Національний технічний університет України
«Київський політехнічний інститут»

Інститут (факультет) ННК "Інститут прикладного системного аналізу"
(повна назва)

Кафедра Системного проектування
(повна назва)

Рівень вищої освіти – другий (магістерський)

Спеціальність 8.05010102 «Інформаційні технології проектування»
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ _____
(підпис) (ініціали, прізвище)
«___»_____20__ р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Галатенко Дмитру Володимировичу
(прізвище, ім'я, по батькові)

1. Тема дисертації Розпізнавання нот піаніно за допомогою нейронних мереж,

науковий керівник дисертації Харченко К.В к.т.н., доц.,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «___»_____ 20__ р. №_____

2. Термін подання студентом дисертації _____

3. Об'єкт дослідження: нейронна мережа для класифікації аудіо контенту

4. Предмет дослідження: класифікація нот піаніно на основі методів штучного інтелекту

5. Перелік завдань, які потрібно розробити

- Проаналізувати існуючі методи, моделі, алгоритми і програмно-апаратні системи аналізу аудіо.
- Розробити алгоритм перетворення аудіо сигналу який дозволить мінімізувати час обробки.
- Розробити модель на основі нейронної мережі для розпізнавання нот піаніно.

- Спроектувати програмні засоби розпізнавання аудіо контенту.
- Розробити стратегію стартап проекту який дозволить реалізувати описану технологію в якості конкурентоспроможного продукту.

6. Орієнтовний перелік ілюстративного матеріалу: презентація на тему "Розпізнавання нот піаніно за допомогою нейронних мереж "

7. Орієнтовний перелік публікацій

Galatenko D. V. Piano note recognition using neural networks / Galatenko : Матеріали XIV всеукраїнської науково – практичної студентської конференції, 22 травня 2017, Київ, Україна : матеріали. – К. : НТУУ «КПІ», 2017. – С. 151.

8. Консультанти розділів дисертації*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

9. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Отримання завдання	30.09.2016	
2	Збір інформації	22.01.2017	
3	Розробка алгоритму та структури програми	11.03.2017	
4	Розробка плану оцінювання	06.04.2017	
5	Розробка програмної моделі	11.05.2017	
6	Тестування додатку та отримання даних	30.05.2017	
7	Отримання допуску до захисту та подача роботи в ДЕК	12.06.2017	

Студент _____ ___Д.В. Галатенко___
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ ___К. В. Харченко_
(підпис) (ініціали, прізвище)

* Консультантом не може бути зазначено наукового керівника магістерської дисертації.

РЕФЕРАТ

магістерської дисертації Галатенка Дмитра Володимировича
на тему «Розпізнавання нот піаніно за допомогою нейронних мереж»

Дана магістерська дисертація присвячена дослідженню розпізнавання нот за допомогою нейронних мереж. Метою роботи є розробка програмного забезпечення для класифікації нот піаніно.

В роботі розглянуто засоби обробки та класифікації аудіосигналів, та визначено найкращий засіб для реалізації програмного продукту. Розроблений алгоритм перетворення аудіосигналу який дозволить мінімізувати час обробки, розроблена модель на основі повнозв'язної нейронної мережі для класифікації звучання нот піаніно, а також розроблена стратегія стартап-проекту, яка дозволить реалізувати описану технологію в якості конкурентоспроможного продукту.

Отже, актуальною науково-прикладною проблемою є створення підходу до розпізнавання звуків піаніно за допомогою систем штучного інтелекту для побудови часово-нотної діаграми натискання клавіш піаніно у режимі реального часу.

Об'єкт дослідження – нейронна мережа для класифікації аудіо контенту. Предмет дослідження – класифікація нот піаніно на основі методів штучного інтелекту.

Мета і задачі дослідження. Метою роботи є розроблення системи розпізнавання нот піаніно задля навчання користувача грі на музичному інструменті без будь-яких музичних знань чи освіти. Розпізнавання відбувається за рахунок використання методів штучного інтелекту таких як нейронні мережі.

Загальний обсяг роботи: 76 сторінок, 10 рисунків, 31 таблиця, 18 бібліографічних найменувань.

Ключові слова: машинне навчання, нейронні мережі, класифікація нот піаніно.

РЕФЕРАТ

магистерской диссертации Галатенко Дмитрия Владимировича
на тему «Распознавание нот пианино с помощью нейронных сетей»

Данная магистерская диссертация посвящена исследованию распознавания нот с помощью нейронных сетей. Целью работы является разработка программного обеспечения для классификации нот пианино.

В работе рассмотрены средства обработки и классификации аудиосигналов, и определено лучшее средство для реализации программного продукта. Разработанный алгоритм преобразования аудиосигнала который позволит минимизировать время обработки, разработана модель на основе полносвязной нейронной сети для классификации звучания нот пианино, а также разработана стратегия стартап-проекта, которая позволит реализовать описанную технологию в качестве конкурентоспособного продукта.

Итак, актуальной научно-прикладной проблемой является создание подхода к распознаванию звуков пианино с помощью систем искусственного интеллекта для построения временно-нотной диаграммы нажатия клавиш пианино в режиме реального времени.

Объект исследования - нейронная сеть для классификации аудио контента. Предмет исследования - классификация нот пианино на основе методов искусственного интеллекта.

Цель и задачи исследования. Целью работы является разработка системы распознавания нот пианино для обучения пользователя игре на музыкальном инструменте без каких-либо музыкальных знаний или образования. Распознавание происходит за счет использования методов искусственного интеллекта как нейронные сети.

Общий объем работы: 76 страниц, 10 рисунков, 31 таблица, 18 библиографических наименований.

Ключевые слова: машинное обучение, нейронные сети, классификация нот пианино.

ABSTRACT

for master's thesis of Dmitry Vladimirovich Galatenko

On "Piano notes recognition using neural networks"

This master's thesis is devoted to the study of piano notes recognition with the help of neural networks. The goal of the research is the development of software for piano notes classification.

The research deals with processing and classification of audio signals, also the best way for implementing software product is determined. The developed audio signal conversion algorithm that will minimize processing time, a model based on a fully-connected neural network to classify the sound of piano notes, and also a start-up project strategy that will allow the described technology to be released as a competitive product.

Thus, the relevant scientific and applied problems is to create recognition approach to piano sounds by means of artificial intelligence to build time-charts musical piano keystrokes in real time.

Object of research - neural network to classify audio content. Purpose of the study - Classification piano music based on artificial intelligence techniques.

The purpose and objectives of the study. The aim is to develop piano music recognition system for training users play a musical instrument without any musical knowledge or education. Recognition is due to the use of artificial intelligence techniques such as neural networks.

Total amount of research: 76 pages, 10 figures, 31 tables, 18 bibliographical names.

Key words: machine learning, neural networks, piano music classification.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

НМ (англ. NN – Neural Network)	нейронна мережа
NLP – Natural Language Processing	обробка природної мови
FFT – Fast Furier Transform	швидке перетворення Фур’є
ВП (англ. WT – wavelet transform)	вейвлет перетворення
SVM – Support Vector Machine	метод опорних векторів
ПЗ (англ. SW – software)	програмне забезвечення
ReLU – The Rectified Linear Unit	прямокутний лінійний елемент
MFCC – mel-frequency cepstral feature	мел-частотні кепстральні хар-ки.
PCA – principal component analysis	метод головних компонент
JSON	JavaScript Object Notation
MIDI – Musical Instrument Digital Interface	цифровий інтерфейс музичних інструментів

ВСТУП

Інтенсивний розвиток комп'ютерної техніки зумовив широке застосування систем та засобів штучного інтелекту в різних галузях науки і техніки. Створення інтелектуального комп'ютера передбачає поєднання роботи п'яти паралельних каналів оброблення інформації – зорового, слухового, нюхового, смакового і тактильного. Велика увага в системах штучного інтелекту приділяється опрацюванню візуальної інформації, в той час як всі інші поки що знаходяться далеко позаду в розвитку технології. Вибір етми обумовлений широким розповсюдженням та великою інформативністю слухових образів а також тим фактом що розпізнавання голосових образів є менш вивченою задачею у порівнянні з візуальними.

Сьогодні серед інтелектуальних систем значного поширення набули системи, так званого, комп'ютерного слуху, в яких актуальними задачами є задачі розпізнавання (аналізу) звукових послідовностей. Стрімке розповсюдження інтернет–технологій, візуалізація в мистецтві, науці й техніці зумовили розвиток комп'ютерної обробки природньої мови, де основними задачами є правила інтерпретація почутого та реалістичне відтворення (синтез) звукових образів. Поєднання цих двох задач надає людству потужні рішення у вигляді існуючих на сьогодні голосових асистентів та розумних помічників як Siri, Cortana, Google Assistant та інші.

Основні задачі аналізу аудіо – це обробка природної мови та класифікація звукових образів. Загальновживаними є такі методи розпізнавання образів: методи цифрової обробки даних, які найбільш трудомісткі з точки зору обчислювальної складності за рахунок великої кількості семплів за секунду; статистичні, де основною проблемою є знаходження функцій умовної густини розподілу імовірності значень ознак для кожного класу; методи, які базуються на штучних нейронних мережах, складність яких полягає у виборі архітектури мережі та алгоритмів її навчання; структурні та синтаксичні, в яких важко

формалізується задача визначення граматик на множині висловлювань, що породжують мову.

