Міністерство освіти і науки України
Департамент науки і освіти Харківської облдержадміністрації
Харківське територіальне відділення МАН України

Відділення: комп'ютерних наук
Секція: інформаційні системи, бази даних та системи штучного інтелекту

РОЗРОБКА СИСТЕМИ КЛАСИФІКАЦІЇ МУЗИЧНИХ КОМПОЗИЦІЙ ЗА ЖАНРАМИ

Роботу виконав:
Харченко Федір Олександрович,
учень 9 класу Харківського
Навчально-виховного комплексу
№45 «Академічна гімназія»
Харківської міської ради
Харківської області

Науковий керівник:
Руккас Кирило Маркович,
професор кафедри теоретичної та
прикладної інформатики
механіко-математичного
факультету Харківського
національного університету
ім. В.Н. Каразіна, доктор
технічних наук, доцент

Харків – 2021

Жанрова класифікація музичних композицій за допомогою машинного навчання;

Автор роботи: Харченко Федір Олександрович;
Харківський навчально-виховний комплекс №45 «Академічна гімназія» Харківської міської ради Харківської області; 9 клас; м. Харків;
Науковий керівник: Руккас Кирило Маркович, доцент Харківського національного університету імені В.Н. Каразіна, кандидат фізико-математичних наук.

Жанри можна визначити як мітки, створені людьми для ідентифікації або характеристики стилю музики. Це дослідження представляє підхід машинного навчання до проблеми автоматичної класифікації музики за жанром з використанням перетворення звукового сигналу. Система розроблена з використанням глибокої нейронної мережі для розпізнавання жанрів. Для представлення музики використовуються характеристики мел-частотних кепстральних коефіцієнтів. Система оцінюється за допомогою наборів даних MIR.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

MFCC — мел-частотні кепстральні коефіцієнти.

ВСТУП

В Інтернеті багато музики, пісень, та відео. Їх кількість продовжує рости. Сьогодні багато музичних файлів залежать від автоматизованих алгоритмів аналізу та індексації музикального контенту. Звідси автоматичне видобування музичної інформації становиться ще важливим, як спосіб організувати та структурувати музичні файли. Цією справою займається MIR(Music Information Retrieval) - міждисциплінарна наука отримання інформації з музики. Це невелика, але зростаюча галузь досліджень із багатьма реальними додатками. Ті, хто бере участь у MIR, можуть мати досвід музикознавства , психоакустики , психології , академічного вивчення музики, обробки сигналів , інформатики , машинного навчання , оптичного розпізнавання музики, обчислювального інтелекту або якоїсь їх комбінації.

Моє дослідження будується на жанрової колекції GTZAN. Набір даних складається з 1000 звукових доріжок по 30 секунд. Він містить 10 жанрів, кожен представлений 100 треками.

В сучасний час на увазі постійно розвиваючихся веб-технологій і закономірно зростаючої чисельністю різноманітних сервісів, що надають користувачам деяку інформацію, росте конкуренція в різних сферах. Зокрема, з’являється все більше музичних сервісів. Для того, щоб бути успішними на ринку, подібні засоби повинні надавати можливість не тільки прослуховувати музичні композиції, а й забезпечити додаткові функції. Наприклад, корисною є можливість аналізу жанрових уподобань користувача по прослуханим композиціям для подальшого складання списку рекомендацій.

Класифікація музичних жанрів іноді вважається суб’єктивною справою. Жанрові теги музичних треків часто відзначаються виконавцем або користувачами.

Нейронні мережі досягли глибокого успіху в області розпізнавання образів. Повторно тренуя мережу, її можна навчити розрізняти критерії, що викори-

стовуються для класифікації, і вона може зробити це узагальнено, дозволяючи успішну класифікацію нових даних, які не використовувались під час навчання. З вибухом цифрової музики в останні роки завдяки Інтернету, застосування технології розпізнавання образів до цифрового аудіо стає все цікавішим. При цьому підході особливий інтерес представляє якість підготовки вхідних даних. Прямий аналіз звукових сигналів у часовій області може потенційно зайняти багато часу в залежності від тривалості та якості запису, і він сам по собі не є найбільш ефективним методом. Складання спектрограм і їх подальший аналіз буде швидше, але все ж не цілком ефективним.

На даний момент одним з найбільш раціональних уявлень записи для подальшого аналізу є метод крейда-частотних кепстральних коефіцієнтів, який широко застосовується для складання характеристик мовних сигналів. Цей метод отримав широке поширення в сфері завдань розпізнавання мови. Дана метрика більше наближена до того, як людина оцінює музичну композицію на слух, так як ми сприймаємо висоту звуку в певний момент часу, а не дані про частоти, які є основою спектрограм.

Метою дослідження є класифікування музичних композицій на жанри за допомогою нейронної мережі.

План для досягнення цієї мети:

1. Здобути набір даних, який представляє набір аудіофайлов, та показує приналежність кожної композиції к її жанру.

2. Знайти та виконати метод перетворення даних, за допомогою якого ми могли характеризувати кожен аудіофайл для машинного навчання.

3. Створення та тренування нейромережі на тренувальних даних.

4. Експерименти, порівняння та покращення результатів.

