Мовлення використовується здебільшого як частина інтерфейсу користувача для створення попередньо визначених або настроюваних мовних команд.

- Використання в навчанні та повсякденному житті

Для вивчення мови розпізнавання мовлення може бути корисним для вивчення другої мови. Це може навчити правильної вимови, а також допомогти людині розвинути вільне мовлення [53].

Сліпі учні або мають дуже слабкий зір можуть скористатися технологією для передачі слів, а потім почути, як комп'ютер їх читає, а також використовувати комп'ютер, керуючи голосом, замість того, щоб дивитися на екран і клавіатура.

Учні з обмеженими фізичними можливостями можуть бути звільнені від необхідності турбуватися про рукописний текст, набір тексту або роботу з писарем над шкільними завданнями за допомогою програм синтезу мовлення в текст. Вони також можуть використовувати технологію розпізнавання мовлення, щоб насолоджуватися пошуком в Інтернеті або використанням

комп’ютера вдома без необхідності фізично керувати мишею та клавіатурою [54].

Розпізнавання мовлення може дозволити учням із вадами навчання стати кращими письменниками. Вимовляючи слова вголос, вони можуть збільшити плавність свого письма та позбутися занепокоєння щодо орфографії, пунктуації та інших механізмів письма [55].

Використання програмного забезпечення для розпізнавання голосу в поєднанні з цифровим аудіозаписувачем і персональним комп’ютером, на якому запущено програмне забезпечення для обробки тексту, виявилося позитивним для відновлення пошкодженої короткочасної пам’яті в осіб, які перенесли інсульт і черепно-мозкову травму.

- Люди з обмеженими можливостями

Люди з обмеженими можливостями можуть скористатися програмами розпізнавання мовлення. Для людей із вадами слуху та слуху програмне забезпечення для розпізнавання мовлення використовується для автоматичного створення закритих субтитрів під час розмов, таких як дискусії в конференц-залах, лекції в класі [56].

Розпізнавання мовлення також дуже корисно для людей, які мають труднощі з використанням рук, починаючи від легких повторюваних стресових травм і закінчуючи інвалідністю, яка перешкоджає використанню звичайних комп’ютерних пристроїв введення.

Розпізнавання мовлення використовується в телефонії для глухих, наприклад голосова пошта в текст, служби ретрансляції та телефон із субтитрами. Люди з обмеженими можливостями навчання, які мають проблеми з комунікацією «думка-папір» (по суті, вони думають про ідею, але вона обробляється неправильно, через що вона виглядає по-іншому на папері), можуть отримати користь від програмного забезпечення, але технологія не є стійкою до помилок [57].

Цей тип технології може допомогти людям з дислексією, але інші вади залишаються під питанням. Ефективність продукту - це проблема, яка заважає йому бути ефективним. Хоча дитина може вимовити слово залежно від того, наскільки чітко вона його вимовляє, технологія може подумати, що вони говорять інше слово, і ввести неправильне. Даючи їм більше роботи для виправлення, змушуючи їх витрачати більше часу на виправлення неправильного слова[58].

1.3 Продуктивність систем розпізнавання мови

Продуктивність систем розпізнавання мови зазвичай оцінюється з точки зору точності та швидкості.[59]Точність зазвичай оцінюється за коефіцієнтом помилок у словах (WER), тоді як швидкість вимірюється за коефіцієнтом реального часу. Інші показники точності включають частоту помилок одного слова (SWER) і частоту успішних команд (CSR).

Однак розпізнавання мовлення машиною є дуже складною проблемою. Вокалізація розрізняється за акцентом, вимовою, артикуляцією, грубістю, назальністю, висотою, гучністю та швидкістю. Мова спотворена фоновим шумом і луною, електричними характеристиками. Точність розпізнавання мовлення може відрізнятися в залежності від наступного: [60]

- Обсяг словникового запасу та можливість сплутування
- Залежність від оратора проти незалежності
- Ізольоване, уривчасте або безперервне мовлення
- Завдання та мовні обмеження
- Прочитане проти спонтанного мовлення
- Несприятливі умови

Точність

Точність розпізнавання мовлення може відрізнятися залежно від таких факторів:

- Рівень помилок зростає зі збільшенням словникового запасу:

напр. 10 цифр від «нуля» до «дев'яти» можна розпізнати практично ідеально, але розмір словника 200, 5000 або 100000 може мати рівень помилок 3%, 7% або 45% відповідно.

- Словниковий запас важко розпізнати, якщо він містить заплутані слова:

напр. 26 літер англійського алфавіту важко розрізнити, оскільки вони плутають слова (найвідоміший набір E: «B, C, D, E, G, P, T, V, Z — коли вимовляється «Z» «zee», а не «zed» залежно від англійського регіону); рівень помилок у 8% вважається хорошим для цього словника.

- Залежність від оратора проти незалежності:

Система, яка залежить від динаміка, призначена для використання одним динаміком.

Система, незалежна від динаміка, призначена для використання будь-яким мовцем (більш складним).

- Ізольоване, переривчасте або безперервне мовлення

При ізольованому мовленні використовуються окремі слова, тому розпізнати мову стає легше.

1.4 Переваги та недоліки розпізнавання голосу

Хоча багато людей вважають розпізнавання голосу частиною людського майбутнього, є деякі недоліки, які слід враховувати. Ось переваги та недоліки розпізнавання голосу:

Переваги:

- Допомагає підвищити продуктивність у багатьох підприємствах, наприклад у галузях охорони здоров'я.
- Можливість перехоплювати мову набагато швидше, ніж диктування
- Використання перетворення тексту в мовлення в режимі реального часу.

- Допомагає тим, хто має проблеми з мовою або зором

Недоліки:

- Голосові дані можуть бути записані, що, побоюються деяких, може вплинути на конфіденційність.
- Програмне забезпечення може мати проблеми зі словниковим запасом, особливо якщо є спеціальні терміни.
- Він може неправильно витлумачити слова, якщо вимова не є чіткою [61].

Висновки до першого розділу

Отже, у першому розділі були розглянуті методи розпізнавання голосу, зокрема приховані марківські моделі, розпізнавання мовлення на основі динамічного викривлення часу (DTW), нейронні мережі, глибокі прямі та рекурентні нейронні мережі, наскрізне автоматичне розпізнавання мовлення.

Також, були розглянуті сучасні системи розпізнавання мовлення, конструкція апаратного забезпечення найпростішої системи безпеки розпізнавання голосу. Також, були розглянуті датчики розпізнавання мовлення, зокрема ультразвуковий датчик та фізіологічний сенсор.

Розглянуто використання розпізнавання голосу у життєдіяльності людей. В основному розпізнавання застосовується для диктування, для перекладання тексту у мову та для використання у комерції. Також, розпізнавання мовлення має застосування у медицині та військовій справі.

Також, розглянуто основні переваги та недоліки використання розпізнавання мовлення у життєдіяльності людей.

РОЗДІЛ 2 Теоретичні засади роботи

2.1 Коефіцієнти мел-частотного кепстру (MFCC)

Коефіцієнти мел-частотного кепстру (MFCC) широко використовуються в автоматичному розпізнаванні мови та мовця. MFCC — це представлення, визначене як реальний кепстр віконного короткочасного сигналу, отриманого з швидкого перетворення Фур'є цього сигналу. Відмінність від справжнього кепстру полягає в тому, що використовується нелінійна частотна шкала, яка апроксимує властивості слухової системи, що підвищує якість розпізнавання мови. Мел шкала пов'язує відчуту частоту або висоту чистого тону з його фактично виміряною частотою. Люди набагато краще розрізняють невеликі зміни висоти на низьких частотах, ніж на високих. Завдяки використанню цієї шкали Мел функції точніше відповідають тому, що чують люди. Мел шкала в залежності від частоти f має наступний вигляд [62]:

$$B(f) = 1125 \ln(1 + f / 700) \qquad (2)$$

Зворотне перетворення від Мел шкали b до частоти має наступний вигляд:

$$B^{-1}(b) = 700(e^{b/1125} - 1) \qquad (3)$$

Для дискретного перетворення Фур'є з N відліків

$$X_a[k] = \sum_{n=0}^{N-1} x[n] e^{-j 2\pi n k / N}, \ 0 \le k < N \qquad (4)$$

можна визначити масив із M фільтрів (m = 1,2, , M), де фільтр m є фільтром трикутної форми, заданим за формулою:

$$H_m[k] = \begin{cases} 0 & k < f[m-1] \\ \dfrac{(k - f[m-1])}{(f[m] - f[m-1])} & f[m-1] \le k \le f[m] \\ \dfrac{(f[m+1] - k)}{(f[m+1] - f[m])} & f[m] \le k \le f[m+1] \\ 0 & k > f[m+1] \end{cases} \qquad (5)\ [62]$$

При застосуванні таких фільтрів обчислюється середній спектр навколо кожної центральної частоти зі збільшенням смуги пропускання (рис.2).

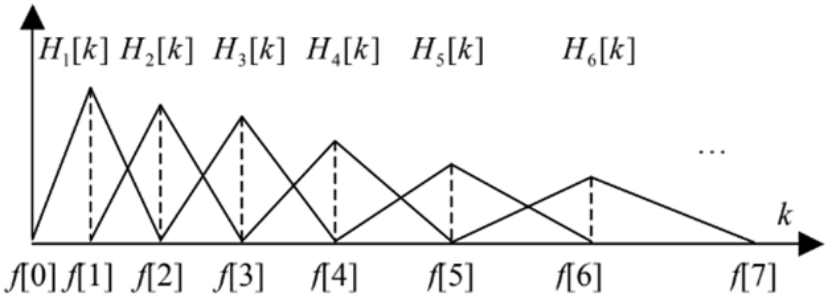


Рис. 1. Трикутні фільтри, які використовуються для обчислення мель-кепстра за допомогою рівняння (5)

Граничні точки $f[m]$ рівномірно розташовані на Мел шкалі:

$$f[m]=\left(\frac{N}{F_s}\right)B^{-1}(B(f_1+m\frac{B(f_h)-B(f_1)}{M+1}) \qquad (6)$$

де f_l та f_h можна визначити як найнижчу та найвищу частоти групи фільтрів у герцах, F_s— частота дискретизації в герцах. Логарифм енергії на виході кожного фільтра можна обчислити як:

$$S[m]=\ln\left[\sum_{k=0}^{N-1}\left|X_a[k]\right|^2 H_m[k]\right],\ 0\le m<M \qquad (7)$$

Тоді мел частотний кепстр є дискретним косинусним перетворенням M виходів фільтрів:

$$c[n]=\sum_{m=0}^{M-1}S[m]\cos(\pi n(m+1/2)/M),\ 0\le n<M \qquad (8)$$

де M змінюється для різних реалізацій від 24 до 40. Для розпізнавання мови зазвичай використовуються тільки перші 13 коефіцієнтів кепстру [62].

Таким чином, в результаті спектральної обробки звукового зразка тривалістю 1 с отримано його спектрограму, яка представляє собою матрицю розміром $n \times m$, де n - число MFC коефіцієнтів, m – кількість часових фреймів. Як правило, довжину фрейму вибирають рівною 20 мс. Зображення спектрограми ключового слова, яка містить 13 MFC коефіцієнтів довжиною 50 часових фреймів наведено на рис. 2.

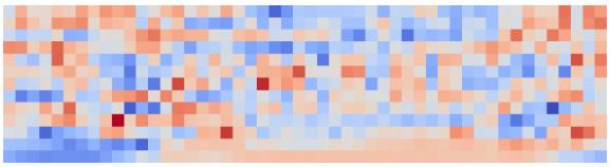


Рис. 2. Спектрограма ключового слова

2.2 Алгоритм MFCC для розпізнавання мови

Розпізнавання мовлення – це контрольоване навчальне завдання. У задачі розпізнавання мовлення входом буде аудіосигнал, і потрібно передбачити текст на основі аудіосигналу. Не можна взяти необроблений аудіосигнал як вхідний сигнал для моделі, оскільки в аудіосигналі буде багато шуму. Помічено, що вилучення функцій із аудіосигналу та використання його як вхідних даних для базової моделі дасть набагато кращу продуктивність, ніж безпосереднє розглядання необробленого аудіосигналу як вхідних даних. MFCC — це широко використовувана техніка для вилучення характеристик із аудіосигналу.

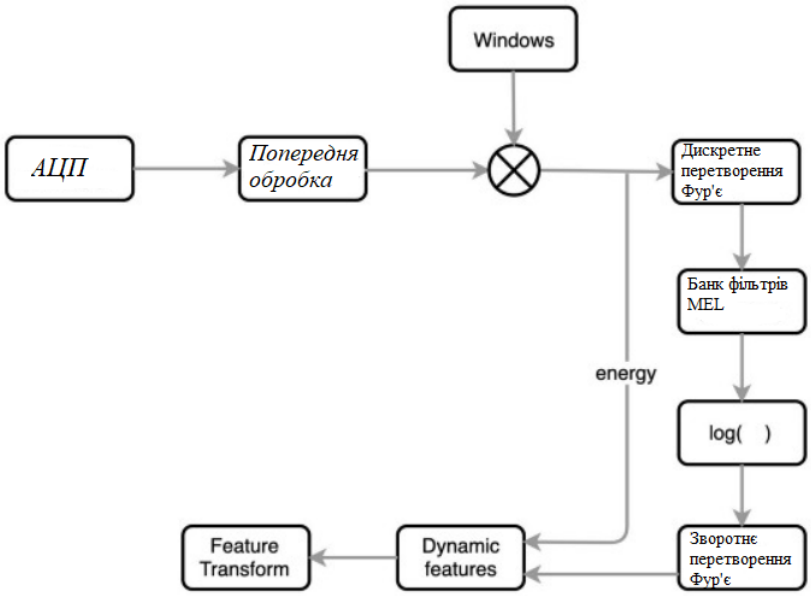


Рис. 3. Схема алгоритму MFCC для розпізнавання мови

1. A/D перетворення

A/D перетворення відбирає аудіозаписи та оцифровує вміст, тобто перетворює аналоговий сигнал у дискретний простір. Часто використовується частота дискретизації 8 або 16 кГц.