Розуміння природної мови іноді вважають AI-повною задачею, тому що розпізнавання живої мови потребує величезних знань системи про навколишнє середовище та можливості взаємодіяти з ним. Саме означення змісту слова «розуміти» — одна з головних задач штучного інтелекту. В наш час значну роль у вирішенні задач з обробки природномовних даних відіграють онтології, наприклад, WordNet, UWN. У процесі дослідження обробки природної мови було досягнуто значних результатів, серед яких розробка потужних лексикографічних систем, програм для машинного перекладу, електронних словників та ін. Однак, існує проблема, яка досі не знайшла свого вирішення, вона корініться у самій природі людської мови. Проблема розуміння людського мовлення полягає саме у його неоднозначності. Можна виділити наступні види неоднозначностей:

1) Синтаксична неоднозначність: у прислів'ї «Час — не кінь, не підженеш і не зупиниш» для обробки природної мови буде абсолютно неясним те, про що саме йдеться у реченні, про коня чи про час.

2) Смислова неоднозначність: у питанні «Де знайти ключ до того замку?» слово замок може мати два абсолютно різні значення, зважаючи на поставлений наголос.

3) Відмінкова неоднозначність: у фразах «Усі були схвильовані перед концертом» та «Не треба давати перед!» слово перед означає час або місце, що абсолютно змінює сенс фрази.

4) Референційна неоднозначність: у фразі «Відкрий поличку та дістань мокру парасольку, я хочу її висушити» займенник її за смисловим значенням матиме відношення до мокрої парасольки, проте для машини, у якої повністю відсутнє розуміння реальності, даний займенник відноситиметься як до полички, так і до парасольки.

Одним із викликів, який виникає у процесі обробки природної мови, можна вважати проблему синонімії, в результаті якої одне поняття може бути

вираженим декількома різними словами. Як наслідок, релевантні документи, в яких використано синоніми понять, що було вказано користувачем у запиті, може бути не визначено системою.

Вплив вище перелічених явищ є особливо відчутним при створенні систем машинного перекладу. Проблема полягає у складності встановлення конкретного відображення дійсної семантико-синтаксичної структури речення у його внутрішнє логічне уявлення, яке автоматично генерується системою. Розв'язання таких типів неоднозначностей можливе за допомогою введення додаткових значень, які збільшать знання програми про ту чи іншу галузь. Сьогодні програм, які «розуміють» усі типи неоднозначностей у великому спектрі галузей, не існує, проте є програми, що можуть коректно реагувати на неоднозначності у дуже вузьких сферах.

Для задач класу NLP характерним є розбиття образів на фонеми. Далі ці фонеми класифікуються і утворюють собою набір звуків які розбиваються на слова. Для пошуку повторюваних елементів виокремлено методи знаходження множини ознак елементів звуку і побудову гіпотези про їхні зв’язки, які базуються на відповідності певній параметричній моделі, та методи, які ґрунтуються на застосуванні автокореляційної функції. Отже, вищевказані підходи до синтезу та аналізу звуків мають різне теоретичне підґрунтя, що ускладнює їхнє використання для задач аналізу та породження різноманітних звукових структур.

Розповсюдженим способом роботи з аудіопослідовностями є підхід базований на цифровій обробці сигналів. За допомогою математичних алгоритмів вхідна послідовність семплів перетворюється в деякий інший сигнал, який має необхідні властивості. Процес перетворення сигналів називається фільтрацією, а пристрій, що виконує фільтрацію, називається фільтр. Оскільки значення сигналів надходять з постійною швидкістю, фільтр повинен встигати обробляти поточний сигнал серії до надходження наступного (частіше — до надходження наступних n семплів, де n — затримка фільтра), тобто обробляти сигнал в реальному часі. Для обробки сигналів (фільтрації) в

реальному часі застосовують спеціальні обчислювальні пристрої — цифрові сигнальні процесори. Це повністю стосується не тільки безперервних сигналів, але і перервних, а також до сигналів, записаних на пристрої зберігання інформації. В останньому випадку швидкість обробки непринципова, так як при повільній обробці дані не будуть втрачені.

Проте такий підхід має недоліки – по-перше, він не є гнучким з точки зору тривалості сигналу, по-друге він зав’язаний на великій кількості числових значень – якщо мова йде про аудіо без втрати якості це 44100 семплів за секунду.

З огляду на названі проблеми необхідно створити підхід до класифікації аудіо сигналів який дозволив би опрацьовувати подану вхідну послідовність за час припустимий для роботи в режимі реального часу. Це дозволить будувати ноти для нерозмічених музичних композицій, навчати користувачів грі на піаніно навіть без будь-якої музичної освіти. Навчання грі на піаніно базується на побудові часової діаграми натискання клавіш. На основі отриманої інформації про звук, який має місце в даний момент часу, нейронна мережа буде співставляти відповідний йому сигнал до клавіші піаніно, яку треба натиснути в певний проміжок часу.

Отже, актуальною науково-прикладною проблемою є створення підходу до розпізнавання звуків піаніно за допомогою систем штучного інтелекту для побудови часово-нотної діаграми натискання клавіш піаніно у режимі реального часу.

Об’єктом дослідження – нейронна мережа для класифікації аудіо контенту.

Предмет дослідження – класифікація нот піаніно на основі методів штучного інтелекту.

Мета і задачі дослідження. Метою роботи є розроблення системи розпізнавання нот піаніно задля навчання користувача грі на музичному інструменті без будь-яких музичних знань чи освіти. Розпізнавання

відбувається за рахунок використання методів штучного інтелекту таких як нейронні мережі.

Для досягнення поставленої мети необхідно розв’язати такі задачі:

- проаналізувати існуючі методи, моделі, алгоритми і програмно-апаратні системи аналізу аудіо;
- розробити алгоритм перетворення аудіосигналу який дозволить мінімізувати час обробки аудіосигналу;
- розробити модель на основі нейронної мережі для класифікації звучання нот піаніно;
- спроектувати програмний додаток розпізнавання аудіо контенту;
- розробити стратегію стартап-проекту, яка дозволить реалізувати описану технологію в якості конкурентоспроможного продукту.

1 ДОСЛІДЖЕННЯ ІСНУЮЧИХ МЕТОДІВ ТА ЗАСОБІВ РОЗПІЗНАВАННЯ МУЗИЧНИХ ДАНИХ

1.1 Класифікація аудіо за допомогою часово-частотного аналізу

Найбільше широко використовуваним методом обробки цифрових сигналів наразі є перетворення Фур'є. Однак воно має ряд недоліків, які привели до появи й розробки нових удосконалених методів цифрового аналізу нестаціонарних сигналів. Найбільшим недоліком перетворення Фур'є можна назвати усереднення характерних рис по всій тривалості сигналу, що робить неможливим застосування даного методу при необхідності аналізу змін сигналу в часі. Для аналізу швидкоплинних нестаціонарних сигналів доцільно використовувати сучасний математичний засіб вейвлет-перетворення [4]. Вейвлет-перетворення на сьогодні широко застосовується для аналізу графічних зображень, аудіо та відео даних, різноманітних технічних сигналів тощо. Цей інструмент дозволяє отримати частотно-часове подання сигналу, що аналізується. В задачах технічної діагностики для визначення стану технічного об’єкту використовується спектральний аналіз складних нестаціонарних сигналів. Для вирішення задачі технічної діагностики в умовах реального часу постає задача слідкування за частотними характеристиками об’єкту шляхом аналізу коротких послідовностей сигналів. У разі виконання спектрального аналізу сигналів за допомогою вейвлетів необхідним є дослідження частотних характеристик материнського вейвлету, що використовується для аналізу. Параметри обраного вейвлету та його налаштування можуть суттєво впливати на результати перетворення й можливість їх правильної інтерпретації. Постановка задачі та мета дослідження.

1.1.1 Вейвлет-перетворення сигналу з точки зору фільтрації

Безперервне вейвлет-перетворення – це розклад сигналу, що аналізується, в базисі деякої материнської вейвлетної функції.

Базис вейвлет-перетворення (ВП) будується шляхом масштабних перетворень та переносів материнського вейвлету з неперервними значеннями базисних параметрів – масштабного коефіцієнту a та параметра зсуву b:

$$\psi_{a,b}(t) = |a|^{-1/2}\,\psi\left(\frac{t-b}{a}\right) \qquad (1)$$

де $\psi_{a,b}(t)$ – дочірній вейвлет базису, t – параметр часу, a, b є R.

В інтегральній формі ВП безперервного сигналу s(t) може бути подано як:

$$W(a,b) = \frac{1}{\sqrt{|a|}}\int_{-\infty}^{\infty} s(t)\psi^{*}_{a,b}(t)dt \qquad (2)$$

де * – оператор комплексного спряження. Використовуючи рівняння Парсеваля, що описує зв'язок між функціями та їхніми образами Фур'є, вираз (2) може бути записаний в альтернативній формі:

$$W(a,b) = \sqrt{a}F^{-1}\left\{ S(f)\Psi^{*}(af)\right\}, \qquad (3)$$

де F^{-1} – оператор зворотного перетворення Фур'є;

$\psi(af)$ – образ Фур'є дочірнього вейвлету;

S(f) – образ Фур'є сигналу;

f – частота сигналу.

Таке подання дозволяє прискорити час виконання вейвлет-коефіцієнтів за рахунок використання швидкого перетворення Фур'є.

Вираз (3) показує, що вейвлет-перетворення може бути розглянуте як спеціальна операція фільтрації з вейвлетом в якості ядра фільтру. Розглянемо особливості вейвлетних фільтрів, використовуючи їхні частотно-часові характеристики [11].