РОЗДІЛ 1.
НАБІР ДАНИХ ТА ВИТЯГ ХАРАКТЕРИСТИК

1.1. Аналіз даних

Звук — коливальний рух частинок середовища, що поширюється у вигляді хвиль у газі, рідині чи твердому тілі [1]. Для зручності ми будемо розглядати лише звук в повітрі. Звук утворюється вібруючими предметами, змушуючи молекули середовища коливатися і врізатися одна в одну. Потрапляючи один на одного, молекули змінюють стан тиску повітря в регіоні, де вони поширюються. Таким чином вони створюють в цьому процесі хвилю. Іншими словами можна сприймати звук як хвиля, яка передає деяку енергію від однієї точки до іншої за допомогою повітря, яка сприймаються сенсорною системою тварин і людини.

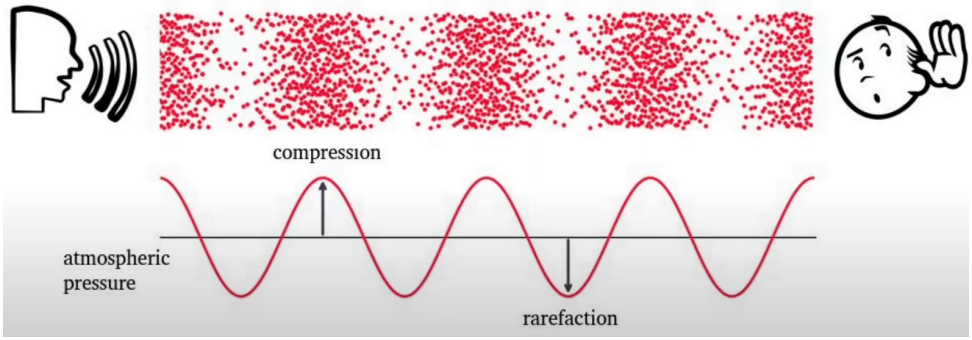

Рис. 1. Звукова хвиля

Розглянемо малюнок 1. Коли молекули врізаються, тим самим вони створюють високий тиск, потім вони віддаляються одна від одної і створюють розрідження. Ми можемо уявити це використовуючи графік тиску.

Розглянемо простий звуковий сигнал. Це синусоїда, яка виражається такою формулою:

$$y_t = A \cdot \sin \cdot (2\pi f t + \varphi)$$

Де A — амплітуда, f — частота, t — час, а φ — фаза.

Такий звук називається аналоговим. Аналоговий сигнал - сигнал даних, у

якого кожен з представлених параметрів описується функцією часу і безперервним безліччю можливих значень.

Частота — це величина, зворотня періоду, яка показує кількість коливань за секунду. Чим більше частота, тим вище звук. Амплітуда показує наскільки високо низько змінюється тиск повітря. Чим більше амплітуда, тим гучніше звук. Останній параметр — це фаза, цей параметр дозволяє нам робити зміщення хвилі праворуч або ліворуч, фаза в основному повідомляє нам, в якому положенні знаходиться хвиля в нульовий момент часу.

Відсутність чітко відмітних один від одного рівнів сигналу призводить до неможливості застосувати для його опису поняття інформації в тому вигляді, як вона розуміється в цифрових технологіях. Для цього ми маємо аналогово-цифровий процес перетворення. Виконуємо два етапи: дискретизація та квантування. Ми відбираємо сигнал через певні інтервали часу, а потім квантуємо амплітуду і представляємо з обмеженою кількістю бітів. Квантування - розбиття діапазону відлікових значень сигналу на кінцеве число рівнів і округлення цих значень до одного з двох найближчих до них рівнів. Чим більше ми маємо бітів для зберігання амплітуди (бітрейт), тим більше буде якість звука. Також на якість впливає кількість зразків на одну секунду - частота дискретизації.

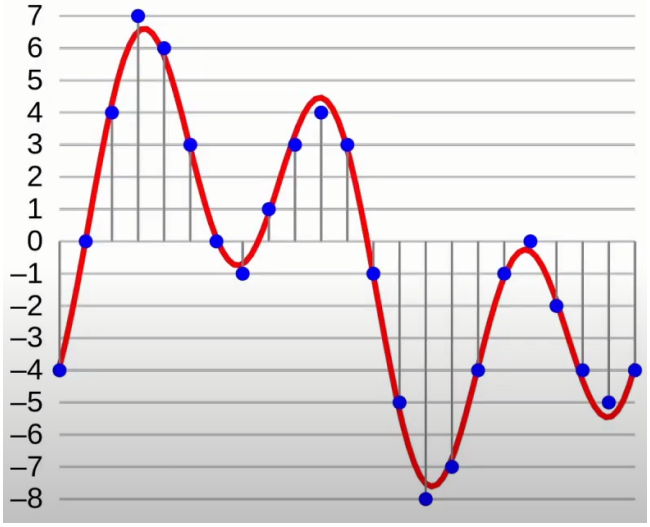

Рис. 2. Аналогово-цифровий процес перетворення

Наш набір даних складається з 1000 звукових доріжок по 30 секунд. Він

містить 10 жанрів, кожен представлений 100 треками. Усі доріжки складаються з 16-бітових аудіофайлів, частота вибірки 22050 Гц у форматі .wav. Формат має передбачає наявність у файлі двох блоків. Перший блок - це заголовок з інформацією про аудіопотоки: бітрейте, частоті, кількості каналів, довжині файлу і т.д. Другий блок складається з «сирих» даних - того самого цифрового сигналу, набору значень амплітуд.