2. Переднаголос

Попереднє наголошення підвищує кількість енергії на високих частотах. Для дзвінких сегментів, таких як голосні, на нижчих частотах більше енергії, ніж на вищих. Це називається спектральним нахилом, який пов'язаний із голосовим джерелом (як голосові складки виробляють звук). Підвищення високочастотної енергії робить інформацію у вищих формантах більш доступною для акустичної моделі. Це покращує точність виявлення телефону. Що стосується людей, у нас починаються проблеми зі слухом, коли не чутно цих високочастотних звуків. Крім того, шум має високу частоту. У сфері інженерії використовується попереднє наголошення, щоб зробити систему менш сприйнятливою до шуму, який з'являється в процесі пізніше. Для деяких програм просто потрібно скасувати посилення в кінці.

Попереднє наголошення використовує фільтр для підвищення високих частот. Нижче наведено сигнал до та після того, як посилюється високочастотний сигнал.

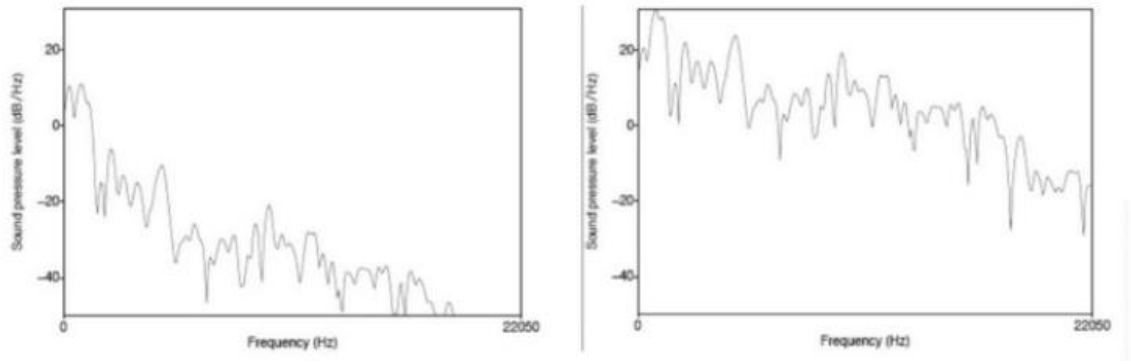


Рис. 4. Сигнал до та після того, як посилюється високочастотний сигнал.

3. Вікна

Вікна включають розрізання аудіосигналу на ковзні кадри.

Але можна просто обрізати його на краю кадру. Раптове падіння амплітуди створить багато шуму, який проявляється у високочастотному діапазоні. Щоб розділити аудіо, амплітуда повинна поступово спадати біля краю кадру.

Скажімо, w — вікно, застосоване до оригінального аудіокліпу в часовій області.

$$x[n] = w[n]s[n] \quad (9)$$

де x[n] – обрізане вікно

s[n] – оригінальний аудіозапис.

Кілька альтернатив для w – це вікно Хеммінга та вікно Ханнінга. На наступній діаграмі показано, як синусоїдальна форма сигналу буде обрізана за допомогою цих вікон. Як показано, для вікна Хеммінга та Ханнінга амплітуда падає біля краю. (Вікно Хеммінга має невеликий раптовий спад на краю, а вікно Ханнінга – ні.)

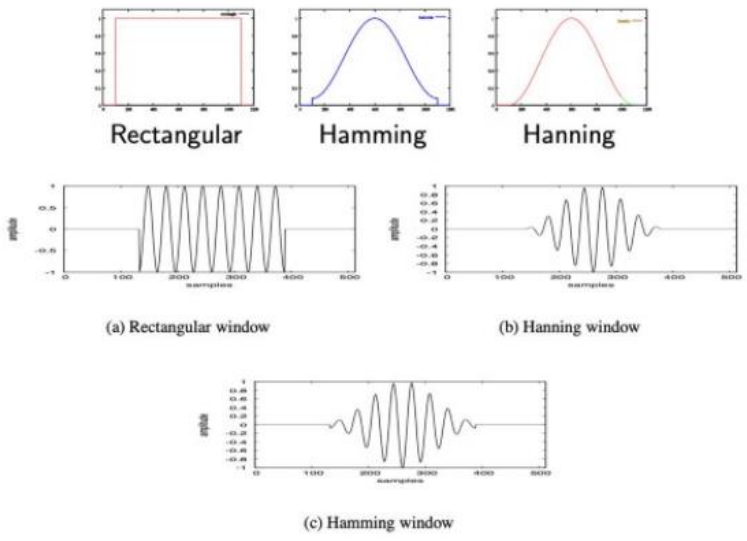


Рис. 5. Синусоїдальна форма сигналів обрізана за допомогою вікон

(a) — Прямокутне вікно, (b) —Вікно Ханнінга , (c) — Вікно Хеммінга

Відповідні рівняння для w:

Hamming ($\alpha = 0.46164$) або Hanning ($\alpha = 0.5$) вікно

$$w[n] = (1-\alpha) - \alpha\cos(\frac{2\pi n}{L-1}) \quad (10)$$

L – ширина вікна

4. Дискретне перетворення Фур'є (ДПФ)

Далі застосовується ДПФ для отримання інформації в частотній області.

$$X[k] = \sum_{n=0}^{N-1} x[n]\exp(-j\frac{2\pi}{N}kn) \quad (11)$$

5. Банк фільтрів Mel

Як згадувалося вимірювання обладнання не є таким же, як людське сприйняття слуху. Для людей сприймана гучність змінюється відповідно до частоти. Крім того, сприймана частотна роздільна здатність зменшується зі збільшенням частоти. тобто люди менш чутливі до високих частот. Діаграма показує, як шкала Мела відображає виміряну частоту на ту, яку люди сприймають в контексті роздільної здатності частоти.

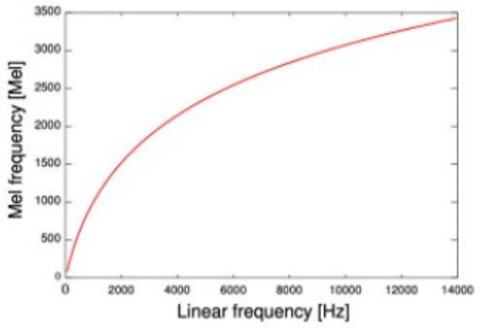


Рис. 6. Шкала Мела відображає виміряну частоту на ту, яку люди сприймають в контексті роздільної здатності частоти.

Усі ці відображення є нелінійними. Під час виділення ознак застосовуються трикутні смугові фільтри, щоб приховати частотну інформацію, щоб імітувати те, що сприймає людина.

По-перше, вихід ДПФ зводиться у квадрат. Це відображає потужність мови на кожній частоті ($x[k]^2$), і це називають спектром потужності DFT. Ці трикутні банки фільтрів застосовуються у масштабі Mel, щоб перетворити його на спектр потужності у масштабі Mel. Вихід для кожного слота спектру потужності шкали Mel представляє енергію з ряду діапазонів частот, які він охоплює. Це відображення називається Мелом Біннінгом.

Трикутна смуга пропускання ширша на високих частотах, щоб відобразити меншу чутливість людського слуху на високих частотах. Зокрема, він лінійно розподілений нижче 1000 Гц і повертається логарифмічно після цього.

Усі ці зусилля намагаються імітувати те, як базилярна мембрана людського вуха сприймає вібрацію звуків. Базилярна мембрана має близько 15 000 волосків всередині при народженні. Діаграма нижче демонструє частотну характеристику цих волосків. Отже, наведена нижче характеристика форми кривої просто апроксимується трикутниками в наборі фільтрів Mel.

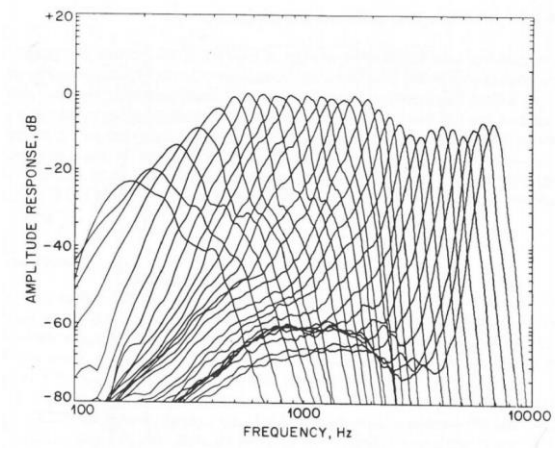


Рис. 7. Характеристика форми кривої

Ми імітуємо, як людські вуха сприймають звук.

6. Логарифм

Mel filterbank видає спектр потужності. Люди менш чутливі до невеликих змін енергії при високій енергії, ніж до невеликих змін при низькому рівні

енергії. Насправді він логарифмічний. Тож наступним кроком буде виведення журналу з виводу банку фільтрів Mel. Це також зменшує акустичні варіанти, які не мають значення для розпізнавання мовлення. Далі потрібно вирішити ще дві вимоги. По-перше, потрібно видалити інформацію F0 (тон) і зробити витягнуті функції незалежними від інших.

7. Кепструм — IDFT

Нижче наведено модель того, як створюється мова.

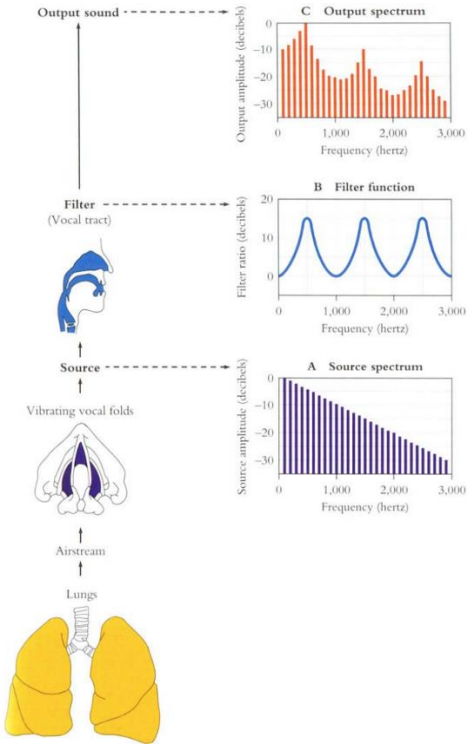


Рис. 8. Модель створення мови

8. Джерело

Наші артикуляції контролюють форму голосового тракту. Модель джерело-фільтр поєднує вібрації, створені голосовими складками, з фільтром, створеним нашими артикуляціями. Форма хвилі голосового джерела буде придушена або посилена на різних частотах формою голосового тракту.

Як було зазначено, логарифмічний спектр складається з інформації, пов’язаної з телефоном і висотою тону. Піки на другій діаграмі визначають форманти, які відрізняють телефони.

Можна застосувати зворотне перетворення Фур’є, щоб відокремити інформацію про висоту тону від формант.

Отже, для розпізнавання мовлення просто потрібні коефіцієнти в крайньому лівому куті, а інші відкидаємо. Насправді MFCC приймає лише перші 12 кепстральних значень. Є ще одна важлива властивість, пов’язана з цими 12 коефіцієнтами. Спектр логарифмічної потужності дійсний і симетричний. Його зворотне ДПФ еквівалентне дискретному косинусному перетворенню (DCT).

DCT є ортогональним перетворенням. Математично перетворення створює некорельовані ознаки. Тому функції MFCC дуже не пов’язані між собою. У ML це робить модель легшою для моделювання та навчання. Якщо ці параметри моделюються за допомогою багатовимірного розподілу Гауса, усі недіагональні значення в коваріаційній матриці будуть дорівнювати нулю.

9. Динамічні характеристики (дельта)

MFCC має 39 функцій. Доопрацьовуємо 12 і які залишилися. 13-й параметр — енергія в кожному кадрі.

У вимові важливі контекст і динамічна інформація. Артикуляції, як і замикання та відпускання, можна розпізнати за формантними переходами. Характеристика змін функцій з часом надає контекстну інформацію для телефону. Інші 13 значень обчислюють значення дельта d(t) нижче. Він вимірює зміни в функціях від попереднього кадру до наступного. Це похідна ознак першого порядку.

$$d(t) = \frac{c(t+1) - c(t-1)}{2} \quad (12)$$

Останні 13 параметрів є динамічними змінами d(t) від останнього кадру до наступного. Він діє як похідна другого порядку c(t).

Отже, 39 параметрів функцій MFCC — це 12 коефіцієнтів Кепструма плюс енергетичний термін. Тоді маємо ще 2 набори, що відповідають значенням дельта та подвійним дельта.

10. Кепстральне середнє та нормалізація дисперсії

Далі можна виконати нормалізацію функції. Нормалізуємо характеристики за допомогою їх середнього значення та ділимо його на його дисперсію. Середнє значення та дисперсія обчислюються за допомогою значення ознаки j для всіх кадрів в одному висловлюванні. Це дозволяє коригувати значення для протидії варіантам у кожному записі.

Однак, якщо аудіозапис короткий, це може бути ненадійним. Замість цього можна обчислити середнє значення та значення дисперсії на основі спікерів або навіть усього набору навчальних даних. Цей тип нормалізації функції ефективно скасовує попередній акцент, зроблений раніше. Нарешті, MFCC не дуже стійкий до шуму.

11. Перцептивне лінійне передбачення (PLP)

PLP дуже схожий на MFCC. Вмотивований сприйняттям на слух, він використовує попередній наголос однакової гучності та компресію кубічного кореня замість логарифмічної компресії.