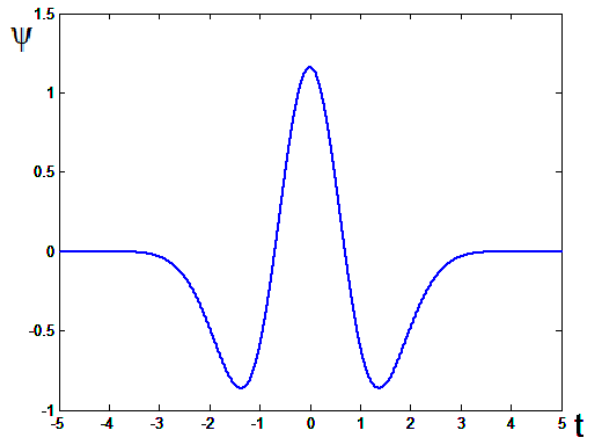
Нехай вейвлет $\psi(t)$ та його образ Фур'є $\Psi(f)$ є функціями-вікнами з центрами t_0 та f_0, радіусами δ_t та δ_f відповідно. Розмір частотно-часового вікна,

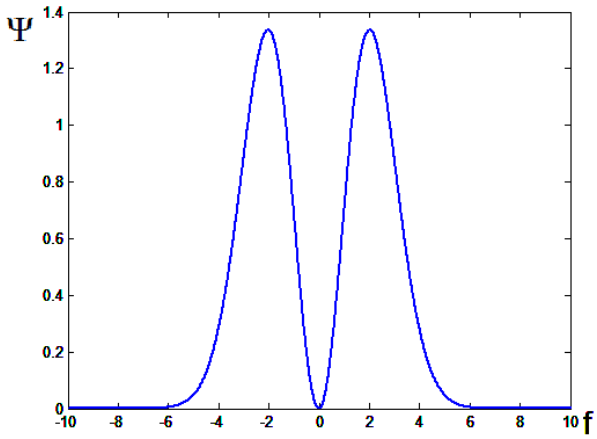
що характеризує, роздільну здатність вейвлет-аналізу, дорівнює $4*\delta_t\delta_f$. При масштабуванні материнського вейвлету радіус вікна дочірнього вейвлету в часовій області збільшується прямо пропорційно масштабу ($a* \delta_t$), а в частотній – радіус вікна зменшується (δ_f /a). Таким чином, ВП надає змінну роздільну здатність в частотно-часовій площині, що дає значну перевагу при аналізі нестаціонарних сигналів. Завдяки властивостям зсуву та масштабування материнського вейвлету, кожний дочірній вейвлет в частотній області на масштабі a буде мати ширину вікна, що дорівнює $2*\delta_f/a$, та центральну частоту, що дорівнює f_0/a. Їхнє відношення не залежить від масштабу перетворення і є постійною величиною $2*\delta_f/f_0$, значення якої залежить від параметрів материнського вейвлету.

Таким чином, ВП можна розглядати як операцію смугової фільтрації з постійною добротністю, коли відношення ширини смуги пропускання до центральної частоти фільтру є постійною величиною.

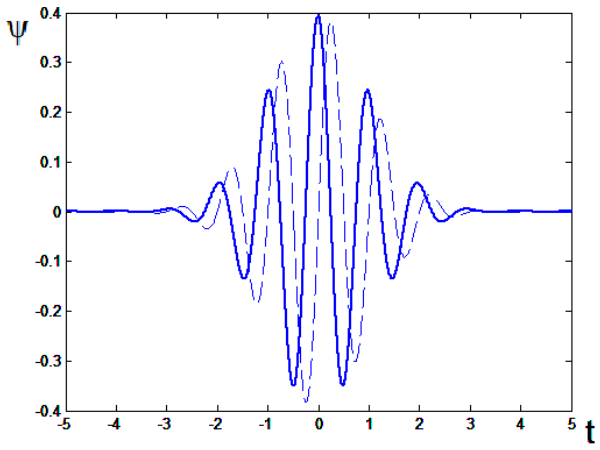
Серед існуючих вейвлет-функцій, що використовуються в безперервному вейвлет-аналізу, вейвлети сімейства Гауса, а саме вейвлет Морлета вейвлет Мексиканський капелюх (Рис. 1), більш за все подібні за формою імпульсним складовим нестаціонарних сигналів. Цей факт робить Гаусове сімейство вейвлетів найбільш відповідним засобом аналізу таких сигналів.

a





б



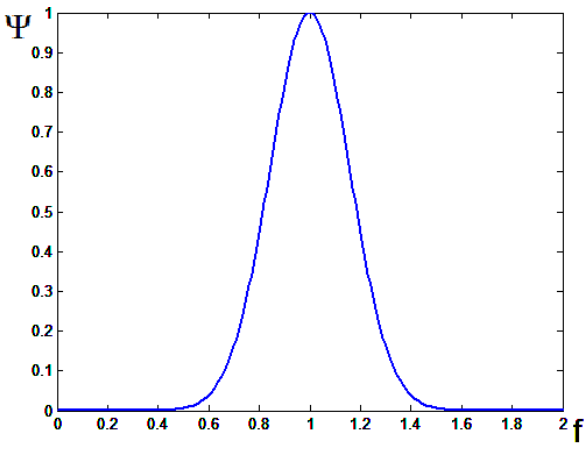


Рис. 1 – Вейвлети Морле (а) та Мексиканський капелюх(б)

Всі вейвлети сімейства Гауса будуються на основі похідних функції Гауса. Наприклад, принципова різниця між вейвлетом Морле і Мексиканським капелюхом в тому, що вейвлет Море має кращі вибіркові властивості в частотній області, а вейвлет Мексиканський капелюх – в часовій. В силу принципу невизначеності Гейзенберга неможливо отримати ідеальну локалізацію одночасно в частотній та часовій областях.

В часовій області комплексний вейвлет Морле представляє собою комплексну експоненту, що модулюється функцією Гауса:

$$\psi(t) = \frac{\sigma}{\sqrt{\pi}} e^{-\sigma^2 t^2} e^{i2\pi f_0 t} \qquad (4)$$

де t – час, σ і f_0 – параметри, що задають форму вейвлета; їхній зміст стає зрозумілим в частотному поданні вейвлету.

В частотній області вейвлет Морлема є форму Гаусова вікна з центральною частотою f_0 і шириною σ:

$$\Psi(f) = \Psi^*(f) = e^{-(\pi^2/\sigma^2)(f-f_0)^2} \qquad (5)$$

де: Ψ(f) – перетворення Фур'є вейвлету Морле;

f_0 – центральна частота вейвлету;

σ – ширина смуги частот, що аналізується;

* – означає комплексне спряження.

Перетворення Фур'є вейвлету Морле дорівнює нулю для відмінних частот, що дозволяє розділити фазові та амплітудні компоненти сигналу при виконанні вейлвет-перетворення. Таким чином, частотна смуга, що покривається вікном вейвлету Морле, обмежена інтервалом $[f_0 - \sigma/2, f_0 + \sigma/2]$.

На рис. 1, б наведено приклад подання вейвлета Морле в часовій та частотній областях при заданих параметрах $\sigma=0,7$ та $f_0=1$. Вейвлет Морле не має компактного носія, однак найбільша частина його енергії зосереджена на інтервалі, ширина якого визначається його смугою пропускання. На практиці при f_0 близькій до нуля вейвлет Морле може бути використаний з мінімальною похибкою.

1.1.2 Метод спектрального аналізу сигналів за допомогою вейвлет-перетворення

Для розділення частотних компонент сигналів використовують банки фільтрів[11]. Як було відзначено, вейвлет-перетворення сигналів можна розглядати як операцію фільтрації з постійною відносною смугою частот. Тому по аналогії зі звичайними гребінками з 1/n–октавних смугових фільтрів[2] пропонується побудувати набір аналізуючих вейвлетних фільтрів та аналізувати сигнали за наступним алгоритмом.

Етап 1. Задати необхідну кількість фільтрів на октаву (k = 1, 2, 3, 4, 6, 12).

Етап 2. Для побудови набору фільтрів із k фільтрами на октаву задати основу для розрахунку масштабів вейвлет-перетворення: $a^i = (2^{1/k})^i$

Етап 3. Параметр добротності обрати таким чином, щоб ширина діапазону кожного фільтра дорівнювала відстані між фільтрами: $Q_W = 2^{1/k} - 1$.

За допомогою цього параметра можна варіювати ширину смуги пропущення фільтрів і ступіньперекриття фільтрів у банку.

Етап 4. Розрахувати параметри фільтрів за допомогою константи Q_W:

$f_i = f_0 / a_i$ та $\sigma_i = f_i Q_W$.

Для параметру материнського вейвлету f_0 максимальним приймемо значення $f_0 = 0.8 * f_{Nyq}$, де f_{Nyq} – частота Найквиста сигналу.

Етап 5. За допомогою виразу (5) побудувати набір фільтрів.

Етап 6. За виразом (3) виконується вейвлет фільтрація сигналу. При цьому використовується розроблений метод зменшення впливу граничних ефектів. Завдяки тому, що вейвлет Морле є комплексним, вейвлет-коефіцієнти, отримані в результаті перетворення на кожному масштабі, є також комплексними значеннями. Реальна частина вейвлет коефіцієнтів Wre (a,b) представляє собою відфільтрований сигнал, а мнима частина Wim(a, b) – є фазою відфільтрованого сигналу.

Етап 7. Розрахувати модуль вейвлет коефіцієнтів (енергію) за формулою:

$$EW(a,b) = \left[W_{re}(a,b)^2 + W_{im}(a,b)^2 \right]^{1/2} \qquad (6)$$

Даний модуль може бути інтерпретований як обвідна вейвлет-перетворення, що дозволяє провести демодуляцію сигналу у необхідному частотному діапазоні.