1.2. Перетворення даних, як вхідні дані до нейромережі

Розглянемо цифровий сигнал, який ми отримали. Він буде містити в собі дуже багато різних звуків, які періодично змінюють один одного або грають одночасно. За амплітудою, яка вже в нас є ми тільки можемо сказати коли музика грає тихіше, коли голосніше.

1.2.1 Перетворення Фур'є

Один із способів добуття інформації - перетворення Фур'є. Перетворення Фур'є - операція, що описує коефіцієнти амплітуди на коливання з різними частотами.

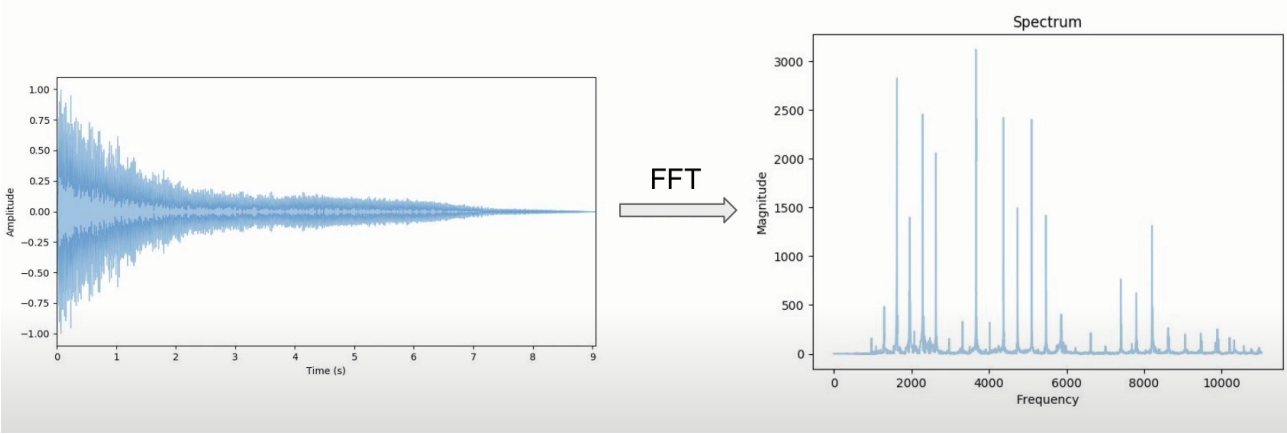

Рис. 3. Перетворення Фур'є на прикладі звуку фортепіано

Спочатку ми маємо амплітуду, як функцію від часу, а отримуємо, амплітуду, як функцію від частоти (подібно до того, як музичний акорд може бути

виражений у вигляді суми музичних звуків, які його складають). Але є один недолік- ми втрачаємо інформацію про час.

1.2.2 Короткочасне перетворення Фур'є

Короткочасне перетворення Фур'є обчислює декілька перетворень Фур'є з різними інтервалами, і при цьому зберігає інформацію про час та еволюцію звуку. розіб'ємо наші дані по невеликих тимчасових проміжків — фреймам. Причому фрейми повинні йти не строго один за одним, а "внахлест". Тобто кінець одного фрейма повинен перетинатися з початком іншого. Фрейми є більш придатною одиницею аналізу даних, ніж конкретні значення сигналу, так як аналізувати хвилі набагато зручніше на деякому проміжку, ніж в конкретних точках. Розташування ж фреймів внахлест дозволяє згладити результати аналізу фреймів[2]. Різні інтервали, на яких ми виконуємо перетворення Фур'є, задаються розміром кадру.

На виході ми отримуємо спектрограму яка дає інформацію про значимість звуку відносно часу та частоти. Безпосередньо порівнювати звукові сигнали в тимчасовій області - довго і не дуже ефективно. Спектрограми - вже швидше, але не набагато ефективніше, найбільш раціональне уявлення — мел-частотні кепстральні коефіцієнти.

1.2.3 Мел-частотні кепстральні коефіцієнти, mel-frequency cepstral coefficients (MFCC)

Мел — одиниця висоти звуку, заснована на сприйнятті цього звуку нашими органами слуху. Застосовується головним чином в музичній акустиці. Назва походить від слова «мелодія». Висота звуку, яка сприймається людським слухом, не зовсім лінійно залежить від його частоти.