12.Джерело

Він також використовує лінійну регресію для остаточного визначення кепстральних коефіцієнтів. PLP має трохи кращу точність і трохи кращу шумостійкість. Але також вважається, що MFCC є безпечним вибором [63].

2.3 CNN TensorFlow (Keras) для використання у вбудованих системах

Для класифікації зразків мови в роботі використано згорткову нейронну мережу. Типова згорткова мережа, наприклад LeNet-5, складається з трьох типів шарів, а саме: згорткового, об'єднаного і повнозв'язаного. Згортковий шар націлений на вивчення представлень характеристик вхідних даних. Як показано на рисунку 9, шар згортки складається з кількох ядер згортки, які використовуються для обчислення різних карт особливостей. Зокрема, кожен нейрон карти ознак з'єднаний із областю сусідніх нейронів у попередньому шарі. Таке сусідство називається рецептивним полем нейрона в попередньому шарі. Нову карту особливостей можна отримати, спершу згорнувши вхідні дані за допомогою навченого ядра, а потім застосувавши поелементну нелінійну функцію активації до згорнутих результатів. Повні карти функцій отримують за допомогою кількох різних ядер. Математично значення параметра в місці (i, j) на k-й карті об'єктів l-го шару, $z^l_{i,j,k}$, обчислюється за формулою:

$$z^l_{i,j,k} = \left(w^l_k\right)^T x^l_{i,j} + b^l_k \qquad (13)$$

де w^l_k і b^l_k є ваговим вектором і зміщенням k-го фільтра l-го шару відповідно, а $x^l_{i,j}$ є вхідним фрагментом з центром у розміщені (i, j) l-го шару. Важливо, що ядро w^l_k, яке генерує карту функцій $z^l_{i,j,k}$ є спільним. Такий механізм розподілу ваги має кілька переваг, зокрема він зменшує складність моделі та полегшити навчання нейронної мережі [64].

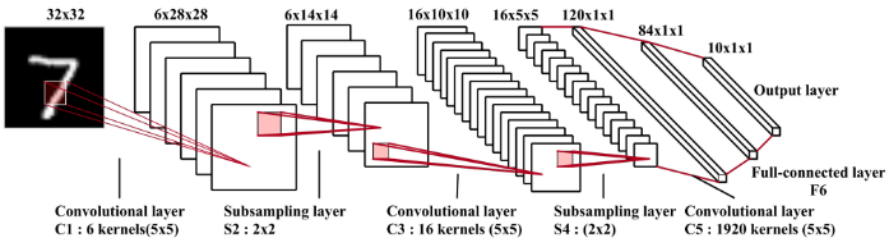


Рис. 9. Архітектура згорткової мережі на прикладі LeNet-5 [64].

Одновимірні згорткові нейронні мережі широко використовуються у додатках обробки різноманітних сигналів та мають ряд суттєвих переваг над

звичайними CNN глибокого навчання. Перш за все, компактні 1D CNN можна ефективно навчити з обмеженим набором даних 1D-сигналів, тоді як CNN глибокого навчання зазвичай потребують масивів даних великого розміру. 1D CNN можна безпосередньо застосовувати до необробленого сигналу (наприклад, струму, напруги, вібрації тощо), не вимагаючи будь-якої попередньої чи пост-обробки, такої як, виділення ознак, зменшення розмірності, зменшення шумів тощо. Крім того, завдяки простоті та компактності конфігурації таких адаптивних одновимірних CNN, які виконують лише лінійні одновимірні згортки (скалярні множення та додавання), можлива недорога апаратна реалізація в реальному часі [65]. На рис. 10 показаний приклад простої одновимірної архітектури CNN.

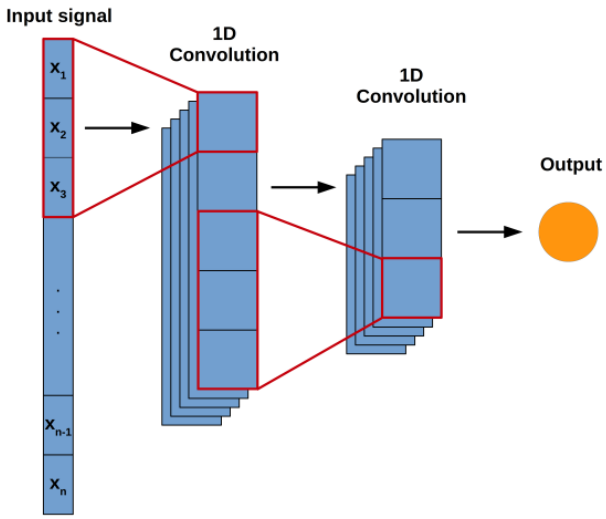


Рис. 10. Архітектура простої 1D згорткової нейромережі [66]

Кожен із етапів згортки на рис. 10 показує набір згорткових фільтрів, які можна навчати, за якими слідує операція узагальнення ознак, виділених згортковими фільтрами (Pooling). Фільтри призначені для виділення функцій високого рівня (наприклад, таких як краї та криві на зображенні) із наданого вхідного сигналу шляхом згортання набору вагових коефіцієнтів із вхідними даними та застосування нелінійної функції активації. Вихідні дані потім подаються в операцію об’єднання, яка зменшує просторовий розмір функцій, виділених згортковими фільтрами, одночасно підкреслюючи домінуючі

особливості, отримані кожним фільтром. У міру проходження вхідних даних через етапи згортки (зліва направо на рис. 10) мережа вивчає більше специфічних для даного сигналу особливостей [66].

CNN має фільтр, який переміщує зображення до створеної карти функцій на шарах згортки. Через це вікно або фільтр ваги мережі можуть ідентифікувати різні характеристики вхідного зображення. Функція активації вирішує, чи є певна функція в певному місці на зображенні. Зазвичай використовує багато фільтрів над зображенням, щоб знайти необхідні функції [67].

CNN часто називають локальною мережею, оскільки окремі одиниці, обчислені в певному місці вікна, залежать від локальної області, на яку зараз дивиться вікно. Конволюційна архітектура координується трьома основними рівнями, розташованими в структурі прямої подачі.

Згортковий рівень для виділення ознак, шари підвибірки, рівень агрегації (об’єднання) для зменшення розмірів вхідних даних і вихідних даних, які є повністю пов’язаним рівнем для прогнозування кінцевих класів [68]. Лінійний фільтр і нелінійна функція активації, один з найважливіших елементів [69]. У згортковому шарі кожна площина з’єднана з однією або декількома картами ознак попереднього шару [16]. Функція активації застосовується до результату, що отримує вихід площини. Площинним виходом є двовимірна матриця, яка називається картою ознак; ця назва виникає тому, що кожен результат згортки вказує на наявність візуальної функції в даному місці пікселя [70]. Шар згортки створює одну або більше карт функцій. Потім кожна карта ознак з’єднується точно з однією площиною в наступному шарі підвибірки (об’єднання) [69].

Спільне використання ваг і розташування має важливе значення для властивостей об’єднання, значення ознак, обчислені в різних місцях, групуються разом і представлені одним значенням, щоб мінімізувати відмінності в витягнутих ознаках уздовж частотного виміру, коли вхідні моделі

зсуваються. Це важливо, коли маємо справу з невеликими частотними зсувами, поширеними в мові, що є наслідком різної довжини вокалу.

Висновки до 2 розділу

Отже, у другому розділі дисертації були розглянуті коефіцієнти мел-частотного кепстру (MFCC), що широко використовуються в автоматичному розпізнаванні мови та мовця. Мел шкала пов’язує відчуту частоту або висоту чистого тону з його фактично виміряною частотою. Люди набагато краще розрізняють невеликі зміни висоти на низьких частотах, ніж на високих. Завдяки використанню цієї шкали Мел функції точніше відповідають тому, що чують люди.

Також, розглянутий алгоритм MFCC для розпізнавання мови та детально розглянуто його кроки. Техніка виділення ознак MFCC в основному включає вікно сигналу, застосування дискретного перетворення Фур’є, реєстрацію величини, а потім деформацію частот за шкалою Мела з подальшим застосуванням зворотного дискретного косинусного перетворення.

Розглянуто нейронну мережу CNN TensorFlow (Keras) для використання у вбудованих системах, принцип роботи даної нейронної мережі та особливості її використання. Одновимірні згорткові нейронні мережі широко використовуються у додатках обробки різноманітних сигналів та мають ряд суттєвих переваг над звичайними CNN глибокого навчання. Також, розглянуто архітектуру простоїв 1D згорткової нейромережі.

РОЗДІЛ 3 Опис програмного та апаратного забезпечення експерименту

Експериментальні дослідження виконувались у наступному порядку. Спочатку на комп’ютері створено модель нейронної мережі, виконано її тренування та завантажено в пам'ять мікроконтролера. Далі перевірено роботу моделі при різних значеннях її параметрів. Для створення моделі використано програмну платформу Edge Impulse [67], яка забезпечує машинне навчання на вбудованих пристроях для сенсорів, аудіо та комп’ютерного бачення з можливістю масштабування моделі нейронної мережі під вибране апаратне забезпечення. Такий підхід дає змогу виконувати оптимізоване машинне навчання вбудованих систем, починаючи від мікроконтролерів і закінчуючи центральними процесорами та спеціальними прискорювачами штучного інтелекту.

3.1 Опис Edge Impulse

Робота Edge Impulse ґрунтується на фреймворку Keras, який є набором функцій глибокого навчання для створення застосунків, написаних на мові програмування Python, для взаємодії з платформою машинного навчання TensorFlow. Edge Impulse дає можливість використовувати достатньо велику кількість налаштувань нейронної мережі, крім того забезпечує індикацію використання ресурсів процесора та орієнтовний час виконання задачі. Достатньо тільки вибрати необхідний тип процесора. Також на платформі Edge Impulse можна виконувати попередню обробку аудіоданих. В результаті можна завантажити код програми з тренованою моделлю мережі для вибраного мікроконтролера.

Нижче наведений алгоритм створення та налаштування проекту на платформі Edge Impulse.

1. Пристрої

Існує велика різноманітність пристроїв, які можна підключити до проекту Edge Impulse. Ці пристрої можуть допомогти зібрати набори даних для проекту, перевірити навчену модель і навіть розгорнути модель безпосередньо на платі розробки за допомогою попередньо створеної бінарної програми.

На вкладці «Пристрої» є список усіх підключених пристроїв і посібник із підключення нових пристроїв, які наразі підтримуються Edge Impulse [71].

На цьому етапі створення було підключено плату Arduino Nano 33 BLE Sense.

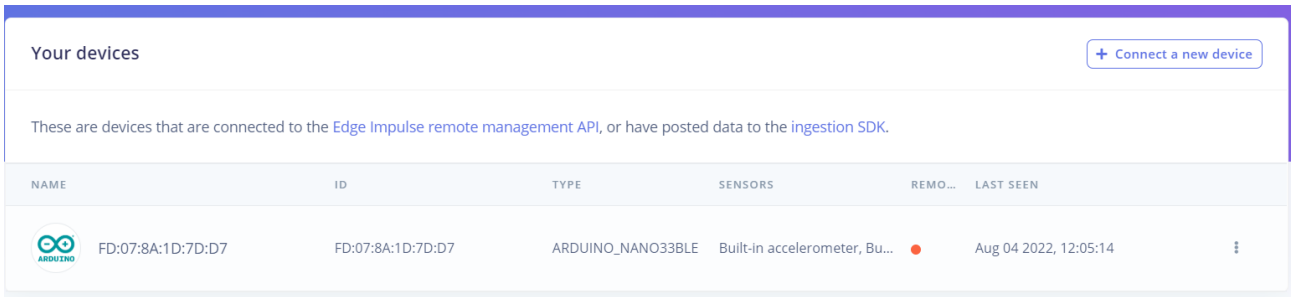


Рис. 11. Підключення плати Arduino Nano 33 BLE Sense до проекту в Edge Impulse

2. Збір даних

Усі зібрані дані для кожного проекту можна переглянути на вкладці Збір даних. Можна побачити, як дані були розділені для набору тренувань/випробувань, а також розподіл даних для кожного класу у наборі даних. Є можливість надсилати нові дані датчиків у свій проект за допомогою завантаження файлу, через WebUSB, Edge Impulse API або Edge Impulse CLI.

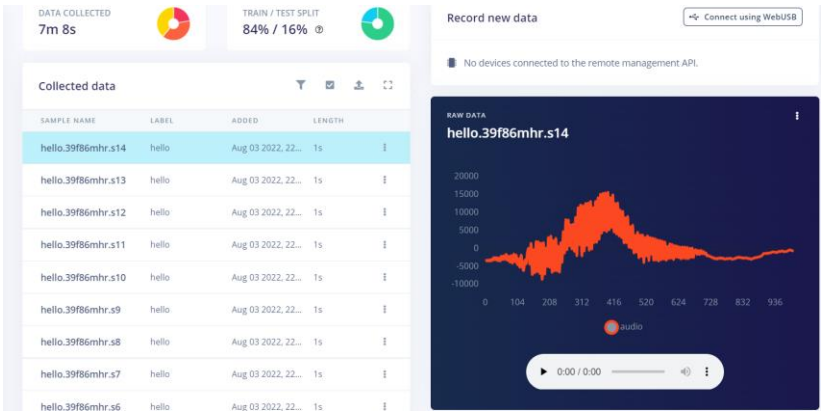


Рис. 12. Дані, що були розділені для набору тренувань/випробувань

3. Запис даних

Панель праворуч дозволяє збирати дані безпосередньо з будь-якої повністю підтримуваної платформи:

- Через WebUSB.
- Використання CLI Edge Impulse daemon.
- З Edge Impulse для Linux CLI.

WebUSB і Edge Impulse daemon працюють з будь-яким повністю підтримуваним пристроєм, завантажуючи попередньо зібрану мікропрограму Edge Impulse на плату.