Етап 8. Для аналізу частотних характеристик сигналу пропонується аналізувати спектр обвідної вейвлет-коефіцієнтів на одному найліпшому масштабі або взаємний спектр на парі масштабів.

У випадку аналізу аудіо сигналів пропонується розглядати взаємний спектр на перших масштабах, які відповідають фільтрам високих частот. Такий підхід представляє собою модифікацію методу аналізу обвідної високочастотної випадкової вібрації.

При побудові системи аналізу сигналів в реальному часі та налаштуванні набору аналізуючих вейвлетних фільтрів слід враховувати компроміс між вимогами до необхідної точності обробки та опису характеристик сигналу та вимогами до системи обробки аудіо сигналів.

1.2 Сучані методи машинного навчання в задачах класифікації

Машинне навчання стало досить розповсюдженим в нашому суспільстві останніми роками. Воно успішно застосовується в задачах пошуку, комп'ютерного зору, медицини, безпілотного керування дронів та автомобілів. Ядром для багатьох з цих додатків є методи та засоби штучного інтелекту, такі як класифікація, локалізація і виявлення. Останні розробки в області нейронних мереж (так зване «глибоке навчання») значно просунули продуктивність цих візуальних систем розпізнавання. Цей розділ націлений на розгляд існуючих методів глибокого машинного навчання, архітектур нейронних мереж з акцентом на вивчення моделей для вирішення цих завдань, зокрема, задачі класифікації.

1.2.1 Оптимізація за допомогою стохастичного градієнтного спуску

Найвідоміший варіант алгоритму навчання нейронної мережі - так званий алгоритм зворотного поширення помики [17]. Існують сучасні алгоритми другого порядку, такі як метод сполучених градієнтів і метод Левенберга-Маркара [16], які на багатьох завданнях працюють істотно швидше (іноді на порядок). Алгоритм зворотного поширення найбільш простий для розуміння, а в деяких випадках він має певні переваги. Розроблено також евристичні модифікації цього алгоритму, добре працюють для певних класів задач, - швидке поширення (Fahlman, 1988) і Дельта-дельта з межею (Jacobs, 1988)

В алгоритмі зворотного поширення обчислюється вектор градієнту поверхні помилок. Цей вектор вказує напрямок найкоротшого спуску по поверхні з даної точки, тому якщо ми "трохи" просунемося по ньому, помилка зменшиться. Послідовність таких кроків (сповільнюється в міру наближення до екстремуму) врешті-решт призведе до мінімуму того чи іншого типу. В процесі використання виникає декілька питань:

1) яку потрібно брати довжину кроків;

2) як боротися із потраплянням в точку локального екстремуму;

3) можливе перенавчання мережі.

При великій довжині кроку збіжність буде швидшою, але є небезпека перестрибнути через рішення або (якщо поверхня помилок має особливо химерну форму) піти в неправильному напрямку. Класичним прикладом такого явища при навчанні нейронної мережі є ситуація, коли алгоритм дуже повільно просувається по вузькому яру з крутими схилами, стрибаючи з одного його боку на іншу (Рис. 2). Навпаки, при маленькому кроці, ймовірно, буде схоплено вірний напрям, однак при цьому потрібно дуже багато ітерацій.

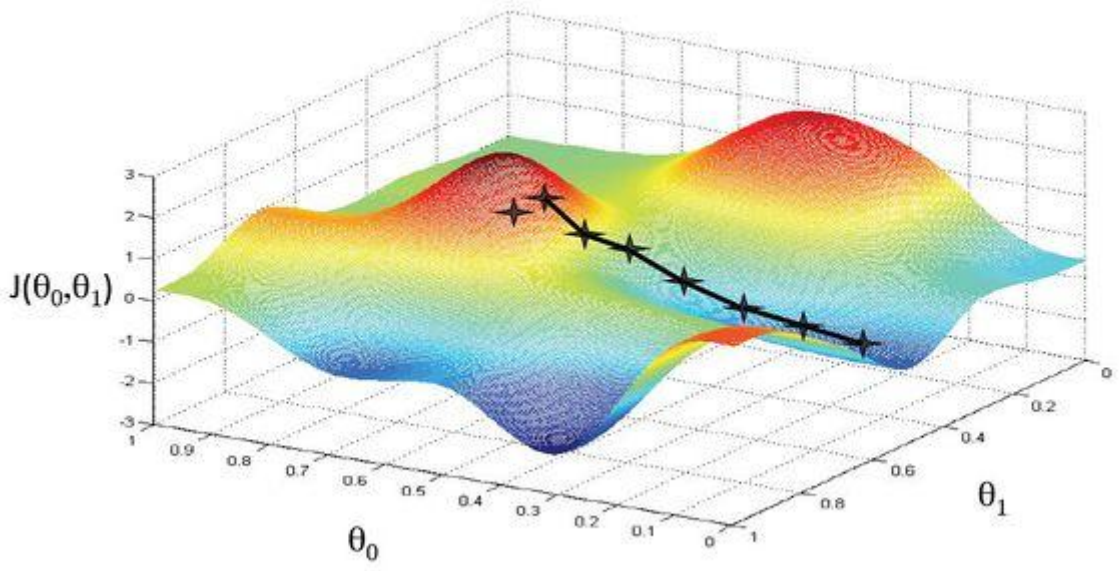


Рис. 2 – Демонстрація роботи методу градієнтного спуску

На практиці величина кроку береться пропорційної крутизні схилу (так що алгоритм уповільнює хід поблизу мінімуму) з деякою константою, яка називається швидкістю навчання. Правильний вибір швидкості навчання залежить від конкретного завдання і зазвичай здійснюється досвідченим шляхом; ця константа може також залежати від часу, зменшуючись у міру просування алгоритму.

Зазвичай цей алгоритм видозмінюється таким чином, щоб включати доданок імпульсу (або інерції). Цей член сприяє просуванню в фіксованому напрямку, тому якщо було зроблено кілька кроків в одному і тому ж напрямку, то алгоритм "збільшує швидкість", що (іноді) дозволяє уникнути локального мінімуму, а також швидше проходити плоскі ділянки.

Таким чином, алгоритм діє ітеративно, і його кроки прийнято називати епохами. На кожній епосі на вхід мережі по черзі подаються всі навчальні спостереження, вихідні значення мережі порівнюються з цільовими значеннями і обчислюється помилка. Значення помилки, а також градієнту поверхні помилок використовується для коригування ваг, після чого всі дії повторюються. Початкова конфігурація мережі вибирається випадковим чином, і процес навчання припиняється або коли пройдено певну кількість епох, або коли помилка досягне деякого певного рівня малості, або коли помилка перестане зменшуватися (користувач може сам вибрати необхідна умова зупинки).

Одна з найбільш серйозних труднощів викладеного підходу полягає в тому, що таким чином ми мінімізуємо не ту помилку, яку насправді потрібно мінімізувати, - помилку, яку можна очікувати від мережі, коли їй будуть подаватися абсолютно нові спостереження. Інакше кажучи, ми хотіли б, щоб нейронна мережа була здатна узагальнювати результат на нові спостереження. Насправді мережа навчається мінімізувати помилку на навчальній множині, і під час відсутності ідеального і нескінченно великого навчальної множини це зовсім не те ж саме, що мінімізувати "справжню" помилку на поверхні помилок в заздалегідь невідомої моделі явища (Bishop, 1995).

Найсильніше ця відмінність виявляється в проблемі перенавчання, або занадто близькою підгонки. Це явище простіше буде продемонструвати не для нейронної мережі, а на прикладі апроксимації за допомогою поліномів, - при цьому суть явища абсолютно та ж.

Мережі з великим числом ваг моделюють більш складні функції і, отже, схильні до перенавчання. Мережа ж з невеликим числом ваг може виявитися

недостатньо гнучкою, щоб змоделювати наявну залежність. Наприклад, мережа без проміжних шарів насправді моделює звичайну лінійну функцію.

Як же вибрати "правильну" ступінь складності для мережі? У більшості випадків більш складна мережа дає меншу помилку, але це може свідчити не про хорошу якість моделі, а про перенавчання.

Відповідь полягає в тому, щоб використовувати механізм контрольної крос-перевірки. Ми резервуємо частина навчальних спостережень і не використовуємо їх в навчанні за алгоритмом зворотного поширення. Замість цього, у міру роботи алгоритму, вони використовуються для незалежного контролю результату. На самому початку роботи помилка мережі на навчальному та контрольному наборі даних буде однаковою (якщо вони істотно відрізняються, то, ймовірно, розбиття всіх спостережень на два набори даних було неоднорідне). У міру того, як мережа навчається, помилка навчання, природно, зменшується, і, поки навчання зменшує дійсну функцію помилок, помилка на контрольному наборі також буде спадати. Якщо ж контрольна помилка перестала зменшуватися або навіть стала рости, це вказує на те, що мережа почала занадто близько апроксимувати дані і навчання слід зупинити. Це явище надто точної апроксимації в процесі навчання і називається перенавчанням. Якщо таке трапилося, то зазвичай радять зменшити число прихованих елементів і / або шарів, бо мережа є занадто потужної для даного завдання. Якщо ж мережа, навпаки, була взята недостатньо багатою для того, щоб моделювати наявну залежність, то перенавчання, швидше за все, не відбудеться, і обидві помилки - навчання і перевірки - не досягнуть достатнього рівня малості.