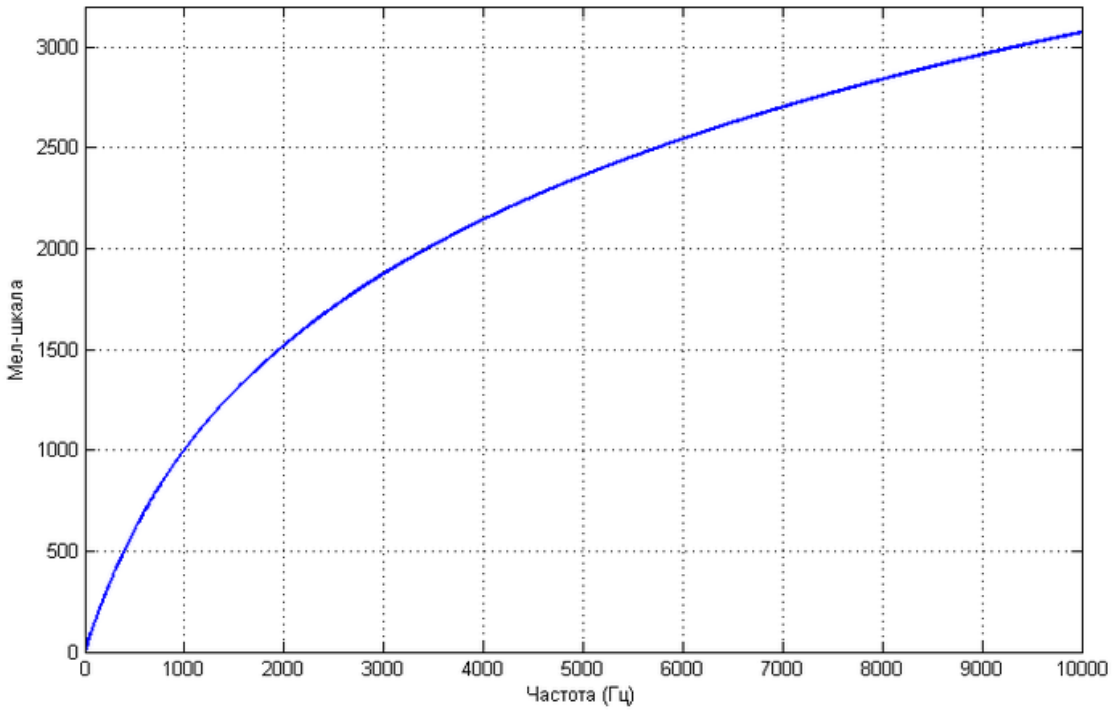

Рис. 4. Графік залежності мел від частоти

Ця залежність описується формулою:

$$m = 1125 \cdot ln(1 + \frac{f}{700})$$

Для зворотного перетворення:

$$f = 700 \cdot (exp \cdot (\frac{m}{1125}) - 1)$$

Метод полягає в тому, що насамперед нам потрібен спектр одного фрейму вихідного сигналу, який ми отримуємо за допомогою перетворення Фур'є. Далі ми використовуємо вікна, рівномірно розташовані на мел осі. Кожне вікно характеризує свій коефіцієнт, тому в залежності скільки в нас буде коефіцієнтів, стільки в нас буде вікон. Якщо перевести цей графік у частотну шкалу, можна побачити що вікна збираються в області низьких частот. Потім перемножуємо вектори спектра сигнала та віконної функції, результат зводимо в квадрат і логарифмуємо, таким чином ми знаємо значення всіх коефіцієнтів одного фрейму.