4. Коефіцієнт розподілу тренування/тесту набору даних

Поділ навчання/тестування – це техніка для навчання й оцінки продуктивності алгоритмів машинного навчання. Він вказує, як дані розподіляються між навчальними та тестовими зразками. Наприклад, розподіл 80/20 вказує на те, що 80% набору даних використовується для навчання моделі, тоді як 20% використовується для тестування моделі.

У цьому розділі також показано, як розподіляються зразки даних у кожному класі, щоб запобігти незбалансованим наборам даних, які можуть внести зміщення під час навчання моделі.

5. Фільтр збору даних

Перехід до деяких категорій даних вручну може зайняти багато часу, особливо якщо є справа з великим набором даних. Фільтр збору даних дозволяє користувачеві фільтрувати зразки даних на основі певних критеріїв вибору. Це може базуватися на:

- Label (Мітка) — клас, до якого відноситься вибірка.
- Sample name (Назва зразка) — унікальний ідентифікатор зразка.
- Signature validity (Дійсність підпису)

- Enabled and disabled samples (Увімкнені та вимкнені зразки)
- Length of sample (тривалість семплу).

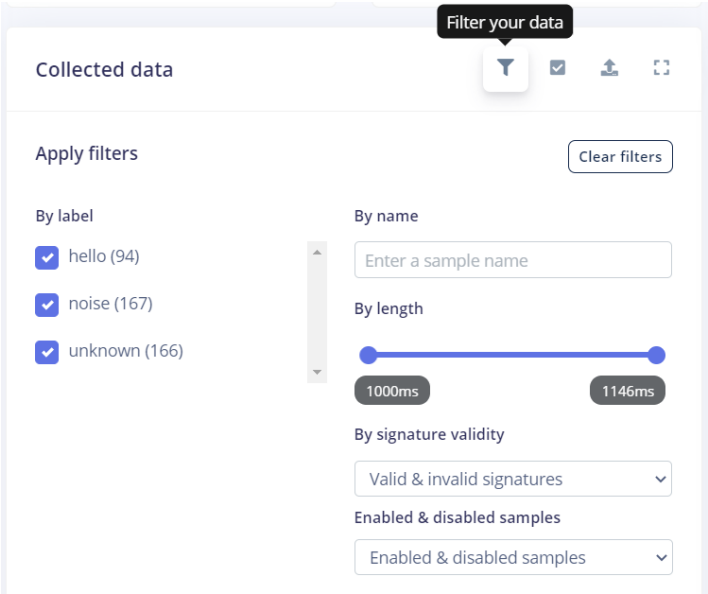


Рис. 13. Фільтр збору даних

Потім можна маніпулювати відфільтрованими зразками, редагуючи мітку, видаляючи, переміщаючи від наборів поїздів до тестового набору і навпаки, як показано на зображенні вище.

Наведені вище операції з даними також можна застосувати на рівні вибірки даних, просто перейшовши до окремої вибірки даних, натиснувши ⋮ і вибравши тип дії, яку можна виконати з конкретною вибіркою. Це може бути перейменування, редагування його мітки, вимкнення, обрізання, розділення, завантаження та навіть видалення зразка за бажанням.

6. Обрізання зразків

Щоб обрізати зразок даних, перейдіть до зразка, який потрібно обрізати, і натисніть ⋮, а потім виберіть «Обрізати зразок». Можна вказати довжину або перетягнути маркери, щоб змінити розмір вікна, а потім перемістити вікно, щоб зробити вибір.

Щоб скасувати кадрування, просто встановіть для довжини зразка велике число, і весь зразок буде вибрано знову.

7. Вибірка даних розбиття

Окрім обрізання, також можна автоматично розділяти дані. Тут можна виконувати один рух кілька разів або вимовляти ключове слово знову і знову, і події виявляються та можуть зберігатися як окремі зразки. Це полегшує дуже швидке створення високоякісного набору даних дискретних подій. Можна встановити довжину вікна, і всі події автоматично виявлятимуться. Якщо аудіодані розділяються, також можна прослухати події, клацнувши вікно, аудіопрогравач автоматично заповнюється цим конкретним розділенням.

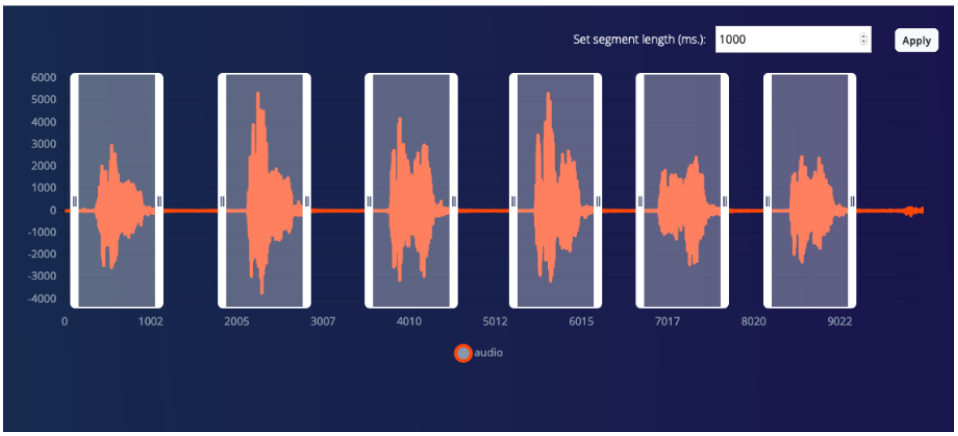


Рис. 14. Розділення даних

Зразки автоматично центруються у вікні, що може призвести до проблем на деяких моделях (нейронна мережа може дізнатися ярлик, де дані в середині вікна завжди пов’язуються з певною міткою), тому можна вибрати «Змінити зразки», щоб автоматично трохи переміщати дані.

Розділення даних, як і обрізання даних, є неруйнівним [72].

8. Створення Impulse

Після збору даних для проекту можна створити свій Impulse. Повний Impulse складатиметься з 3 основних будівельних блоків: блоку введення, блоку обробки та блоку навчання.

Цей вид є одним із найважливіших, тут створюється власний конвеєр машинного навчання.

Приклад імпульсу для класифікації руху за допомогою даних акселерометра

9. Блок введення

Блок введення вказує на тип вхідних даних, з якими навчається модель. Це можуть бути часові ряди (аудіо, вібрація, рухи) або зображення.

Часовий ряд (аудіо, вібрація, рухи)

- У полі осей введення перелічено всі осі, на які посилається навчальний набір даних
- Розмір вікна — це розмір необроблених функцій, які використовуються для навчання
- Збільшення вікна використовується для штучного створення додаткових функцій (і живлення блоку навчання додатковою інформацією)
- Частота розраховується автоматично на основі навчальних зразків. Можна змінити це значення, але наразі не можете використовувати значення, нижчі за 0,000016 (менше 1 вибірки кожні 60 с).
- Zero-pad data: додає нульові значення, якщо необроблена функція відсутня

10. Обробка блоків

Блок обробки — це, по суті, екстрактор функцій. Він складається з операцій DSP (цифрової обробки сигналів), які використовуються для виділення функцій, які вивчає модель. Ці операції відрізняються залежно від типу даних, які використовуються у проекті.

Edge Impulse зазвичай використовує зірочку, щоб позначити найбільш рекомендований блок обробки на основі вхідних даних, як показано на зображенні нижче.

У випадку, коли доступні блоки обробки не підходять для програми, можна створити власні блоки обробки та імпортувати їх у свій проект.

11.Навчальні блоки

Після додавання блоку обробки потрібно додати блок навчання, щоб завершити імпульс. Навчальний блок — це нейронна мережа, що навчається на даних.

Навчальні блоки відрізняються залежно від того, що потрібно зробити з моделлю, і типу даних у навчальному наборі даних. Алгоритми включають: класифікацію, регресію, виявлення аномалій, навчання передачі зображень, виявлення ключових слів або виявлення об’єктів. Також можна створити власний навчальний блок (корпоративна функція) [73].

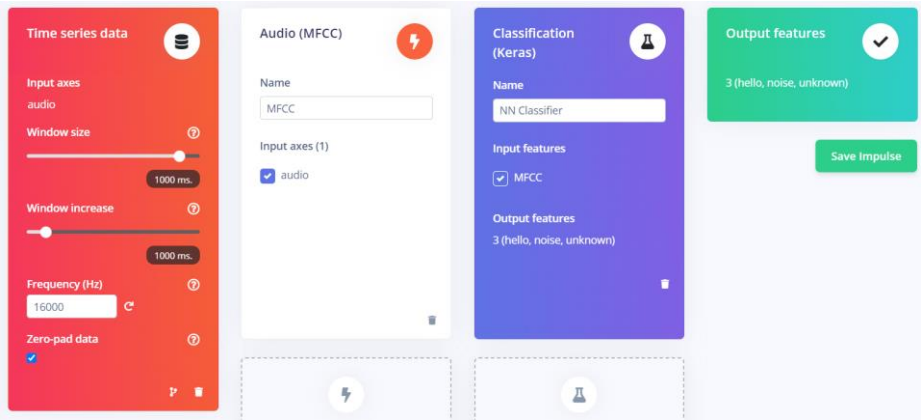


Рис. 15. Обрані блоки для нейронної мережі

12.Обробка блоків

Отримання значущих функцій із даних має вирішальне значення для створення невеликих і надійних моделей машинного навчання, а в Edge Impulse це робиться за допомогою блоків обробки. Існує низка блоків обробки для загальних даних датчиків (таких як вібрація та звук):

- Необроблені дані
- Розплющити
- Зображення

- Спектральні особливості
- Спектрограма
- Аудіо MFE
- Аудіо MFCC
- Audio Syntiant
- IMU Syntiant

У роботі використовувався блок Аудіо MFCC

Блок Audio MFCC отримує коефіцієнти з аудіосигналу. Використовується нелінійна шкала під назвою Мел-шкала. Це еталонний блок для розпізнавання мовлення, а також може добре працювати в деяких випадках використання нелюдського голосу.

Параметри аудіо MFCC

- Кепстральні коефіцієнти Mel Frequency
- Кількість коефіцієнтів: кількість кепстральних коефіцієнтів, які потрібно зберегти після застосування дискретного косинусного перетворення
- Довжина кадру: довжина кожного кадру в секундах
- Крок кадру: крок між послідовними кадрами в секундах
- Номер фільтра: кількість трикутних фільтрів, застосованих до спектрограми
- Довжина FFT: розмір FFT
- Низька частота: край нижньої смуги наборів фільтрів шкали Mel
- Висока частота: найвищий край смуги наборів фільтрів шкали Mel
- Розмір вікна: розмір ковзного вікна для нормалізації локального кепстрального середнього. Розмір вікон повинен бути непарним [74].

13.Навчальні блоки

Після вилучення значущих функцій із необробленого сигналу за допомогою обробки сигналу тепер можна навчити модель за допомогою навчального блоку:

- Classification (Keras).
- Regression (Keras).
- Anomaly Detection (K-means).
- Image Classification (using Transfer Learning).
- Keyword Spotting (using Transfer Learning).
- Object Detection (using MobileNetV2 SSD FPN).
- Object Detection (using FOMO)

Був обраний блок Classification (Keras).

Основна ідея блоку Classification (Keras) полягає в тому, що класифікатор нейронної мережі візьме деякі вхідні дані та виведе оцінку ймовірності, яка вказує, наскільки ймовірно, що вхідні дані належать до певного класу.

Нейронна мережа складається з ряду шарів, кожен з яких складається з певної кількості нейронів. Нейрони першого шару з’єднані з нейронами другого шару і так далі. Вага зв'язку між двома нейронами в шарі визначається випадковим чином на початку процесу навчання. Потім нейронній мережі надається набір навчальних даних, який є набором прикладів, які вона повинна передбачити. Вихідні дані мережі порівнюються з правильною відповіддю, і на основі результатів коригуються ваги зв’язків між нейронами в шарі. Цей процес повторюється кілька разів, поки мережа не навчиться передбачати правильну відповідь для навчальних даних.

Конкретне розташування шарів називається архітектурою, і різні архітектури корисні для різних завдань. Таким чином, після багатьох ітерацій,

нейронна мережа навчається; і згодом стане набагато кращим у прогнозуванні нових даних.

14.Налаштування нейронної мережі

- Кількість циклів навчання: кожен раз, коли алгоритм навчання виконує один повний прохід через усі навчальні дані зі зворотним поширенням і оновлює параметри моделі по ходу, це називається епохою або циклом навчання.
- Швидкість навчання: швидкість навчання контролює, наскільки внутрішні параметри моделі оновлюються під час кожного кроку процесу навчання. Або також можна побачити це як швидкість навчання нейронної мережі. Якщо мережа швидко переповнюється, можна зменшити швидкість навчання
- Розмір набору для перевірки: відсоток набору для навчання, відокремленого для перевірки, правильне значення за умовчанням становить 20%.
- Набір даних автоматичного балансу. Змішування більше копій даних із незвичайних класів. Це може допомогти зробити модель більш надійною проти переобладнання, якщо є мало даних для деяких класів.

15.Архітектура нейронної мережі

Залежно від типу проекту можна запропонувати вибрати між різними налаштуваннями архітектури, щоб допомогти розпочати роботу.

Архітектура нейронної мережі приймає як вхідні дані, отримані від вас, і передає їх на кожен рівень архітектури. У випадку класифікації останнім використаним шаром є шар softmax. Саме цей останній шар дає ймовірність приналежності до одного з класів.

16.Продуктивність моделі

У цьому розділі наведено огляд продуктивності моделі та допоможе оцінити її. Це може допомогти визначити, чи здатна модель задовольнити потреби, чи потрібно перевірити інші гіперпараметри та архітектури.

З останніх тренувальних виступів можна отримати свою точність перевірки та втрати.