Описані проблеми з локальними мінімумами і вибором розміру мережі призводять до того, що при практичній роботі з нейронними мережами, як правило, доводиться експериментувати з великим числом різних мереж, часом навчаючи кожну з них по кілька разів (щоб не бути введеним в оману локальними мінімумами) і порівнюючи отримані результати. Головним показником якості результату є тут контрольна помилка. При цьому, відповідно

до загальнонаукових принципом, згідно з яким при інших рівних слід віддати перевагу більш просту модель, з двох мереж з приблизно рівними помилками контролю має сенс вибрати ту, яка менше.

Необхідність багаторазових експериментів веде до того, що контрольне безліч починає відігравати ключову роль у виборі моделі, тобто стає частиною процесу навчання. Тим самим послаблюється його роль як незалежного критерію якості моделі - при великій кількості експериментів є ризик вибрати "вдалу" мережу, що дає хороший результат на контрольному безлічі. Для того, щоб надати остаточної моделі належну надійність, часто (принаймні, коли обсяг навчальних даних це дозволяє) надходять так: резервують ще одне - тестове безліч спостережень. Підсумкова модель тестується на даних з цього безлічі, щоб переконатися, що результати, досягнуті на навчальному та контрольному множинах реальні, а не є артефактами процесу навчання. Зрозуміло, для того щоб добре грати свою роль, тестове безліч повинно бути використано тільки один раз: якщо його використовувати повторно для коригування процесу навчання, то воно фактично перетвориться на контрольну множину.

1.2.2 Метод зворотного поширення помилки

Метод зворотного поширення помилки (backpropagation) — це ітеративний градієнтний алгоритм, який використовується з метою мінімізації помилки роботи багатошарового перцептрону та отримання бажаного виходу. Основна ідея цього методу полягає в поширенні сигналів помилки від виходів мережі до її входів, в напрямку, зворотному прямому поширенню сигналів у звичайному режимі роботи. Для можливості застосування методу зворотного поширення помилки функція активації нейронів повинна бути диференційованою.

Навчання нейронних мереж можна представити як задачу оптимізації. Оцінити — означає вказати кількісно, добре чи погано мережа вирішує поставлені їй завдання. Для цього будується функція оцінки. Вона, як правило,

явно залежить від вихідних сигналів мережі і неявно — від всіх її параметрів. Найпростіший і найпоширеніший приклад оцінки — сума квадратів відстаней від вихідних сигналів мережі до їх необхідних значень:

$$H = \frac{1}{2} \sum_{\tau \in v_{out}} (Z(\tau) - Z^*(\tau))^2$$

де $Z^*(\tau)$ — необхідне значення вихідного сигналу.

Метод найменших квадратів далеко не завжди є найкращим вибором оцінки. Ретельне конструювання функції оцінки дозволяє на порядок підвищити ефективність навчання мережі, а також одержувати додаткову інформацію — «рівень впевненості» мережі у відповіді.

Алгоритм зворотного поширення помилки застосовується для багатошарового перцептрону. У мережі є множина входів $x_1,...,x_n$, множина виходів *Outputs* і безліч внутрішніх вузлів. Перенумеруємо всі вузли (включаючи входи і виходи) числами від 1 до N (наскрізна нумерація, незалежно від топології шарів). Позначимо через $\omega_{i,j}$ вагу зв'язку, що з'єднує i-й і j-й вузли, а через o_i — вихід i-го вузла. Якщо нам відомий навчальний приклад (правильні відповіді мережі t_k, $k \in Outputs$), то функція помилки, отримана за методом найменших квадратів, виглядає так:

$$E(\{w_{i,j}\}) = \frac{1}{2} \sum_{k \in Outputs} (t_k - o_k)^2.$$

а модифікація вагів відбувається за рахунок стохастичного градієнтного спуску наведеного в попередньому пункті.

1.2.3 Архітектури нейронних мереж та функції активації

Область нейронних мереж спочатку була натхненна переважно метою моделювання біологічних нейронних систем, але з тих пір розродилася і стала справою техніки та досягла хороших результатів у задачах машинного навчання.

Математична форма розрахунків моделі нейрона може виглядати знайомою. З лінійними класифікаторами, нейрон має здатність «подібності»

(активація близько одного) або «відмінності» (активація поблизу нуля) по відношенню до лінійних ділянок його вхідного простору. Таким чином, з відповідною функцією втрат на виході нейрона, ми можемо перетворити один нейрон в лінійний класифікатор:

Двійковий софтмакс класифікатор. Наприклад, ми можемо інтерпретувати $\sigma (\Sigma_i w_i x_i + Ь)$, щоб мати ймовірність одного з класів $P (Y_I = 1 | x_i; W)$. Імовірність іншого класу буде $P (y = 0 | x_i; W) = 1-P (y = 1 | X_I; W)$, так як вони повинні давати одиницю в сумі . При такій інтерпретації, ми можемо сформулювати втрати крос-ентропі, як ми вже бачили в розділі лінійної класифікації та оптимізації його привели б до двійкового софтмакс класифікатора (також відомий як логістична регресія). Так як область значень сигмовидної функції обмежена [0-1], передбачення даного класифікатора засновані на виходах нейрона більших, ніж 0,5.

Розповсюджені функції активації.

Сигмовидна нелінійна функція має математичну форму $\sigma (x) = 1 / (1 + e^{-x})$ і показана на Рис. 2 (а). Вона приймає на вхід дійсне число і стискає його в діапазон від 0 до 1. Зокрема, великі негативні числа стають 0 і великі позитивні числа стають 1. Сигмовидна функція часто використовувалась протягом довгого часу, так як вона має хорошу інтерпретацію як збудження нейрона: від спокійного стану (0) до повністю насиченого збудження при максимальному значення (1). Зараз сигмовидна нелінійна функція рідко використовується на практиці.

Тангенціальна функція стискає дійсне число і в діапазон від -1 до 1. (Рис. 2 б). На відміну від сигмовидної функції вихід тангенціальної дорівнює нуль в центрі. Тому на практиці TANH завжди краще сигмовидної нелінійності. Також відзначимо, що TANH функція просто масштабується сигмовидною, зокрема, справедливо наступне: $\tanh(x) = 2\sigma (2x) -1$.

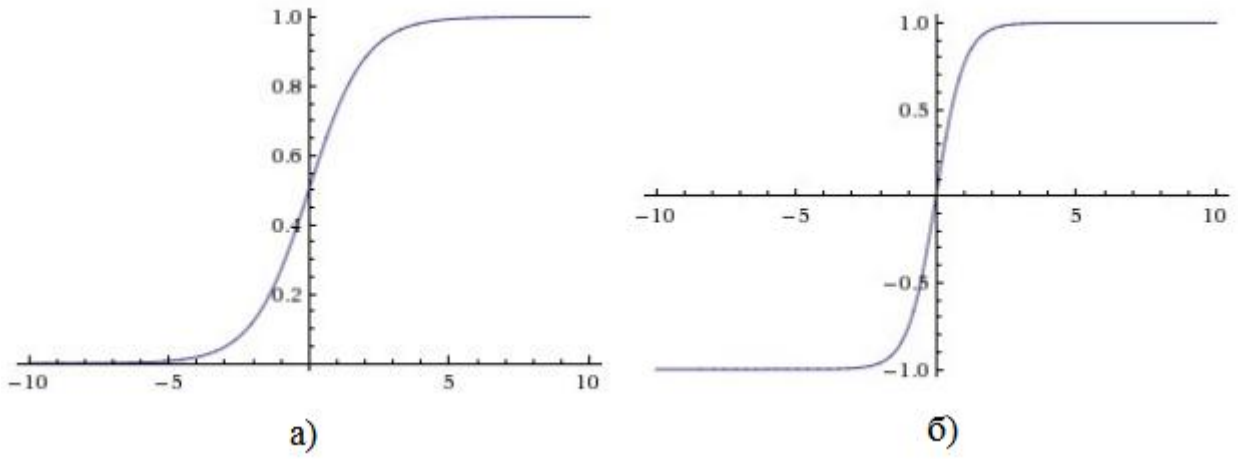


Рис. 3 – Сигмовидна нелінійна функція (а) та Тангенціальна функція (б)

ReLU стала дуже популярною в останні кілька років. Вона обчислює функцію F (X) = max (0, x). Іншими словами, активація відбувається по нульовому порогу (Рис. 4).

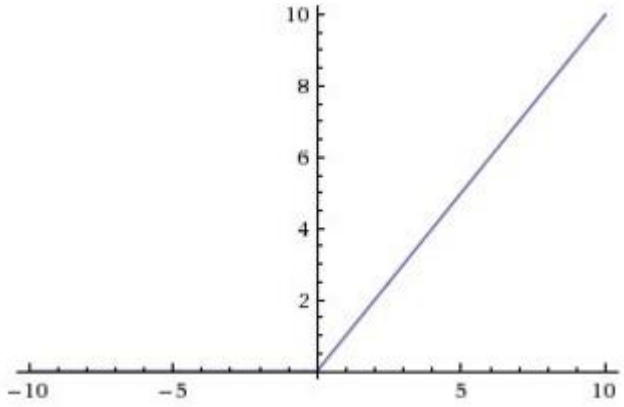


Рис. 4 – ReLU функція

Є кілька переваг та недоліків використання ReLUs:

- значне прискорення збіжності стохастичного градієнтного спуску в порівнянні з функціями TANH / сигмовидною.
- в порівнянні з функціями активації TANH / сигмовидною, які включають дорогі операції (експонент і т.д.), *ReLU* може бути реалізований за допомогою простої порогової матриці активацій в нулі.
- одиниці *ReLU* можуть бути крихкими під час тренування і може «померти». Наприклад, великий градієнт, що протікає через нейрон *ReLU* може привести ваги оновити таким чином, що нейрон ніколи не буде

активувати на будь-якій точці даних знову. Якщо це станеться, то градієнт, що протікає через пристрій завжди буде дорівнює нулю з цього моменту. Тобто, блоки *ReLU* можуть незворотньо померти під час тренування, так як вони можуть зіб'ють колектор даних. Наприклад, ви можете виявити, що майже 40% від вашої мережі може бути «мертвими» (тобто нейрони, які ніколи не активують по всьому навчальному набору даних), якщо швидкість навчання встановлена занадто високими. При правильному налаштуванні швидкості навчання це не є частою проблемою.