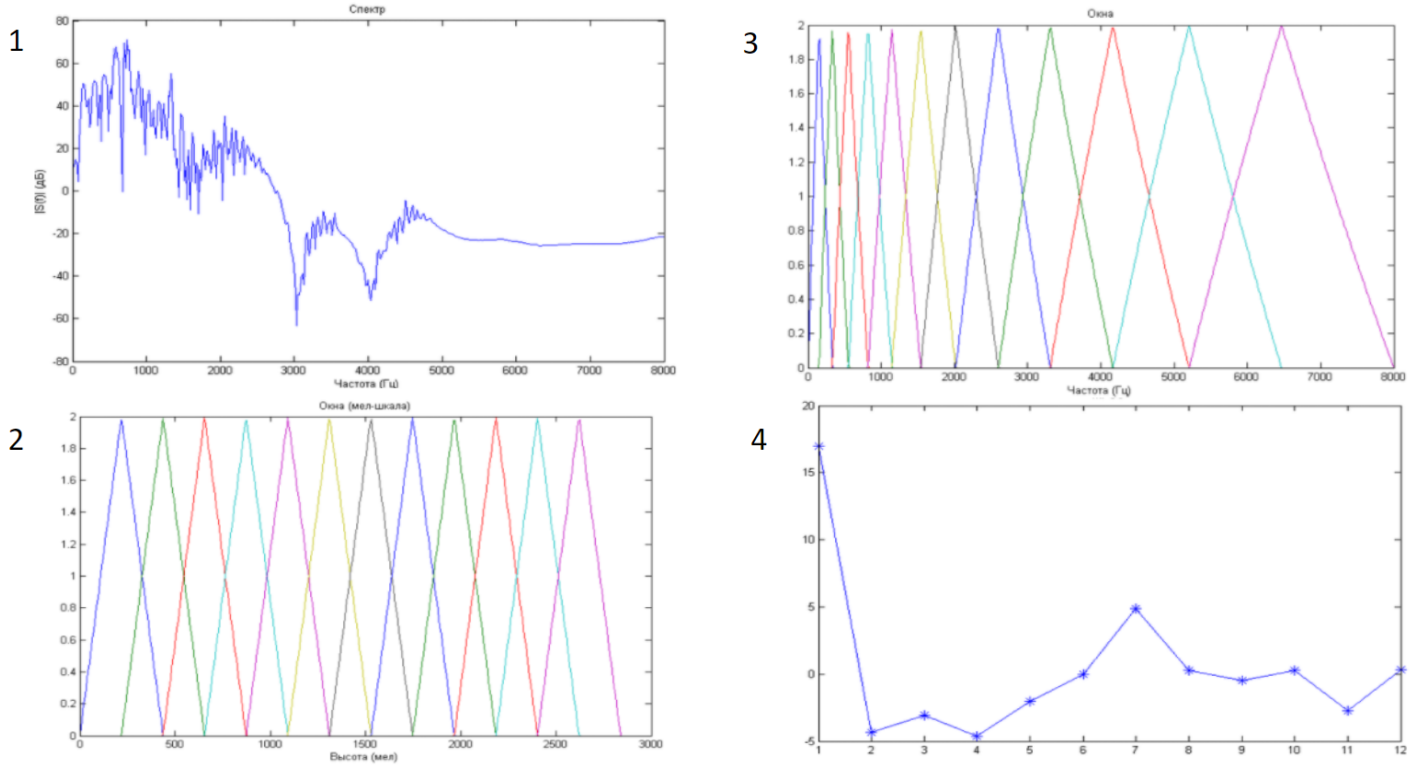

Рис. 5. Виконання методу на прикладі 12 коефіцієнтів

Тепер повторюємо ці операції на кожному фреймі, у результаті ми отримуємо мел-спектрограму, невеликий двовимірний набір значень, який при розпізнаванні успішно замінює тисячі зразків сигналу [3].

Мел-частотні кепстральні коефіцієнти зазвичай використовуються як функції в системах розпізнавання мови, таких як системи, які можуть автоматично розпізнавати номери, промовлені в телефоні. Вони також дедалі частіше знаходять застосування у програмах пошуку музичної інформації, таких як жанрова класифікація тощо.

1.2.4 Підготовка набору даних за допомогою Python

У моєму дослідженні я використовував мову програмування Python. В ній дуже зручно працювати з аудіоданими завдяки бібліотеці librosa. В програмі можна зазначити скільки MFCC потрібно, я використовував 13, та розбив кожен трек на 10 сегментів, щоб не перевантажити нейромережу.

Результатом програми PreparingData.py являється json файл. У файлі роз-

ташований тривимірний масив який являє собою 13 mfcc для кожного фрейму кожного треку нашого набору даних, та ярликами, які показують до якого жанру належить кожна композиція.

РОЗДІЛ 2.
ВПРОВАДЖЕННЯ НЕЙРОННОЇ МЕРЕЖІ ДЛЯ КЛАСИФІКАЦІЇ ЖАНРУ МУЗИКИ

Машинне навчання вміщує у собі багато методів, але я використовую конкретно нейромережу. Нейронна мережа — це послідовність нейронів, з'єднаних між собою синапсами. Синапс (з біології) — місце контакту між двома нейронами, служить для передачі нервового імпульсу між двома клітинами, причому в ході синаптичної передачі амплітуда і частота сигналу можуть регулюватися. Структура нейронної мережі прийшла в світ програмування прямо з біології. Іншими словами, нейромережа — це машинна інтерпретація мозку людини, в якому знаходяться мільйони нейронів передають інформацію у вигляді електричних імпульсів [4].

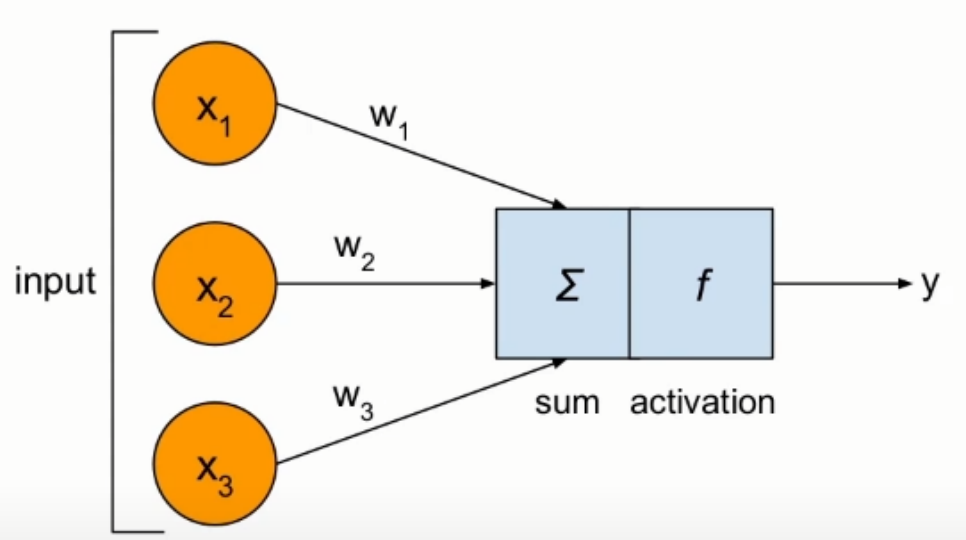

Рис. 6. Структура штучного нейрона

Розглянемо базову структуру штучного нейрона (Рис. 9). На вхід подається декілька значень з певними вагами, пов'язаними з цими різними нейронами. Далі у нас є сам нейрон, який робить кілька речей. Він робить обчислення у вигляді суми вхідних даних та активації за допомогою нелінейною активайійною функцію, потім виводячи результат.

Сума описується такою формулою:

$$h = \sum_{i=1}^{N} x_i \cdot w_i$$

h, що є загальним входом, це сума по всіх вхідних даних, помножена на їх вагу відповідних шляхов. $h = x_1 w_1 + x_2 w_2 + x_3 w_3$.

Тепер розглянемо другу фазу нейрона, де ми маємо саму активацію.

$$y = f(h) = f(x_1 w_1 + x_2 w_2 + x_3 w_3)$$

y, що є виходом нейрона, дорівнює активаційної функції від загального входа.