Матриця є одним із найкорисніших інструментів для оцінки моделі. він зводить у таблицю всі правильні та неправильні відповіді, які дає модель на основі набору даних. Мітки збоку відповідають фактичним міткам у кожному зразку, а мітки вгорі відповідають прогнозованим міткам із моделі.

Провідник функцій, як і в переглядах блоків обробки, вказав просторовий розподіл вхідних об’єктів. На цій сторінці можна візуалізувати, які класифіковані правильно, а які ні.

Продуктивність на пристрої: на основі цілі, яку обрано на сторінці «Інформаційна панель», платформа Edge Impulse виводить оцінки для часу визначення, максимального використання оперативної пам’яті та використання флеш-пам’яті. Це допоможе переконатися, що модель зможе працювати на пристрої на основі його обмежень [75].

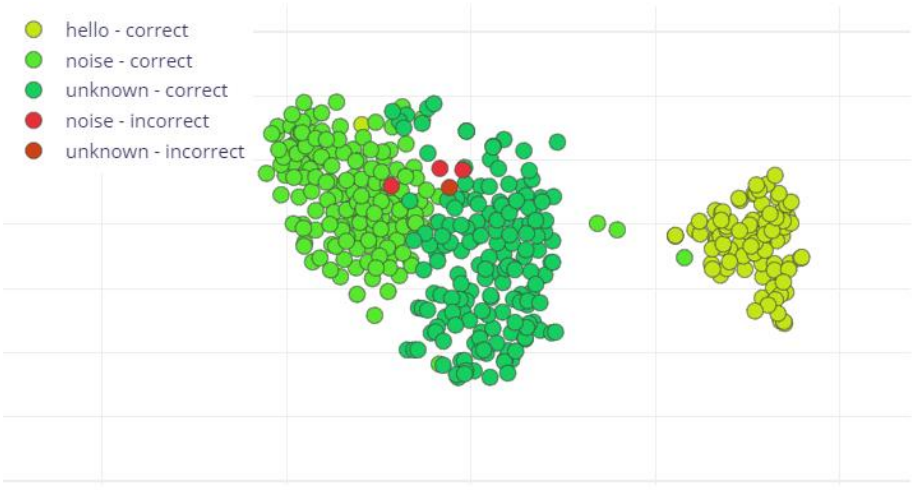


Рис. 16. Загальний вигляд повного набору даних

17.Встановлення моделі

Після навчання та перевірки моделі тепер можна розгорнути її на будь-якому пристрої. Завдяки цьому модель працює без підключення до Інтернету, мінімізує затримку та працює з мінімальним енергоспоживанням.

Сторінка розгортання містить різноманітні варіанти розгортання, які можна вибрати залежно від цільового пристрою. Незалежно від того, використовується чи ні, Edge Impulse надає параметри розгортання через бібліотеку C++, за допомогою якої можна розгортати модель на будь-яких цілях (за умови, що ціль має достатньо обчислювальних ресурсів для виконання завдання).

Нижче наведено 5 основних категорій варіантів розгортання, які зараз підтримуються Edge Impulse:

- Встановити як настроювану бібліотеку
- Встановити як попередньо зібрану мікропрограму для повністю підтримуваних плат розробки
- Запускайте безпосередньо на телефоні чи комп’ютері
- Використовуйте Edge Impulse для Linux для цілей Linux
- Створення спеціального блоку розгортання (функція Enterprise)

Встановлення як настроюваної бібліотеки

Цей варіант розгортання дає змогу перетворити імпульс на повністю оптимізований вихідний код, який можна додатково налаштувати та інтегрувати з програмою. Цей параметр підтримує такі бібліотеки:

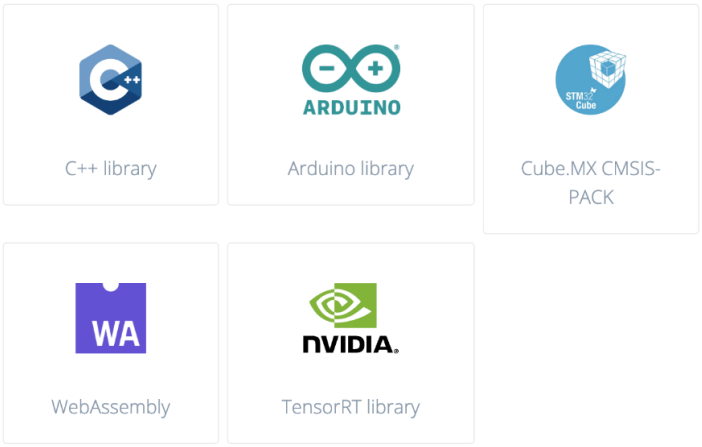


Рис. 17. Бібліотеки, що підтримуються Edge Impulse

Було обрано бібліотеку Arduino

18.Встановити як попередньо зібрану мікропрограму

Для цього варіанту можна використовувати готовий двійковий файл для плати розробки, який об’єднує блоки обробки сигналів, блоки конфігурації та навчання в єдиний пакет. Ця опція наразі доступна лише для повністю підтримуваних плат розробки, як показано на зображенні нижче:

Рис. 18. Плати, що підтримуються Edge Impulse[76]

Було обрано плату Arduino Nano 33 BLE Sense.

3.2 Опис плати Arduino Nano 33 BLE Sense

Після створення та навчання моделі нейронної мережі згенеровано програмний код під плату розробки Arduino Nano 33 BLE Sense [68]. Плата містить 32-розрядний ARM® Cortex™-M4 мікроконтролер nRF52840, що працює на частоті 64 МГц.

Плата має декілька датчиків, а саме: 9-осьовий інерційний вимірювальний пристрій, датчики температури, тиску, вологості, світла, кольору та жестів. Для введення аудіо сигналів доступний мікрофон. Основні електричні характеристики плати наступні:

- Частота тактового генератора - 64МГц
- Об’єм пам’яті програм (FLASH) - 1МБ
- Об’єм оперативної пам’яті (SRAM) - 256КБ
- АЦП - 12 біт, 200 000 значень в секунду

Зовнішній вигляд плати Arduino Nano 33 BLE Sense наведено на рис.19.

Рис. 19. Плата Arduino Nano 33 BLE Sense

Вбудовані датчики в Arduino Nano 33 BLE Sense:

LSM9DS1: 9-ти осевий інерційний вимірювальний пристрій. Мікросхема має 3D цифровий датчик лінійного прискорення, 3D цифровий датчик кутової швидкості та 3D цифровий магнітний датчик. Використовується для

вимірювання лінійного прискорення, кутової швидкості та напруженості магнітного поля відповідно по всіх трьох осях. Максимальна частота оновлення для акселерометра та гіроскопа становить 952 Гц і до 80 Гц для магнітометра.

HTS221: ємнісний цифровий датчик вологості та температури HTS221 — це надкомпактний датчик відносної вологості та температури. Він постачається з чутливим елементом і ASIC (спеціальною інтегральною схемою) зі змішаним сигналом, щоб надавати інформацію про зовнішнє середовище через цифрові послідовні інтерфейси.

APDS9960: чіп має розширене визначення жестів, визначення наближення, цифрове визначення навколишнього освітлення (ALS) і визначення кольору (RGBC). Мікросхема містить ІЧ-світлодіод і заводський калібрований світлодіодний драйвер для сумісності з наявними посадочними місцями. Виявлення жестів використовує набір із чотирьох спрямованих фотодіодів для визначення відбитої ІЧ-енергії для перетворення фізичної інформації про рух (тобто швидкості, напрямку та відстані) у цифрову інформацію.

LPS22HB: надкомпактний п’єзорезистивний датчик абсолютного тиску, який функціонує як цифровий вихідний барометр. Чіп поставляється з чутливим елементом і інтерфейсом IC, який спілкується через інтерфейс I2C або SPI з платою Nano.

MP34DT05-A: надкомпактний, малопотужний, всеспрямований цифровий мікрофон MEMS, побудований з ємнісним елементом і інтерфейсом IC. Мікрофон здатний виявляти акустичні хвилі, він виготовлений за допомогою спеціального процесу мікрообробки кремнію, призначеного для створення аудіодатчика.

Розміщення виводів Arduino Nano 33 BLE Sense наведено на рис. 20

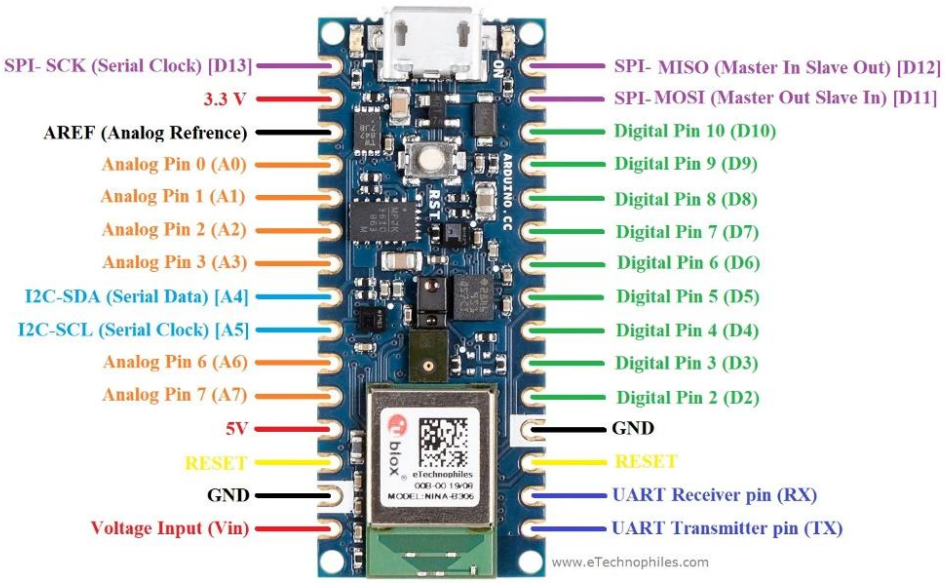


Рис. 20. Розміщення виводівплати Arduino Nano 33 BLE Sense

Короткий опис виводів плати Arduino Nano 33 BLE Sense:

- Vin — напруга живлення плати.
- Цифрові виводи — 14 контактів цифрового введення/виведення. Цифрові контакти Arduino можуть зчитувати лише два стани: коли є сигнал напруги та коли сигналу немає. Цей тип вхідних даних зазвичай називають цифровими (або двійковими), а ці стани називаються ВИСОКИЙ і НИЗЬКИЙ або 1 і 0.
- Виводи ШІМ — усі цифрові виводи на Arduino Nano 33 BLE sense є контактами з підтримкою ШІМ (широтно-імпульсної модуляції), які пронумеровані від D0 до D13. Кожен із цих цифрових контактів може генерувати сигнал широтно-імпульсної модуляції з роздільною здатністю 23 біта.
- Аналогові контакти — плата має 8 контактів АЦП, пронумерованих від A0 до A7, тоді як аналоговий вихід може бути досягнутий лише за допомогою контактів ШІМ. Це означає, що можна підключити до

плати до 8 датчиків аналогового входу. Кожен з аналогових виводів має вбудований АЦП з роздільною здатністю 2^{12} біт.

- Виводи I2C (Inter-Integrated Circuits) — двопровідний послідовний протокол зв'язку. Інтерфейс I2C використовує два контакти для зв’язку, надсилання та отримання даних: контакт послідовного синхронізації (SCL) і контакт послідовного передачі даних (SDA).
    - SCL (Serial Clock). Він визначається як лінія, яка передає дані годинника. Він використовується для синхронізації передачі даних між двома пристроями. Послідовний годинник генерується головним пристроєм.
    - SDA (Serial Data). Він визначається як лінія, яка використовується підлеглим і головним для надсилання та отримання даних. Ось чому його називають лінією даних, тоді як SCL називають лінією синхронізації.

Виводи I2C на платі: A4(SDA), A5(SCL)

- Виводи SPI (Serial Peripheral Interface) — використовуються мікроконтролерами для швидкого зв’язку з одним або кількома периферійними пристроями. Є три загальні контакти для всіх периферійних пристроїв:
    - SCK (Serial Clock). Цей висновок генерує тактові імпульси, які використовуються для синхронізації передачі даних.
    - MISO (Master Input/Slave Output). Ця лінія даних у контакті MISO використовується для надсилання даних головному.
    - MOSI (Master Output/Slave Input). Ця лінія використовується для надсилання даних до підлеглих/периферійних пристроїв.

Контакти SPI на платі: D13(SCK), D12(MISO), D11(MOSI)

- Виводи TXD і RXD використовуються для послідовного зв'язку. TXD використовується для передачі даних, а RXD використовується для отримання даних під час послідовного зв’язку. Це також символізує успішний потік даних від комп’ютера до плати. Виводи UART: D0(TX), D1(RX)

- Інші виводи: 3,3 В — контакт 3,3 В працює як вихідна регульована напруга 3,3 В [77].

3.2.1 Опис мікроконтролера nRF52840

Мікроконтролер nRF52840 —однокристальна система для сучасних бездротових додатків, яка спроектована для використання у складі систем Інтернету речей. Мікроконтролер підтримує основні функції Bluetooth® 5 і використовує переваги збільшених можливостей Bluetooth 5, включаючи режими великої дальності та високої пропускної здатності. Мікроконтролер nRF52840 забезпечує найкращий у своєму класі захист для серії CortexTM-M із вбудованим криптографічним прискорювачем ARM® CryptoCell.

nRF52840 використовує ту саму апаратну та програмну архітектуру, що й існуючі SoC серії nRF52. Його ядром є процесор ARM Cortex-M4F, який дозволяє швидше й ефективніше обчислювати складні функції для DSP і тих, що потребують математики з плаваючою комою. Має великий обсяг пам’яті як флеш-пам’яті, так і оперативної пам’яті, 1 МБ/256 КБ відповідно.