Вибір обсягу мережі

Правильний вибір обсягу мережі має велике значення. Побудувати невелику і якісну модель часто буває просто неможливо, а велика модель буде просто запам'ятовувати приклади з навчальної вибірки і не виробляти апроксимацію, що, природно, призведе до некоректної роботи класифікатора. Існують два основні підходи до побудови мережі - конструктивний і деструктивний. При першому з них спочатку береться мережу мінімального розміру, і поступово збільшують її до досягнення необхідної точності. При цьому на кожному кроці її заново навчають. Також існує так званий метод каскадної кореляції, при якому після закінчення епохи відбувається коригування архітектури мережі з метою мінімізації помилки. При деструктивному підході спочатку береться мережу завищеного обсягу, і потім з неї видаляються вузли та зв'язки, які мало впливають на рішення. При цьому корисно пам'ятати наступне правило: число прикладів в навчальній множині має бути більшим за кількість елементів вагової матриці. Інакше замість узагальнення мережу просто запам'ятає дані і втратить здатність до класифікації - результат буде невизначений для прикладів, які не ввійшли в навчальну вибірку.

Вибір архітектури мережі

При виборі архітектури мережі зазвичай випробовується кілька конфігурацій з різною кількістю елементів. При цьому основним показником є обсяг навчальної множини і узагальнююча здатність мережі. Зазвичай використовується алгоритм навчання Back Propagation з підтверджуючою множиною (validation set).

Fully-connected NN. Нейронні мережі моделюються як набори нейронів, з'єднаних у вигляді ациклічного графа[8]. Іншими словами, виходи деяких нейронів можуть стати входи інших нейронів. Цикли не допускається, так як це буде означати нескінченний цикл в прямому проході мережі. Замість аморфних згустків пов'язаних нейронів, модель нейронної мережі часто об'єднані в окремі шари нейронів. Для звичайних нейронних мереж, найбільш поширений тип шару є fully-connected шар, в якому нейрони між двома сусідніми шарами повністю з'єднані попарно, але нейрони в межах одного шару не мають зв’язків.

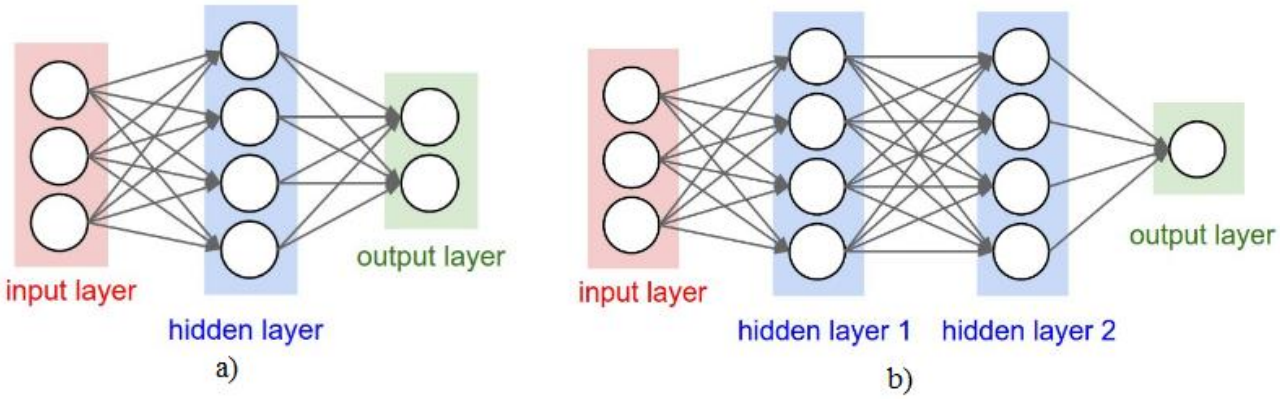


Рис. 5 – 2-шарова (a) та 3-шарова (b) fully-connected NN

Вище наведено два приклади топології нейронної мережі, які використовують стек повністю пов'язаних (fully-connected) шарів:

a) 2-шарова нейронна мережа (один прихований шар з 4-х нейронів і один вихідний шар з 2-х нейронів), і три входи;
b) 3-шарова нейронна мережа з трьома входами, двома прихованими шарами 4 нейронів кожен і один вихідний шар. Зверніть увагу на те, що в

обох випадках є зв’язки (синапси) між нейронами між шарами, але не в межах шару.

1.3 Порівняння фреймворків для роботи з НМ

Важливим аспектом у вирішенні задач машинного навчання є вибіл інструментів для зручної роботи з нейронними мережами. В таблиці 1. Представлений порівняльний аналіз фреймворків які спрощують роботі з машинним навчанням.

Табл. 1 – Порівняння фреймворків для роботи з машинним навчанням

Software	Open source	Platform	Interface	CUDA support	Has pretrained models	Recurrent nets	Convolutional nets	RBM/DBNs	Parallel execution (multi node)
Caffe	Yes	Linux, Mac OS X, Windows	Python, MATLAB	Yes	Yes	Yes	Yes	No	No
Microsoft Cognitive Toolkit	Yes	Windows, Linux (OSX via Docker)	Python, C++, Command line, BrainScript	Yes	Yes	Yes	Yes	No	Yes
TensorFlow	Yes	Linux, Mac OS X, Windows	Python, C/C++, Java, Go	Yes	Yes	Yes	Yes	Yes	Yes
Theano	Yes	Cross-platform	Python	Yes	Through Lasagne's models	Yes	Yes	Yes	Yes
Torch	Yes	Linux, Mac OS X, Windows,Android, iOS	Lua, LuaJIT, C, utility library for C++/OpenCL	Yes	Yes	Yes	Yes	Yes	Yes

1.3.1 Theano

Theano є одним з найбільших ветеранів МН і стабільних бібліотек. Початок бібліотек глибокого навчання є предметом суперечки між Theano і Caffe. Theano це низькорівнева бібліотека, яка слідує Tensorflow стилю. Вона використовується н стільки для глибокого навчання, як для чисельних розрахунків оптимізації, таких як автоматичне обчислення функції градієнта, який разом з інтерфейсом Python і його інтеграція з Numpy, зробив цю бібліотеку однією з найбільш часто використовуваних для загального призначення в сфері Deep Learning.

На сьогоднішній день, вона все ще існує і використовується, але той факт, що вона не підтримує мульти-GPU обчислень, дає підстави задуматись над вибором іншого фреймворку.

1.3.2 Caffe

Caffe на ряду з Theano є одним з найперших фреймворків. Він зосереджений тільки в комп'ютерному зорі, але виконує свої функції бездоганно. Експериментально визначено, що підготовка архітектури CaffeNet зайняла 5 разів менше, ніж в Caffe Keras (використовуючи бекенд Theano). Недоліки в тому, що Theano не є гнучким. Якщо необхідно ввести нові зміни, потрібно програмувати на C ++ і CUDA, існує можливість використовувати його інтерфейси Python або Matlab.

Недоліками є відсудність хорошої документації та складності в інсталяції через те що вона має багато залежностей. Проте як інструмент для продакшн система комп'ютерного зору Caffe є безумовним лідером. Оскільки тема даної роботи не пов’язана з комп’ютерним зором Caffeне підходить для вирішення поставлених задач.

1.3.3 Tensorflow

TensorFlow визначається як бібліотека програмного забезпечення з відкритим вихідним кодом для Machine Intelligence, але більш точне визначення: TensorFlow є відкритим вихідним кодом бібліотеки програмного забезпечення для чисельного розрахунку з використанням графіків потоку даних. Даний фреймворк не включає себе в рамки Deep Learning так само як і Теано.

Tensorflow підтримує Python і C ++, допускає розподілені CPU та GPU обчислення, а також горизонтальне масштабування з використанням gRPC.

У підсумку: Tensorflow є найкращим розподіленим фреймворком для роботи з машинним навчанням, а його функції побудовий графів даних допомагають відлагодити модель треновану в Tensorflow.

1.3.4 Torch

У Torch, існує кілька способів (стек шарів або шари графа), щоб визначити мережу, але по суті, мережа визначається як граф шарів. Через таку ступінь деталізації, Torch іноді вважається менш гнучким, оскільки для нових типів шарів, користувачі повинні реалізувати нову мережу, зворотні зв'язки, і градієнт поновлення введення.

Однак, на відміну від Caffe, визначаючи новий шар в Torch набагато простіше, тому що вам не потрібно програмувати на C ++. Крім того, в Torch різниця між новим визначенням рівня і визначенням мережі мінімальна. В Caffe шари визначені в C ++ в той час як мережі визначаються через Protobuf.

Torch є більш гнучким, ніж TensorFlow і Theano в тому, що вкрай важливо, в той час як TF / Theano є декларативним (тобто в них потрібно ініціалізувати обчислювальний граф). Це суттєво прощує деякі операції (на приклад, пошук променя) в Torch повівнюючи з іншими фреймворками.