2.1. Функції активації

Функція активації - це спосіб нормалізації вхідних даних. Тобто, якщо на вході у вас буде велика кількість, пропустивши його через функцію активації, ви отримаєте вихід в потрібному вам діапазоні. Є багато різних функцій активіції.

Я використовую активаційну функцію ReLu.

$$f(x) = max(0, x)$$

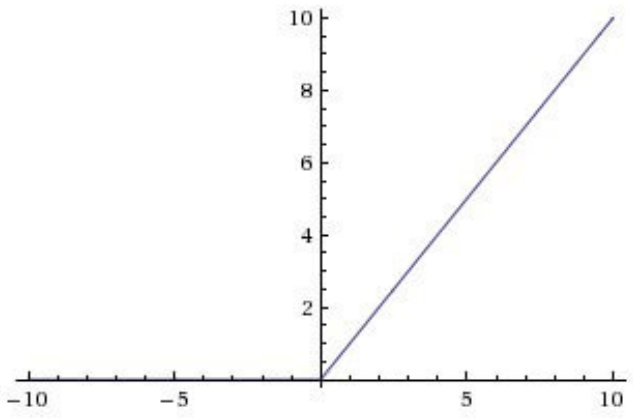

Рис. 7. ReLu

ReLu повертає значення x, якщо x позитивно, і 0 в іншому випадку. Обчислення сигмоїди і гіперболічного тангенса вимагає виконання ресурсномістких операцій, таких як спорудження до рівня, в той час як ReLU може бути реалізований за допомогою простого порогового перетворення матриці активацій в нулі. Крім того, ReLU не схильний до насичення [5].

На вихідному шарі я використовую функцію softmax, яка рахує з яким відсотком цей трек підходить до кожного жанру.

2.2. Структура багатошарового персептрону

Багатошаровий персептрон, який в мене вийшов складається з вхідного шару MFCC, трьох прихованих шарів, кожен по 512, 256, 64 нейронів, та вихідний шар складається з 10 нейронів.

```
def build_mlp_model(input_shape):
    """Генерирует MLP модель
        :param input_shape (tuple): Shape of input set
        :return model: MLP model
        """
    # Строим архитектуру
    model = keras.Sequential([

        # Делаем входной слой
        keras.layers.Flatten(input_shape=input_shape),

        # Первый скрытый слой dense - связянный со всеми нейронами следующего слоя
        keras.layers.Dense(512, activation='relu', kernel_regularizer=keras.regularizers.l2(0.001)),
        keras.layers.Dropout(0.3),

        # 2 скрытый слой
        keras.layers.Dense(256, activation='relu', kernel_regularizer=keras.regularizers.l2(0.001)),
        keras.layers.Dropout(0.3),

        # 3 скрытый слой
        keras.layers.Dense(64, activation='relu', kernel_regularizer=keras.regularizers.l2(0.001)),
        keras.layers.Dropout(0.3),

        # Выходной слой
        keras.layers.Dense(10, activation='softmax')
    ])

    return model
```

Рис. 8. Багатошаровий персептрон у моєму коді на Python

2.3. Структура згорткової нейромережі

Також я вирішив порівняти багатошаровий персептрон з згортковою нейромережою.