Мікроконтролер має повношвидкісний (12 Мбіт/с) контролер USB 2.0. Широкий асортимент периферійних пристроїв доступний з низкою високопродуктивних цифрових інтерфейсів, таких як високошвидкісний SPI (32 МГц), що забезпечує пряме підключення до дисплеїв і зовнішніх джерел пам'яті. nRF52840 може працювати з напругою живлення від +5,5 В до 1,7 В, що дозволяє пряме живлення від акумуляторних батарей і джерел USB.

Ключові особливості мікроконтролера:

• Мультипротокольне радіо з підтримкою Bluetooth 5

• Підтримка швидкості передачі даних Bluetooth 5: 2 МБ, 1 МБ, 500 КБ, 125 КБ

• 32-розрядний ARM Cortex-M4F на 64 МГц

• Висока швидкість передачі даних 2 Мбіт/с

• Бюджет зв’язку до 111 дБ для режиму дальньої дії Bluetooth

• Повношвидкісний USB-контролер 12 Мбіт/с

• NFC-A на кристалі

• Стеки програмного забезпечення доступні для завантаження

• Розробка додатків незалежно від стеку протоколів

• Програмована вихідна потужність від +8 дБм до -20 дБм

• Чутливість -96 дБм для Bluetooth з низьким енергоспоживанням

• В ефірі сумісний із серіями nRF51, nRF24L і nRF24AP

• Криптографічний прискорювач ARM Cryptocell CC310

• RSSI

• Широкий діапазон напруги живлення від +5,5 до 1,7 В

• Повний вибір інтерфейсів SPI/UART/PWM

• Програмований периферійний інтерфейс - PPI

• Високошвидкісний інтерфейс SPI 32 МГц

• Інтерфейс Quad SPI 32MHz

• EasyDMA для всіх цифрових інтерфейсів

• RAM відображено FIFO за допомогою EasyDMA

• АЦП 12 біт/200 КБ SPS

• 128-розрядний співпроцесор AES/ECB/CCM/AAR

• Односторонній антенний вихід (балун на кристалі)

• Вбудований понижуючий перетворювач DC-DC

• Квадратурний демодулятор

• Регульоване живлення зовнішніх компонентів до 25 мА [78]

3.2.2 Опис мікрофона MP34DT05-A

MP34DT05-A — це надкомпактний, малопотужний, всеспрямований цифровий мікрофон MEMS, побудований з ємнісним чутливим елементом і інтерфейсним блоком.

Чутливий елемент, здатний виявляти акустичні хвилі, виготовляється за допомогою спеціального процесу мікрообробки кремнію, призначеного для виробництва аудіодатчиків.

Інтерфейсний блок забезпечеє цифровий сигнал у форматі PDM.

MP34DT05-A — це цифровий мікрофон із низьким рівнем спотворень із співвідношенням сигнал/шум 64 дБ і чутливістю –26 дБFS ±3 дБ.

MP34DT05-A доступний у корпусі з верхнім портом, SMD-сумісним, EMI-екранованим корпусом і гарантовано працює в розширеному діапазоні температур від -40 °C до +85 °C.

Особливості мыкрофону MP34DT05-A:

• Одна напруга живлення

• Низьке енергоспоживання

• AOP = 122,5 dBSPL

• Співвідношення сигнал/шум 64 дБ

• Всеспрямована чутливість

• –26 dBFS ±3 dB чутливість

• Вихід PDM

• Корпус HCLGA

– Конструкція з верхнім портом

– SMD-сумісність

– Захищений від електромагнітних перешкод

– Відповідає вимогам ECOPACK, RoHS та “Green” [79].

3.3 Опис експерименту

Для проведення експерименту створено три групи даних з назвами "hello", "unknown", "noise" і ключовим словом «hello». Група "hello" містить 94 зразки слова «привіт» англійською мовою, вимовлених жіночим голосом. Група "unknown" містить 167 зразків інших слів, вимовлених як жіночим, так і чоловічим голосами. Група " noise" містить 166 зразків шумів та випадкових звуків. Згідно з рекомендації Edge Impulse 80% зразків з кожної з груп даних було використано для тренування моделі нейронної мережі, відповідно 20% зразків для перевірки. В результаті платформою Edge Impulse згенеровано код програми, який містить початкові налаштування, алгоритми обробки звуку, функції обчислення MFCC та модель 1D згорткової нейронної мережі. Перед завантаженням в мікроконтролер код доопрацьовано для підвищення стабільності процесу розпізнавання: додано можливість гнучкої синхронізації

початку вимовлення ключового слова з початком інтервалу його онлайн обробки.

В першій частині експерименту досліджено вплив кількості MFC коефіцієнтів на якість розпізнавання голосової команди і, відповідно об'єм пам'яті програм (FLASH) та оперативної пам'яті (RAM), які використані для зберігання коду 1D згорткової нейронної мережі та роботи з даними. Можна припустити, що чим більше коефіцієнтів має бути обчислено, а потім оброблено мережею в процесі розпізнавання, тим більше має бути задіяно пам'яті і тим більше буде час розпізнавання звукового зразка. Тому для аналізу використання пам'яті мікроконтролера будувались і використовувались моделі 1D згорткової мережі для наступного числа коефіцієнтів: 12, 13, 15, 17. Значення реальних затрат FLASH та RAM пам'яті отримано після компіляції програми в середовищі Arduino. Значення витраченого часу обчислювалось мікроконтролером и виводилось в термінальне вікно середовища розробки Arduino разом зі значеннями точності визначення голосу.

В другій частині експерименту досліджувалась залежність якості розпізнавання голосової команди від типу вибраної згорткової нейронної мережі, а саме 1D CNN або 2D CNN для 12 та 13 MFC коефіцієнтів.

Аналіз результатів експерименту

В таблиці 1 наведені значення точності визначення ключового слова для 20-ти спроб 1D згорткової мережі, середнє значення точності, часу обробки звукового зразка, об'єму використаної оперативної пам'яті (в байтах) та об'єму використаної пам'яті програм (в байтах) в залежності від кількості MFC коефіцієнтів. Також в таблиці наведено оціночні значення часу обробки зразка голосу, часу розпізнавання, об'ємів необхідної FLASH та RAM пам'яті, розраховані Edge Impulse для вибраного типу мікроконтролера або мікропроцесора.

Аналіз результатів показує, що зі збільшенням кількості MFC коефіцієнтів з 12 до 17, а відповідно і точності розпізнавання ключового слова, об'єм пам'яті програм, зайнятої кодом, зростає на 480 байт (менше 1%). Для мікроконтролера nRF52840 це не є суттєвим збільшенням. Об'єм використаної оперативної пам'яті в процесі експерименту не змінювався. Хоча час обчислення точності визначення кодового слова збільшився всього на 14 мс (менше 5%) зі збільшенням кількості MFC коефіцієнтів, процедура обчислення є достатньо тривалою (приблизно 0,3 с) в порівнянні з довжиною звукового зразка в 1с. Це може бути певним обмеженням при обробці звукового сигналу 32-х бітними мікроконтролерами. Для аналізу фраз або речень необхідно використовувати більш потужні мікроконтролери або мікропроцесори.

Таблиця 1 Залежність точності розпізнавання від кількості MFCC для 1D згорткової мережі

Номер експерименту	**Кількість коефіцієнтів**			
	12	***13***	***15***	***17***
1	0,93915	0,94531	0,81707	0,99974
2	0,67989	0,98438	0,97485	0,99783
3	0,65780	0,87109	0,97858	0,99925
4	0,74887	0,99609	0,99259	0,99946
5	0,49337	0,92188	0,98968	0,90597
6	0,92425	0,96484	0,96796	0,99969
7	0,59760	0,99609	0,89386	0,99986
8	0,54474	0,99609	0,99286	0,99933
9	0,51489	0,99609	0,99861	0,99971
10	0,50484	0,99609	0,93233	0,99997
11	0,75275	0,97656	0,98724	0,99933
12	0,59564	0,89062	0,99282	0,99992
13	0,63440	0,99219	0,92486	0,9021

14	0,82102	0,99609	0,9768	0,99769
15	0,92104	0,70312	0,96122	0,99934
16	0,78673	0,99609	0,99618	0,99983
17	0,85399	0,99609	0,98708	0,99497
18	0,56973	0,99609	0,98177	0,99625
19	0,82023	0,98438	0,99645	0,99998
20	0,91910	0,99609	0,98598	0,99998
Середнє значення	0,71400	0,95976	0,96644	0,98951
FLASH, Байт	171272	171368	171560	171752
RAM, Байт	52216	52216	52216	52216
Час, мс	283	283	290	297
Оціночний час обробки звуку, мс	429	437	452	465
Оціночна RAM для MFCC, КБ	23	23	24	24
Оціночний час розпізнавання, мс	41	13	20	21
Оціночна RAM для CNN, КБ	5,0	5,0	5,3	5,5
Оціночний об’єм FLASH, КБ	34,5	34,7	34,5	34,6

На рисунку 21 наведено залежність точності визначення ключового слова від кількості MFC коефіцієнтів. Як видно з рисунка, точність визначення ключового слова значно зростає коли кількість коефіцієнтів складає 13 і більше, що добре корелюється з загальноприйнятою для використання кількістю (в межах 10-20).

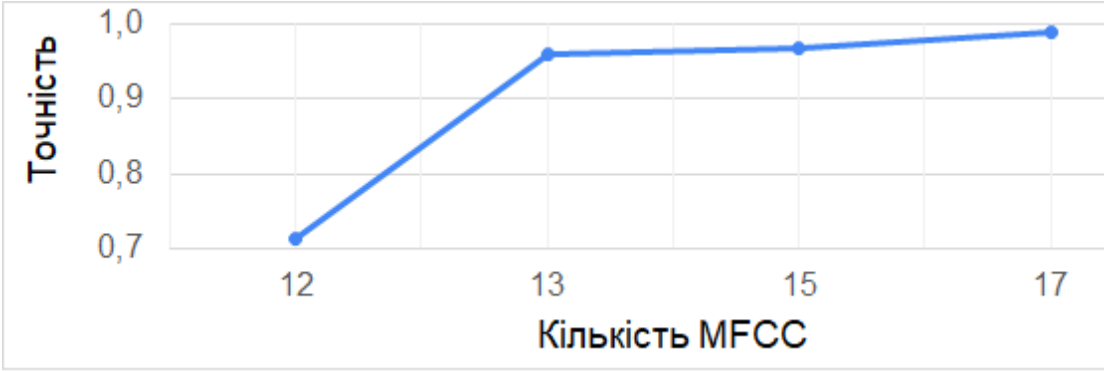


Рис. 21. Залежність точності визначення ключового слова від кількості MFC коефіцієнтів

Порівнюючи оціночні значення параметрів, розрахованих Edge Impulse, з отриманими після компіляції та в ході експерименту можна зробити висновок, що розрахунковий час майже вдвічі перевищує отриманий експериментально. Витрати FLASH та RAM пам'яті майже не міняються при зміні кількості MFC коефіцієнтів. П'ятикратне перевищення абсолютного значення реальних затрат пам'яті над розрахованими можна пояснити тим Edge Impulse не враховує розмір коду завантажувача програми.

Результати другої частини експерименту, а саме, порівняння точності визначення ключового слова в залежності від типу згорткової нейромережі (1D або 2D) для 12 та 13 MFC коефіцієнтів представлені у таблиці 2.

ТАБЛИЦЯ 2 ПОРІВНЯННЯ ТОЧНОСТІ 1D ТА 2D НЕЙРОМЕРЕЖ

Номер експерименту	**Кількість коефіцієнтів**			
	12		**13**	
	1D	***2D***	***1D***	***2D***
1	0,93915	0,9983	0,94531	0,96578
2	0,67989	0,9484	0,98438	0,99747
3	0,65780	0,9967	0,87109	0,96232
4	0,74887	0,8787	0,99609	0,96722
5	0,49337	0,9844	0,92188	0,99725
6	0,92425	0,9060	0,96484	0,99704
7	0,59760	0,9978	0,99609	0,99951
8	0,54474	0,9698	0,99609	0,86226
9	0,51489	0,9382	0,99609	0,91124
10	0,50484	0,9847	0,99609	0,99942
11	0,75275	0,9956	0,97656	0,99837

12	0,59564	0,9547	0,89062	0,99813
13	0,63440	0,9550	0,99219	0,97548
14	0,82102	0,9928	0,99609	0,99986
15	0,92104	0,9831	0,70312	0,99999
16	0,78673	0,9755	0,99609	0,95702
17	0,85399	0,9674	0,99609	0,98774
18	0,56973	0,9990	0,99609	0,99995
19	0,82023	0,9349	0,98438	0,99997
20	0,91910	0,9983	0,99609	0,91524
Середнє значення	0,71400	0,9680	0,95976	0,974563
FLASH, Б	171272	180456	171368	180456
RAM, Б	52216	51552	52216	51552
Час, мс	283	281	283	281

Результати порівняння показують перевагу 2D мережі у точності визначення ключового слова як для 12, так і для 13 MFC коефіцієнтів. Особливо це помітно для випадку з 12-ма коефіцієнтами, де точність підвищилась з 0,7 до 0,97. Однак при цьому об’єм використаної FLASH пам’яті збільшився на 5%. Об’єм використаної RAM пам’яті у випадку 2D мережі дещо зменшився. Час обробки зразка голосу для обох типів мереж є практично однаковим.

Висновки до третього розділу

За результатами експериментальних досліджень можна констатувати той факт, що обчислювальних ресурсів 32-х бітних мікроконтролерів цілком достатньо для розпізнавання голосових команд з можливістю попередньої цифрової обробки звукового сигналу, зокрема, використання мел-частотних кепстральних коефіцієнтів. Вибір кількості коефіцієнтів не впливає значним чином на об’єм задіяної FLASH та RAM пам’яті мікроконтролера nRF52840.