1.3.5 Microsoft Cognitive Toolkit

CNTK система глибокого навчання більш відомий в співтоваристві NLP, ніж в загальному глибокому вивченні спільноти. У CNTK (як в TensorFlow і Теано), мережа визначається як символічний графік векторних операцій, таких як матриця додавання / множення або згортки. Шар просто склад цих операцій. Тонка зернистість будівельних блоків (операції) дозволяє користувачам винайти нові складні типи шарів без їх реалізації на мові низького рівня (як в Caffe).

Спосіб використання CNTK, схожий на Caffe, щоб вказати файл конфігурації і запустити командний рядок. CNTK має підтримку Python починаючи з версії 2.0 і C # підтримки в процесі. Подібно до Caffe, CNTK також є крос-платформним C ++ рішенням. Таким чином, розгортання повинно бути легким в більшості випадків. Проте, він не працює на ARM архітектурі, що обмежує його можливості на мобільних пристроях.

Висновок

При побудові системи аналізу сигналів в реальному часі та налаштуванні набору аналізуючих вейвлетних фільтрів слід враховувати компроміс між вимогами до необхідної точності обробки та опису характеристик сигналу та вимогами до системи обробки аудіо сигналів.

Стиснення за допомогою вейвлет перетвореть дає непогані результати на практиці. Однак метод мел-частотних кепстральних характеристик рекомендований як найкращий метод для розпізнавання аудіосигналів у роботі [9]. Таке перетворення дасть можливість використовувати існуючі музичні датасети для подальшої класифікації аудіо сигналів з використання методів машинного навчання.

На основі проведених досліджень виділені існуючі проблеми в процесі навчання та надані можливі рішення для їх усунення/мінімізації впливі на результат навчання. До таких проблем належать: вибір довжини кроків під час роботи стохастичного градієнтного спуску; боротьба із потраплянням в точку локального екстремуму; можливе перенавчання мережі.

Порівняльний аналіз фреймворків для роботи з машинним навчанням показав сильні та слабкі сторони найбільш популярних продуктів використовуваних в продакшн рішеннях. Враховуючи особливості завдання поставленого в роботі прийнято рішення використовувати TensorFlow оскільки він є досить гнучким інструментом. Він також володіє функцією побудови інтерактивних графів даних, які допомагають відлагодити модель треновану в Tensorflow та підібрати найбільш оптимальні параметри навчання.

2 АЛГОРИТМ КЛАСИФІКАЦІЇ МУЗИЧНИХ ДАНИХ ЗА ДОПОМОГОЮ НЕЙРОННИХ МЕРЕЖ

Цілі які ставляться перед алгоритмом.

Алгоритм обробки даних побудований таким чином що може аналізувати аудіо послідовність в режимі реального часу або в фоновому режимі обробляючи весь файл за один прохід. На початку іде обробка так званих «сирих даних», для приближення до реальних умов нейронна мережа навчалась на вибірці .wav файлів (Рис. 6).

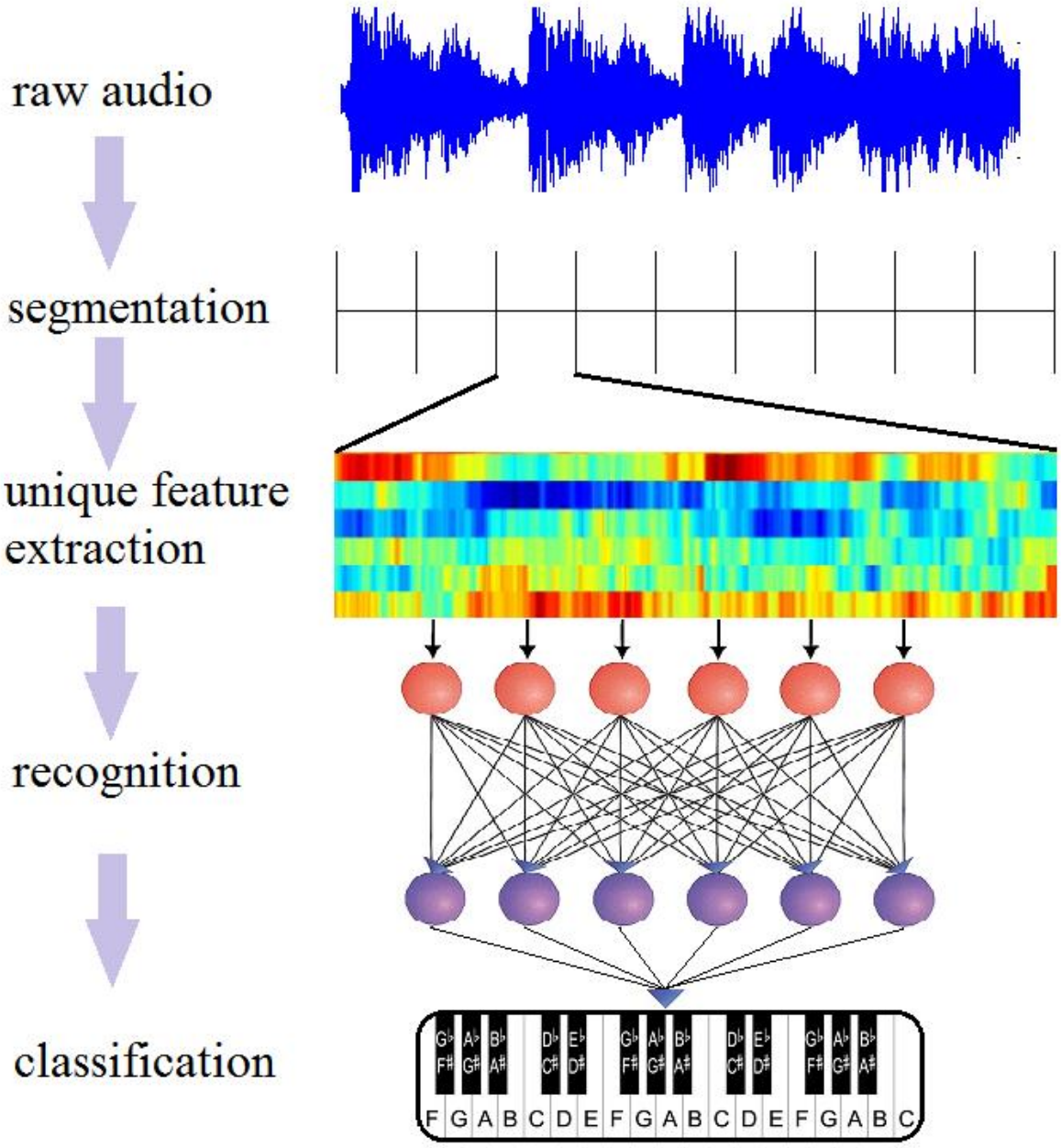


Рис. 6 – Схема проходу даних через алгоритм розпізнавання нот.

Далі алгоритм розбиває вхідну послідовність на інтервали довжиною в 0.2 сек. для подальшої побудови мел-частотних кепстральних характеристик (MFCC) отриманого інтервалу. Це дає можливість стиснути сигнал до декількох числових значень, зберігши при цьому швидкість обчислень та унікальність отриманих даних по відношенню до нот піаніно. Після обчислення MFCC коефіцієнти, які є «відбитком» аудіосигналу, подаються на вхід нейронної мережі. Вона організована в якості повнозв'язного багатошарового перецептрону. В якості класифікатора на виході мережі використовується софтмакс регресія. Фінальна стадія – перетворення виходу нейронної мережі (число в діапазоні від 0 до 127 яке характеризує висоту звуку в форматі MIDI) в ноту піаніно, зідно до предсталеної ничже таблиці. 2:

Табл. 2 – Відповідність ноти піаніно до висоти MIDI звуку.

Octave	Октава	Note Numbers											
		C	C#	D	D#	E	F	F#	G	G#	A	A#	B
-1	—	0	1	2	3	4	5	6	7	8	9	10	11
0	Субконтроктава	12	13	14	15	16	17	18	19	20	21	22	23
1	Контроктава	24	25	26	27	28	29	30	31	32	33	34	35
2	Большая	36	37	38	39	40	41	42	43	44	45	46	47
3	Малая	48	49	50	51	52	53	54	55	56	57	58	59
4	Первая	60	61	62	63	64	65	66	67	68	69	70	71
5	Вторая	72	73	74	75	76	77	78	79	80	81	82	83
6	Третья	84	85	86	87	88	89	90	91	92	93	94	95
7	Четвертая	96	97	98	99	100	101	102	103	104	105	106	107
8	Пятая	108	109	110	111	112	113	114	115	116	117	118	119
9	—	120	121	122	123	124	125	126	127				

2.1 Датасет для навчання музичного класифікатора

NSynth є аудіо набір даних, що містить 305,979 музичних нот, кожної з унікальним тембром і висотою звуку. Для інструментів з 1006 комерційних бібліотек зразків, згенеровано чотирисекундні, однотонні 16кГц аудіо фрагменти, або ноти, які ранжовані по висоті звуку стандартного MIDI піаніно (Табл. 2) (21-108), а також п'яти різними швидкостями (25, 50, 75, 100, 127). Нота звучить протягом перших трьох секунд і затухає протягом останньої секунди.

Деякі інструменти не здатні виробляти всі 88 градацій в цьому діапазоні, в результаті чого маємо в середньому 65,4 тонів звуку на інструмент. Крім того, комерційні пакети зразка іноді містять повторювані звуки на кілька швидкостей, в результаті чого в середньому 4,75 унікальних швидкостей на тон.