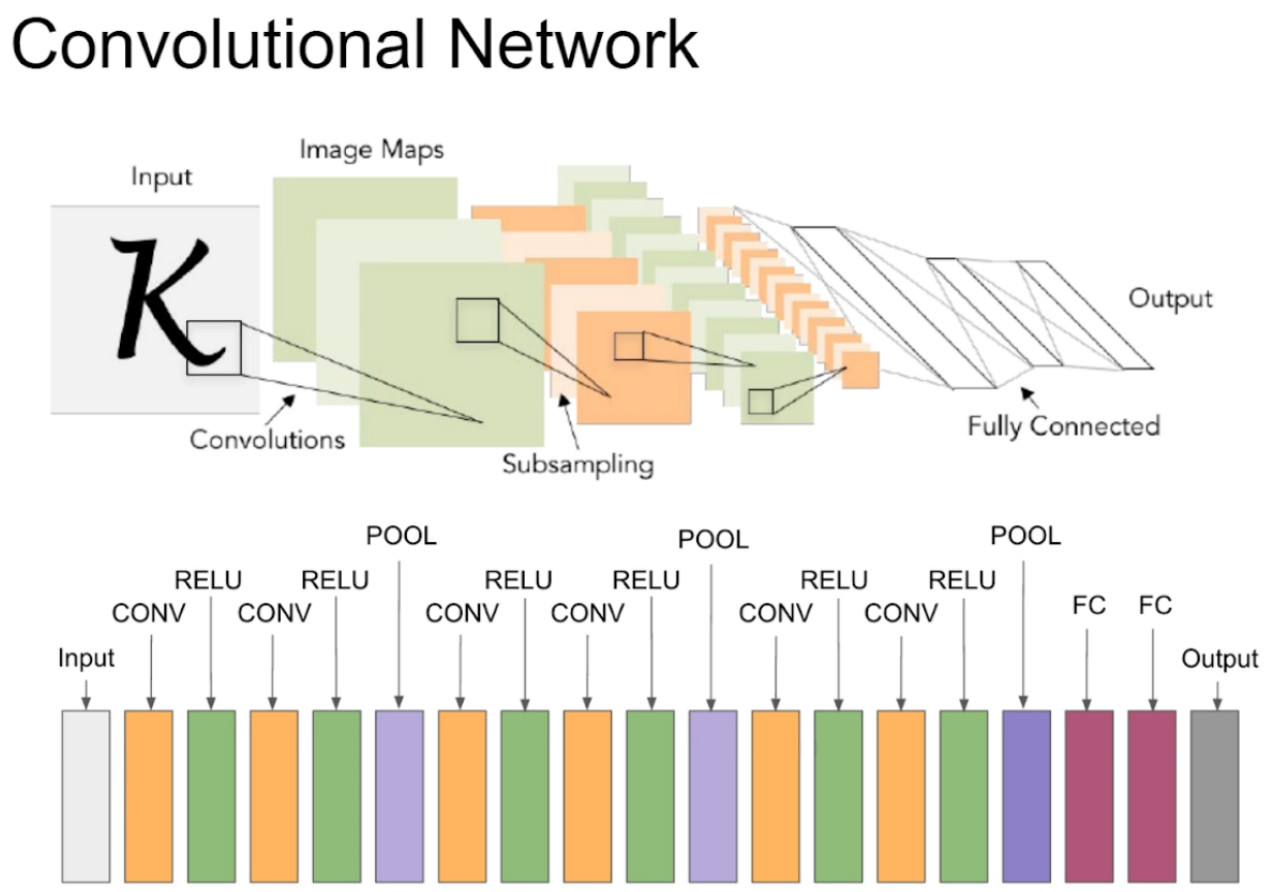

Рис. 9. Принцип праці згорткової нейромережі

Згорткові нейромережі складаються з таких шарів: згорткові та підвибіркові. Які чередують один одного.

Згортковий шар являє собою набір карт, у кожної карти є синаптичне ядро. Ядро являє собою фільтр, який ковзає по всій області попередньої карти і знаходить певні ознаки об’єктів. Це ядро зазвичай 3*3. Тобто на клітинах карт знаходятся нейрони, а на клітинах ядра знаходятся ваги, які треба регулювати.

Підвибірковий шар також, як і згортковий має карти, але їх кількість співпадає з попереднім шаром. Мета шару - зменшення розмірності карт попереднього шару. Зазвичай, кожна карта має ядро розміром 2x2, що дозволяє змен-

шити попередні карти шару в 2 рази. Вся карта ознак поділяється на осередки 2x2 елемента, з яких вибираються максимальні за значенням [6].

Згорткова нейромережа, яка в мене вийшла складається з 3 пар згорткового та підвибіркового шару, кожен по 32 карти, потім, йде звичайний шар з 64 нейронами та вихідний з 10 нейронами.

```
def build_cnn_model(input_shape):
    """Генерирует CNN модель
    :param input_shape (tuple): Shape of input set
    :return model: CNN model
    """

    # создает топологию нейросети
    model = keras.Sequential()

    # 1 сверточный слой
    model.add(keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape=input_shape, padding='same'))
    model.add(keras.layers.MaxPooling2D((3, 3), strides=(2, 2), padding='same'))
    model.add(keras.layers.BatchNormalization())

    # 2 сверточный слой
    model.add(keras.layers.Conv2D(32, (3, 3), activation='relu', padding='same'))
    model.add(keras.layers.MaxPooling2D((3, 3), strides=(2, 2), padding='same'))
    model.add(keras.layers.BatchNormalization())

    # 3 сверточный слой
    model.add(keras.layers.Conv2D(32, (3, 3), activation='relu', padding='same'))
    model.add(keras.layers.MaxPooling2D((2, 2), strides=(2, 2), padding='same'))
    model.add(keras.layers.BatchNormalization())

    # преобразуем в flatten(из 2D в 1D) и подаём в Dense слой
    model.add(keras.layers.Flatten())
    model.add(keras.layers.Dense(64, activation='relu'))
    model.add(keras.layers.Dropout(0.3))

    # выходной слой
    model.add(keras.layers.Dense(10, activation='softmax'))

    return model
```

Рис. 10. Згорткова нейромережа у моєму коді на Python

2.4. Тренування нейромережі

Обі наші моделі нейромережі будуть тренуватися з використанням алгоритму стохастичного градієнтного спуску, та алгоритму Адам, а ваги оновлюються з використанням алгоритму зворотного поширення помилки.

Тепер ми можемо тренувати наші моделі, спочатку поділив усі дані на тре-

нувальні, яких буде 70 відсотків, та тестові, яких буде 30 відсотків.

```
if optimiser == "adam":

    optimiser = keras.optimizers.Adam(learning_rate=0.0001)

elif optimiser == "sgd":

    optimiser = keras.optimizers.SGD(learning_rate=0.001)

# компилируем модель
model.compile(optimizer=optimiser,
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
model.summary()

epochs = int(input("Введіть кількість епох: "))

# тренируем модель
history = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=epochs)
```

Рис. 11. Тренування нейромереж на Python

ВИСНОВКИ

В ході мого дослідження, я прийшов к результатам, які показані на таблиці. Я порівнював дві моделі нейромереж: модель багатошарового персептрону та згорткової нейромережі. Також я порівнював два види оптимізатора: Адам та стохастичний градієнтний спуск. На таблиці показана точність нейромережі в залежності від її виду та оптимізатору. Нейромережі порівнювалися в залежності від точності на тестових даних. Найліпший результат був досягнут згортковою нейромережою використовуючи алгоритм Адам.