Використання для розпізнавання зразка голосу одновимірної згорткової нейромережі у проведеному експерименті забезпечує економію приблизно 5%

пам’яті. Якість розпізнавання ключового слова при кількості MFC коефіцієнтів 12 складає приблизно 0,7. Для 17-ти MFC коефіцієнтів якість розпізнавання становить вже 0,97. Таким чином, 1D згорткової нейромережі мають певні переваги у мікроконтролерних застосунках для обробки та розпізнавання голосу.

Обмеженням розглянутого варіанту розпізнавання голосу на мікроконтролері є достатньо довгий час обробки звукового зразка (приблизно 0,3 с) при тривалості самого зразка в1с, що можна пояснити достатньо низькою тактовою частотою в 64 МГц. Збільшення тактової частоти дозволить зменшити час обчислень.

ВИСНОВКИ

У роботі були розглянуті методи розпізнавання голосу, зокрема приховані марківські моделі, розпізнавання мовлення на основі динамічного викривлення часу (DTW), нейронні мережі, глибокі прямі та рекурентні нейронні мережі, наскрізне автоматичне розпізнавання мовлення. Також, були розглянуті сучасні системи розпізнавання мовлення, конструкція апаратного забезпечення найпростішої системи безпеки розпізнавання голосу. Були розглянуті датчики розпізнавання мовлення, зокрема ультразвуковий датчик та фізіологічний сенсор. Розглянуто використання розпізнавання голосу, основні переваги та недоліки використання розпізнавання мовлення у життєдіяльності людей.

Були розглянуті коефіцієнти мел-частотного кепстру (MFCC). Мел шкала пов’язує відчуту частоту або висоту чистого тону з його фактично виміряною частотою. Люди набагато краще розрізняють невеликі зміни висоти на низьких частотах, ніж на високих. Завдяки використанню цієї шкали Мел функції точніше відповідають тому, що чують люди.

Також, розглянутий алгоритм MFCC для розпізнавання мови та детально розглянуто його кроки. Розглянуто нейронну мережу CNN TensorFlow (Keras) для використання у вбудованих системах, принцип роботи даної нейронної мережі та особливості її використання. Також, розглянуто архітектуру простоїв 1D згорткової нейромережі.

У роботі проаналізовано вплив параметрів обробки голосу та архітектури нейронної мережі на ступінь використання ресурсів мікроконтролера. Для цього була створена база даних зразків ключового слова, зразків інших слів і голосів, зразків шумів. Оцінювалась ймовірність розпізнавання ключового слова серед інших слів і шумів, залежність обсягу використовуваної пам'яті від мікроконтролера та встановлено час прийняття рішення від кількості коефіцієнтів MFC, а також встановлено залежність обсягу використаної пам’яті

мікроконтролера та часу прийняття рішення від типу згорткової нейронної мережі.

Під час експерименту використовувалася плата Arduino Nano 33 BLE Sense. Модель нейронної мережі була побудована та навчалась на програмній платформі Edge Impulse. Для проведення експерименту було створено три групи даних з назвами «hello», «невідомо», «шум». За результатами експериментальних досліджень можна зробити висновок, що обчислювальних ресурсів 32-х бітних мікроконтролерів цілком достатньо для розпізнавання голосових команд з можливістю попередньої цифрової обробки звукового сигналу, зокрема, використання мел-частотних кепстральних коефіцієнтів. Вибір кількості коефіцієнтів не впливає значним чином на об’єм задіяної FLASH та RAM пам’яті мікроконтролера nRF52840.

Використання для розпізнавання зразка голосу одновимірної згорткової нейромережі у проведеному експерименті забезпечує економію приблизно 5% пам’яті. Якість розпізнавання ключового слова при кількості MFC коефіцієнтів 12 складає приблизно 0,7. Для 17-ти MFC коефіцієнтів якість розпізнавання становить вже 0,97. Таким чином, 1D згорткової нейромережі мають певні переваги у мікроконтролерних застосунках для обробки та розпізнавання голосу.

ЛІТЕРАТУРА

[1] S. Misra, T. Das, P. Saha, U. Baruah and R. H. Laskar, "Comparison of MFCC and LPCC for a fixed phrase speaker verification system, time complexity and failure analysis," 2015 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2015], 2015, pp. 1-4, doi: 10.1109/ICCPCT.2015.7159307.

[2] Zheng, F., Zhang, G. & Song, Z. “Comparison of different implementations of MFCC”, J. Computer Science & Technology 16, 2001, pp.582–589, doi: 10.1007/BF02943243.

[3] Md Sahidullah, G. Saha, "Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition," Speech Communication, Volume 54, Issue 4, May 2012, pp. 543-565, doi: 10.1016/j.specom.2011.11.004.

[4] O. Cheng, W. Abdulla and Z. Salcic, "Hardware–Software Codesign of Automatic Speech Recognition System for Embedded Real-Time Applications," in IEEE Transactions on Industrial Electronics, vol. 58, no. 3, pp. 850-859, March 2011, doi: 10.1109/TIE.2009.2022520.

[5] F. Barkani, H. Satori, M. Hamidi, O. Zealouk and N. Laaidi, "Amazigh Speech Recognition Embedded System," 2020 1st International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET), 2020, pp. 1-5, doi: 10.1109/IRASET48871.2020.9092014.

[6] A. G. Howard M. Zhu B. Chen D. Kalenichenko W. Wang T. Weyand et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications" arXiv preprint arXiv 17 Apr 2017, pp. 1-9, doi: 10.48550/arXiv.1704.04861

[7] D. Sinha and M. El-Sharkawy, "Ultra-thin MobileNet," 2020 10th Annual Computing and Communication Workshop and Conference (CCWC), 2020, pp. 0234-0240, doi: 10.1109/CCWC47524.2020.9031228.

[8] Y. -C. Ling, H. -H. Chin, H. -I. Wu and R. -S. Tsay, "Designing A Compact Convolutional Neural Network Processor on Embedded FPGAs," 2020 IEEE Global Conference on Artificial Intelligence and Internet of Things (GCAIoT), 2020, pp. 1-7, doi: 10.1109/GCAIoT51063.2020.9345903.

[9] S. M. A. H. Jafri, A. Hemani and L. Intesa, "SPEED: Open-Source Framework to Accelerate Speech Recognition on Embedded GPUs," 2017 Euromicro Conference on Digital System Design (DSD), 2017, pp. 94-101, doi: 10.1109/DSD.2017.89.

[10] F. Raffaeli and S. Awad, "Portable low-cost platform for embedded speech analysis and synthesis," 2016 12th International Computer Engineering Conference (ICENCO), 2016, pp. 117-122, doi: 10.1109/ICENCO.2016.7856455.

[11] A. P. Pant, K. -R. Wu and Y. -C. Tseng, "Speak to Action: Offline and Hybrid Language Recognition on Embedded Board for Smart Control System," 2020 International Computer Symposium (ICS), 2020, pp. 85-90, doi: 10.1109/ICS51289.2020.00026.

[12] F. Sutton, R. Da Forno, R. Lim, M. Zimmerling and L. Thiele, "Demonstration abstract: Automatic speech recognition for resource-constrained embedded systems," IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks, 2014, pp. 323-324, doi: 10.1109/IPSN.2014.6846784.

[13] I. Kramberger, M. Grasic and T. Rotovnik, "Door phone embedded system for voice based user identification and verification platform," in IEEE Transactions on Consumer Electronics, vol. 57, no. 3, pp. 1212-1217, August 2011, doi: 10.1109/TCE.2011.6018876.

[14] Q. Qu and L. Li, "Realization of embedded speech recognition module based on STM32," 2011 11th International Symposium on Communications & Information Technologies (ISCIT), 2011, pp. 73-77, doi: 10.1109/ISCIT.2011.6092186.

[15] “TensorFlow”, TensorFlow.org сайт: https://www.tensorflow.org/ (дата звернення 05.09.2022)

[16] "Keras: The Python deep learning API", Keras: the Python deep learning API. сайт: https://keras.io/ (дата звернення 04.09. 2022).

[17] C. M. J. Galangque and S. A. Guirnaldo, "Speech Recognition Engine using ConvNet for the development of a Voice Command Controller for Fixed Wing Unmanned Aerial Vehicle (UAV)," 2019 12th International Conference on Information & Communication Technology and System (ICTS), 2019, pp. 93-97, doi: 10.1109/ICTS.2019.8850961.

[18] J. Dudak, M. Kebisek, G. Gaspar and P. Fabo, "Implementation of machine learning algorithm in embedded devices," 2020 19th International Conference on Mechatronics - Mechatronika (ME), 2020, pp. 1-6, doi: 10.1109/ME49197.2020.9286705.

[19] O'Brien S. Voice Recognition | RingCentral UK Blog. *RingCentral UK Blog*. URL: https://www.ringcentral.com/gb/en/blog/definitions/voice-recognition/ (дата звернення: 01.11.2022).

[20] VOICE RECOGNITION SYSTEM using microcontroller. *One moment,https://microcontrollerslab.com/voice-recognition-system-using-microcontroller/ please...* URL: https://microcontrollerslab.com/voice-recognition-system-using-microcontroller/ (дата звернення: 01.11.2022).

[21]Goel, Vaibhava; Byrne, William J. (2000). "Minimum Bayes-risk automatic speech recognition". Computer Speech & Language. **14** (2): 115–135. doi:10.1006/csla.2000.0138

[22] Mohri, M. (2002). "Edit-Distance of Weighted Automata: General Definitions and Algorithms" (PDF). International Journal of Foundations of Computer Science. **14** (6): 957–982. doi:10.1142/S0129054103002114. Archived (PDF) from the original on 18 March 2012. Retrieved 28 March 2011.

[23] Waibel, A.; Hanazawa, T.; Hinton, G.; Shikano, K.; Lang, K. J. (1989). "Phoneme recognition using time-delay neural networks". IEEE Transactions on Acoustics, Speech, and Signal Processing. **37** (3): 328–339. doi:10.1109/29.21701. hdl:10338.dmlcz/135496.

[24] Bird, Jordan J.; Wanner, Elizabeth; Ekárt, Anikó; Faria, Diego R. (2020). "Optimisation of phonetic aware speech recognition through multi-objective evolutionary algorithms" (PDF). Expert Systems with Applications. Elsevier BV. **153**: 113402. doi:10.1016/j.eswa.2020.113402. ISSN 0957-4174. S2CID 216472225.

[25] Wu, J.; Chan, C. (1993). "Isolated Word Recognition by Neural Network Models with Cross-Correlation Coefficients for Speech Dynamics". IEEE Transactions on Pattern Analysis and Machine Intelligence. **15** (11): 1174–1185. doi:10.1109/34.244678.

[26] S. A. Zahorian, A. M. Zimmer, and F. Meng, (2002) "Vowel Classification for Computer based Visual Feedback for Speech Training for the Hearing Impaired," in ICSLP 2002

[27] Hu, Hongbing; Zahorian, Stephen A. (2010). "Dimensionality Reduction Methods for HMM Phonetic Recognition" (PDF). ICASSP 2010. Archived (PDF) from the original on 6 July 2012.

[28]Fernandez, Santiago; Graves, Alex; Schmidhuber, Jürgen (2007). "Sequence labelling in structured domains with hierarchical recurrent neural networks" (PDF). Proceedings of IJCAI. Archived (PDF) from the original on 15 August 2017.

[29] Graves, Alex; Mohamed, Abdel-rahman; Hinton, Geoffrey (2013). "Speech recognition with deep recurrent neural networks". arXiv:1303.5778 [cs.NE]. ICASSP 2013.

[30] Waibel, Alex (1989). "Modular Construction of Time-Delay Neural Networks for Speech Recognition" (PDF). Neural Computation. **1** (1): 39–46. doi:10.1162/neco.1989.1.1.39. S2CID 236321.

[31] Maas, Andrew L.; Le, Quoc V.; O'Neil, Tyler M.; Vinyals, Oriol; Nguyen, Patrick; Ng, Andrew Y. (2012). "Recurrent Neural Networks for Noise Reduction in Robust ASR". Proceedings of Interspeech 2012.

[32] Deng, Li; Yu, Dong (2014). "Deep Learning: Methods and Applications" (PDF). Foundations and Trends in Signal Processing. **7** (3–4): 197–387. CiteSeerX 10.1.1.691.3679. doi:10.1561/2000000039.

[33] Yu, D.; Deng, L.; Dahl, G. (2010). "Roles of Pre-Training and Fine-Tuning in Context-Dependent DBN-HMMs for Real-World Speech Recognition" (PDF). NIPS Workshop on Deep Learning and Unsupervised Feature Learning.

[34] Dahl, George E.; Yu, Dong; Deng, Li; Acero, Alex (2012). "Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition". IEEE Transactions on Audio, Speech, and Language Processing. **20** (1): 30–42. doi:10.1109/TASL.2011.2134090. S2CID 14862572.

[35] Deng L., Li, J., Huang, J., Yao, K., Yu, D., Seide, F. et al. Recent Advances in Deep Learning for Speech Research at Microsoft. ICASSP, 2013.

[36] Yu, D.; Deng, L. (2014). "Automatic Speech Recognition: A Deep Learning Approach (Publisher: Springer)".

[37] Tüske, Zoltán; Golik, Pavel; Schlüter, Ralf; Ney, Hermann (2014). "Acoustic Modeling with Deep Neural Networks Using Raw Time Signal for LVCSR" (PDF). Interspeech 2014.

[38] Jurafsky, Daniel (2016). Speech and Language Processing.

[39] Graves, Alex (2014). "Towards End-to-End Speech Recognition with Recurrent Neural Networks" (PDF). ICML.

[40] Amodei, Dario (2016). "Deep Speech 2: End-to-End Speech Recognition in English and Mandarin". arXiv:1512.02595 [cs.CL].

[41]"LipNet: How easy do you think lipreading is?". YouTube.