Нейромережа Оптимізатор	Багатошаровий перспетрон	Згорткова нейромережа
Стохастичний градієнтний спуск	0.6165388226509094 (500 епох)	0.7045682072639465(100 епох)
Адам	0.6172057390213013(200 епох)	0.7712571024894714(50 епох)

Рис. 12. Таблиця порівнянь нейромереж

На малюнку 13 показан докладний графік навчання згорткової нейромережі. Щоб відстрочити перенавчання був використаний метод dropout, який випадково може виключити декілька нейронів на одну епоху. Якщо ми робимо це, ми збільшуємо стійкість мережі, оскільки мережа не може занадто сильно покладатися на конкретні нейрони. Але можемо помітити після 12 епохи, результати на тестових даних стають гірше, ніж на тренувальних.

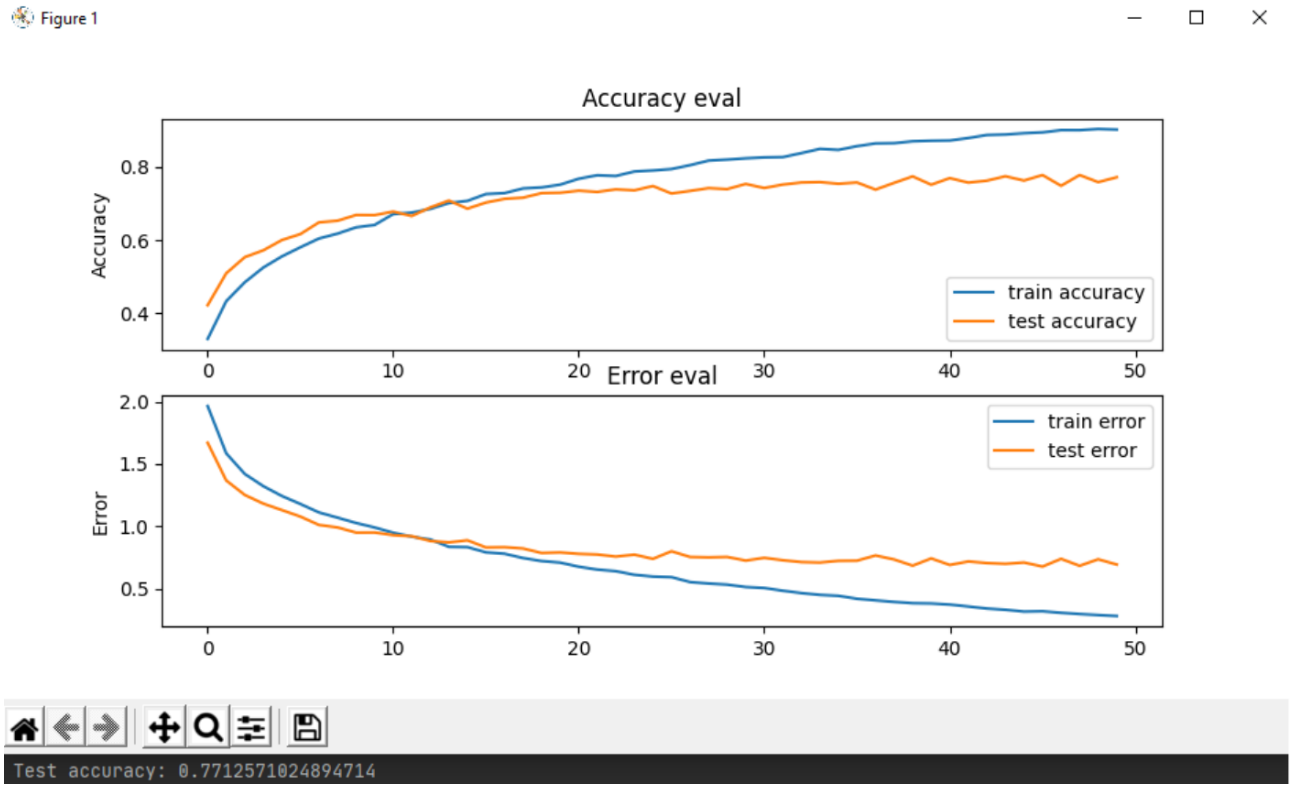

Рис. 13. Характеристика нейронної мережі

Висновок: в ході аналізу результатів нейромереж, для вирішення завдання класифікації музичних композицій за жанрами краще використовувати згорткову нейромережу, найвища точність цієї нейромережі — 77 відсотків.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Звук — Википедия. URL : https://ru.wikipedia.org/wiki/Звук

2. Распознавание речи для чайников / Хабр. URL : https://habr.com/ru/post/226143/

3. Мел-кепстральные коэффициенты (MFCC) и распознавание речи / Хабр. URL : https://habr.com/ru/post/140828/

4. Нейронные сети для начинающих. Часть 1. / Хабр. URL : https://habr.com/ru/post/312450/

5. Это нужно знать: Ключевые рекомендации по глубокому обучению (Часть 2) URL : http://datareview.info/article/eto-nuzhno-znat-klyuchevyie-rekomendatsii-po-glubokomu-obucheniyu-chast-2/

6. Сверточная нейронная сеть, часть 1: структура, топология, функции активации и обучающее множество / Хабр. URL : https://habr.com/ru/post/348000/