[42] Assael, Yannis; Shillingford, Brendan; Whiteson, Shimon; de Freitas, Nando (5 November 2016). "LipNet: End-to-End Sentence-level Lipreading".

[43] Shillingford, Brendan; Assael, Yannis; Hoffman, Matthew W.; Paine, Thomas; Hughes, Cían; Prabhu, Utsav; Liao, Hank; Sak, Hasim; Rao, Kanishka (13 July 2018). "Large-Scale Visual Speech Recognition".

[44] Chan, William; Jaitly, Navdeep; Le, Quoc; Vinyals, Oriol (2016). "Listen, Attend and Spell: A Neural Network for Large Vocabulary Conversational Speech Recognition"

[45] Bahdanau, Dzmitry (2016). "End-to-End Attention-based Large Vocabulary Speech Recognition".

[46] Chorowski, Jan; Jaitly, Navdeep (8 December 2016). "Towards better decoding and language model integration in sequence to sequence models".

[47] Chan, William; Zhang, Yu; Le, Quoc; Jaitly, Navdeep (10 October 2016). "Latent Sequence Decompositions".

[48] Chung, Joon Son; Senior, Andrew; Vinyals, Oriol; Zisserman, Andrew (16 November 2016). "Lip Reading Sentences in the Wild". 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3444–3453.

[49] Englund, Christine (2004). Speech recognition in the JAS 39 Gripen aircraft: Adaptation to speech at different G-loads (PDF) (Masters thesis). Stockholm Royal Institute of Technology.

[50] "The Cockpit". Eurofighter Typhoon.

[51] "Eurofighter Typhoon – The world's most advanced fighter aircraft". www.eurofighter.com.

[52] Schutte, John (15 October 2007). "Researchers fine-tune F-35 pilot-aircraft speech system". United States Air Force.

[53] Cerf, Vinton; Wrubel, Rob; Sherwood, Susan. "Can speech-recognition software break down educational language barriers?". Curiosity.com. Discovery Communications.

[54] "Speech Recognition for Learning". National Center for Technology Innovation. 2010.

[55] Follensbee, Bob; McCloskey-Dale, Susan (2000). "Speech recognition in schools: An update from the field". Technology And Persons With Disabilities Conference 2000.

[56] "Overcoming Communication Barriers in the Classroom". MassMATCH. 18 March 2010.

[57] Garrett, Jennifer Tumlin; et al. (2011). "Using Speech Recognition Software to Increase Writing Fluency for Individuals with Physical Disabilities". Journal of Special Education Technology. **26** (1): 25–41. doi:10.1177/016264341102600104. S2CID 142730664.

[58] Tang, K. W.; Kamoua, Ridha; Sutan, Victor (2004). "Speech Recognition Technology for Disabilities Education". Journal of Educational Technology

Systems. **33** (2): 173–84. CiteSeerX 10.1.1.631.3736. doi:10.2190/K6K8-78K2-59Y7-R9R2. S2CID 143159997.

[59] Ciaramella, Alberto. "A prototype performance evaluation report." Sundial workpackage 8000 (1993)

[60] National Institute of Standards and Technology. "The History of Automatic Speech Recognition Evaluation at NIST at the Wayback Machine".

[61] O'Brien S. Voice Recognition | RingCentral UK Blog. *RingCentral UK Blog*. URL: https://www.ringcentral.com/gb/en/blog/definitions/voice-recognition/ (дата звернення: 02.11.2022).

[62] X. Huang, A. Acero, H.-W. Hon, R. Reddy, "Spoken Language Processing - A Guide to Theory, Algorithm, and System Development", Prentice Hall, 2001, 965pp.

[63] Hui J. Speech Recognition–Feature Extraction MFCC & PLP. *Medium*. URL: https://jonathan-hui.medium.com/speech-recognition-feature-extraction-mfcc-plp-5455f5a69dd9 (дата звернення: 01.11.2022).

[64] Gu, J., et al., "Recent advances in convolutional neural networks", Pattern Recognition, 2018, 77: pp. 354-377, doi: 10.48550/arXiv.1512.07108

[65] S. Kiranyaz, T. Ince, O. Abdeljaber, O. Avci and M. Gabbouj, "1-D Convolutional Neural Networks for Signal Processing Applications," ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp. 8360-8364, doi: 10.1109/ICASSP.2019.8682194

[66] A. Shenfield, M. Howarth. "A Novel Deep Learning Model for the Detection and Identification of Rolling Element-Bearing Faults" Sensors 2020, 20, 5112. doi: 10.3390/s20185112

[67] Edge impulse, edgeimpulse.com, сайт: https://www.edgeimpulse.com/ (дата звернення 05.09. 2022).

[68] Arduino Nano 33 BLE, store.arduino.cc, сайт: https://store.arduino.cc/products/arduino-nano-33-ble (дата звернення 05.09. 2022).

[69] Kubanek M., Bobulski J., Kulawik, J. (2019) A method of speech coding for speech recognitionusing a convolutional neural network. Symmetry, 11(9), 1185.

[70] Nwankpa C., Ijomah W., Gachagan, A., Marshall, S. (2018) Activation functions: Comparisonof trends in practice and research for deep learning. arXiv preprint arXiv:1811.03378.

[71] Devices - Edge Impulse Documentation. *Getting Started - Edge Impulse Documentation*. URL: https://docs.edgeimpulse.com/docs/edge-impulse-studio/devices (дата звернення: 10.11.2022).

[72] Data acquisition - Edge Impulse Documentation. *Getting Started - Edge Impulse Documentation*. URL: https://docs.edgeimpulse.com/docs/edge-impulse-studio/data-acquisition (дата звернення: 10.11.2022).

[73] Impulse design - edge impulse documentation. *Getting Started - Edge Impulse Documentation*. URL: https://docs.edgeimpulse.com/docs/edge-impulse-studio/impulse-design (дата звернення: 11.11.2022).

[74] Audio MFCC - Edge Impulse Documentation. *Getting Started - Edge Impulse Documentation*. URL: https://docs.edgeimpulse.com/docs/edge-impulse-studio/processing-blocks/audio-mfcc (дата звернення: 11.11.2022).

[75] Classification (Keras) - Edge Impulse Documentation. *Getting Started - Edge Impulse Documentation*. URL: https://docs.edgeimpulse.com/docs/edge-impulse-studio/learning-blocks/classification (дата звернення: 12.11.2022).

[76] Deployment - Edge Impulse Documentation. *Getting Started - Edge Impulse Documentation*. URL: https://docs.edgeimpulse.com/docs/edge-impulse-studio/deployment (дата звернення: 13.11.2022).

[77] Arduino Nano 33 BLE Sense Pinout, Introduction & Specifications. *eTechnophiles*. URL: https://www.etechnophiles.com/arduino-nano-33-ble-sense-pinout-introduction-specifications/ (дата звернення: 14.11.2022).

[78] Advanced multi-protocol System-on-Chip nRF52840 Retrieved from https://infocenter.nordicsemi.com/index.jsp?topic=%2Fps_nrf52840%2Fkeyfeatures_html5.html

[79] MEMS audio sensor omnidirectional digital microphone Datasheet MP34DT05-A Retrieved from https://www.st.com/resource/en/datasheet/mp34dt05-a.pdf

[80] Seeed_Arduino_Sketchbook/WioTerminal_EI_Microphone_Inference.ino at master · Seeed-Studio/Seeed_Arduino_Sketchbook. *GitHub*. URL: https://github.com/Seeed-Studio/Seeed_Arduino_Sketchbook/blob/master/examples/WioTerminal_TinyML_2_Audio_Scene_Recognition/WioTerminal_EI_Microphone_Inference/WioTerminal_EI_Microphone_Inference.ino (дата звернення: 14.11.2022).

Додаток 1. Приклад коду для навченої моделі Edge Impulse для розпізнавання ключового слова [80].

```
/* Includes ---------------------------------------------------------------
-- */

#include <PDM.h>

#include <anna.ginger-project-1_inferencing.h>


/** Audio buffers, pointers and selectors */

typedef struct {

    int16_t *buffer;

    uint8_t buf_ready;

    uint32_t buf_count;

    uint32_t n_samples;

} inference_t;


static inference_t inference;

static signed short sampleBuffer[2048];

static bool debug_nn = false; // Set this to true to see e.g. features
generated from the raw signal


/**

 * @brief      Arduino setup function

 */

void setup()

{

    // initialize digital pin LED_BUILTIN as an output.

    pinMode(LED_BUILTIN, OUTPUT);


    // put your setup code here, to run once:

    Serial.begin(115200);


    Serial.println("Edge Impulse Inferencing Demo");
```

```
    // summary of inferencing settings (from model_metadata.h)
    ei_printf("Inferencing settings:\n");
    ei_printf("\tInterval: %.2f ms.\n", (float)EI_CLASSIFIER_INTERVAL_MS);
    ei_printf("\tFrame size: %d\n", EI_CLASSIFIER_DSP_INPUT_FRAME_SIZE);
    ei_printf("\tSample length: %d ms.\n", EI_CLASSIFIER_RAW_SAMPLE_COUNT
/ 16);
    ei_printf("\tNo. of classes: %d\n",
sizeof(ei_classifier_inferencing_categories) /
sizeof(ei_classifier_inferencing_categories[0]));


    if (microphone_inference_start(EI_CLASSIFIER_RAW_SAMPLE_COUNT) ==
false) {
        ei_printf("ERR: Failed to setup audio sampling\r\n");
        return;
    }
}


/**
 * @brief      Arduino main function. Runs the inferencing loop.
 */
void loop()
{
    while(!Serial.available()){}
    String str = Serial.readString();
    ei_printf("Starting inferencing in 2 seconds...\n");


    delay(1000);


    ei_printf("Recording...\n");
    digitalWrite(LED_BUILTIN, HIGH);    // turn the LED on (HIGH is the
voltage level)


    bool m = microphone_inference_record();
    if (!m) {
        ei_printf("ERR: Failed to record audio...\n");
        return;
```

```
    }

    digitalWrite(LED_BUILTIN, LOW);     // turn the LED off by making the
voltage LOW
    ei_printf("Recording done\n");

    /*
    for(int i = 0; i < inference.n_samples; i++)
        Serial.println(inference.buffer[i]);

    Serial.println();
    */


    signal_t signal;
    signal.total_length = EI_CLASSIFIER_RAW_SAMPLE_COUNT;
    signal.get_data = &microphone_audio_signal_get_data;
    ei_impulse_result_t result = { 0 };

    EI_IMPULSE_ERROR r = run_classifier(&signal, &result, debug_nn);
    if (r != EI_IMPULSE_OK) {
        ei_printf("ERR: Failed to run classifier (%d)\n", r);
        return;
    }

    // print the predictions
    ei_printf("Predictions ");
    ei_printf("(DSP: %d ms., Classification: %d ms., Anomaly: %d ms.)",
        result.timing.dsp,                    result.timing.classification,
result.timing.anomaly);
    ei_printf(": \n");
    for (size_t ix = 0; ix < EI_CLASSIFIER_LABEL_COUNT; ix++) {
        ei_printf("        %s:  %.5f\n",  result.classification[ix].label,
result.classification[ix].value);
    }
#if EI_CLASSIFIER_HAS_ANOMALY == 1
```

```
    ei_printf("    anomaly score: %.3f\n", result.anomaly);
#endif
}

/**
 * @brief      PDM buffer full callback
 *             Get data and call audio thread callback
 */
static void pdm_data_ready_inference_callback(void)
{
    int bytesAvailable = PDM.available();

    // read into the sample buffer
    int bytesRead = PDM.read((char *)&sampleBuffer[0], bytesAvailable);

    if (inference.buf_ready == 0) {
        for(int i = 0; i < bytesRead>>1; i++) {
            inference.buffer[inference.buf_count++] = sampleBuffer[i];

            if(inference.buf_count >= inference.n_samples) {
                inference.buf_count = 0;
                inference.buf_ready = 1;
                break;
            }
        }
    }
}

/**
 * @brief      Init inferencing struct and setup/start PDM
 *
 * @param[in]  n_samples  The n samples
 *
 * @return     { description_of_the_return_value }
 */
```

```
static bool microphone_inference_start(uint32_t n_samples)
{
    inference.buffer = (int16_t *)malloc(n_samples * sizeof(int16_t));

    if(inference.buffer == NULL) {
        return false;
    }

    inference.buf_count  = 0;
    inference.n_samples  = n_samples;
    inference.buf_ready  = 0;

    // configure the data receive callback
    PDM.onReceive(&pdm_data_ready_inference_callback);

    PDM.setBufferSize(4096);

    // initialize PDM with:
    // - one channel (mono mode)
    // - a 16 kHz sample rate
    if (!PDM.begin(1, EI_CLASSIFIER_FREQUENCY)) {
        ei_printf("Failed to start PDM!");
        microphone_inference_end();

        return false;
    }

    // set the gain, defaults to 20
    PDM.setGain(127);

    return true;
}

/**
 * @brief      Wait on new data
```

```
 *
 * @return     True when finished
 */
static bool microphone_inference_record(void)
{
    inference.buf_ready = 0;
    inference.buf_count = 0;

    while(inference.buf_ready == 0) {
        delay(10);
    }

    return true;
}

/**
 * Get raw audio signal data
 */
static int microphone_audio_signal_get_data(size_t offset, size_t length,
float *out_ptr)
{
    numpy::int16_to_float(&inference.buffer[offset], out_ptr, length);

    return 0;
}

/**
 * @brief      Stop PDM and release buffers
 */
static void microphone_inference_end(void)
{
    PDM.end();
    free(inference.buffer);
}
```

```
#if    !defined(EI_CLASSIFIER_SENSOR)    ||    EI_CLASSIFIER_SENSOR    !=
EI_CLASSIFIER_SENSOR_MICROPHONE

#error "Invalid model for current sensor."

#endif
```