Також кожна нота анотована трьома додатковими лейблами інформації, заснованих на поєднанні людської оцінки та евристичних алгоритмів:

- джерело: спосіб отримання звуку для інструменту в нотах. Воно може бути акустичним або електронним для інструментів, які були записані на акустичних або електронні прилади, відповідно, або синтетичний для синтезованих інструментів.
- сім'я: сім'я високого рівня, членом якої є інструмент з даним звучанням. Кожен інструмент складається рівно з однієї сім'ї. Дивіться повний список і їх частоти нижче.
- якість: якість ноти від Sonic. Дивіться опис якості і їх супутні входження нижче. Кожна нота має анотацію нуль або більше якостей.

Повний набір даних розділяється на три групи:

- тренувальний набір з 289,205 прикладів для навчання. Інструменти не збігаються з дійсним або тестом.
- валідаційний набір для перевірки містить 12,678 прикладів. Прилади не перекриваються з тренувальною вибіркою.
- тестовий набір з 4,096 прикладами. Прилади не перекриваються з тренувальною вибіркою.

Кожен елемент будь якої вибірки містить такі дані зведені в таблицю 3:

Табл. 3 – Ознаки аудіо файлу в датасеті NSynth

Ознака	Тип	Опис
note	int64	Унікальний числовий ідентифікатор для ноти
note_str	bytes	Унікальний строковий ідентифікатор ноти формат: <instrument_str>-<pitch>-<velocity>.
instrument	int64	Унікальний послідовний ідентифікатор інструменту з якого отримана дана нота
instrument_str	bytes	Унікальний строковий ідентифікатор інструменту з якого отримана дана нота формат <instrument_family_str>-<instrument_production_str>-<instrument_name>.
pitch	int64	MIDI-базована висота ноти в діапазоні [0,127]
sample_rate	int64	Кількість семплів за секунду.
audio*	[float]	Набір аудіоданих в діапазоні [-1,1].
qualities	[int64]	Двійковий вектор який показує яка sonic особливість притамання даній ноті.
qualities_str	[bytes]	Строковий список який показує яка sonic особливість притамання даній ноті.
instrument_family	int64	Індекс сім'ї до якої належить інструмент.
instrument_family_str	bytes	Строкове представлення сім'ї до якої належить інструмент.
instrument_source	int64	Індекс sonic джерела для інструменту.
instrument_source_str	bytes	Строковий індекс sonic джерела для інструменту.

Датасет містить анотації звукового забарвлення для 10 різних якостей, описаних нижче. Жоден з тегів не є взаємовиключними за визначенням для «яскравого» і «темного», за винятком. Проте, це можливо примітка бути ні «яскравим», ні «темний».

Табл. 4 – Забарвлення ноти в датасеті NSynth

№	ID	Опис
0	bright	Велика кількість високий вміст частоти і сильних вищих гармоніки.
1	dark	Явна відсутність високочастотних, що дає приглушене і басовий звук. Крім того, іноді називають «Теплий».
2	distortion	Waveshaping, який виробляє характерний хрусткий звук і наявність багатьох гармонік.
3	fast_decay	Амплітудні огинають все гармоніки затухають по суті до «записної від" точки на 3 секунді.
4	long_release	Амплітуда обвідної загасає повільно після точки «зверніть увагу,-офф», іноді все ще присутній в кінці зразка 4 сек.
5	multiphonic	Наявність обертонів частот, що відносяться до більш ніж однієї основний частоті.
6	nonlinear_env	Модуляція звуку з відмінною поведінкою що огинає, відмінним від монотонного убування ноти. Може також включати конверти фільтра, а також динамічні конверти.
7	percussive	Гучний негармонійною звук в ноті початку.
8	reverb	Акустика приміщення, які не змогли бути видалені з вихідного зразка.
9	tempo-synced	Ритмічна модуляція звуку в фіксованому темпі.

Приклад опису ноти в форматі JSON:


```
"bass_synthetic_033-022-050": {
   "note": 201034,
   "sample_rate": 16000,
   "instrument_family": 0,
   "qualities": [0,1,0,0,0,0,0,0,0,0],
   "instrument_source_str": "synthetic",
   "note_str": "bass_synthetic_033-022-050",
   "instrument_family_str": "bass",
```

"instrument_str": "bass_synthetic_033",
"pitch": 22,
"instrument": 417,
"velocity": 50,
"instrument_source": 2,
"qualities_str": ["dark"]
}

2.2 Попередня обробка аудіосигналу

Музичний звук має характерну структуру в спектральному діапазоні. У його складі є основний тон, як правило, з максимальною амплітудою, і супутні гармоніки - обертони, найбільш значущими є перші кілька гармонік. Саме ці обертони визначають висоту звуку і відповідно музичну ноту. Однак спектр являє собою великий набір даних, які недоцільно використовувати в початковому вигляді для вирішення задачі розпізнавання. У зв'язку з цим необхідно визначити значущий набір ознак. В якості такого набору було вирішено використовувати мел-частотні кепстральні коефіцієнти (MFCC). Дані коефіцієнти були визначені як кращі ознаки для розпізнавання музичних інструментів в роботі [9].

Мел-частотні кепстральні коефіцієнти являють собою нелінійний спектр спектра, добре апроксимують слухову систему людини, а також успішно використовуються для вирішення завдань розпізнавання мови.

Алгоритм обчислення MFCC можна описати наступним чином [14]:

- обчислення віконного перетворення Фур'є;
- нелінійне розбиття спектра на n частин із застосуванням мел-шкали;
- обчислення енергії сигналу для кожного інтервалу із застосуванням трикутних фільтрів (з перекриттям);
- обчислення логарифма енергії сигналу для кожного інтервалу;
- виконання дискретного косинусного перетворення.

Для зниження складності отриманого простору ознак може бути використаний метод головних компонент (PCA). Це дозволило нам зменшити

кореляцію ознак і видалити найменш значущі з них. Алгоритм обчислення головних компонент може бути описаний таким чином:

- визначення матриці кореляції;

- знаходження власних значень і відповідних власних векторів;

- упорядкування власних векторів за відповідними їм собственнимзначеніям (по спадаючій);

- знаходження проекцій вхідних даних на власні вектори;

- відкидання останніх м проекції.

Перші проекції представляють найбільш значущі компоненти в вихідному векторі даних, і, відповідно, останні проекції представляють найменш значущі. Більш детально алгоритм описаний в [13, 18].

Мел – це психофізична одиниця висоти звуку, застосовується головним чином в музичній акустиці [10]. Кількісна оцінка звуку по висоті заснована на статистичній обробці великого числа даних про суб'єктивне сприйняття висоти звукових тонів. Результати досліджень показують, що висота звуку пов'язана головним чином з частотою коливань, але залежить також від рівня гучності звуку і його тембру. Перетворення відбувається по формулі:

$$M(f) = 1125 \ln(1 + f/700)$$

Відповідно графік перетворення мел шкали в частотну область виглядає як показано на Рис. 7

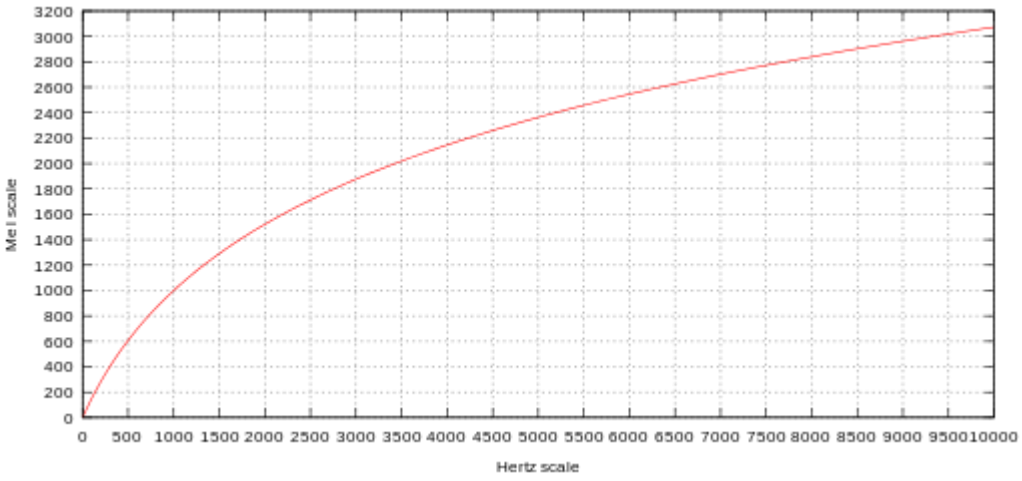


Рис. 7 – Перетворення мел шкали в частотну область

Звукові коливання частотою 1000 Гц при ефективному звуковому тиску $2*10^{-3}$ Па (тобто при рівні гучності 40 фон), що впливають спереду на спостерігача з нормальним слухом, викликають у нього сприйняття висоти звуку, що оцінюється за визначенням в 1000 мел. Звук частоти 20 Гц при рівні гучності 40 фон володіє за визначенням нульовою висотою (0 мел). Залежність нелінійна, особливо при низьких частотах (для «низьких» звуків).

2.3 Архітектурна модель нейронної мережі

Для вирішення поставленого завдання використовуємо архітектуру повнозвязного багатошарового персептрона прямого розповсюдження (fully-connected feed-forward neural network).

Параметри нейронної мережі для класифікації аудіо: вхідний шар містить 13 нейронів, далі два прихованих шари по 128 нейронів і вихідних шар для логістичної регресій також містить 128 нейронів. Детальний вигляд моделі представлений на Рис. 8.

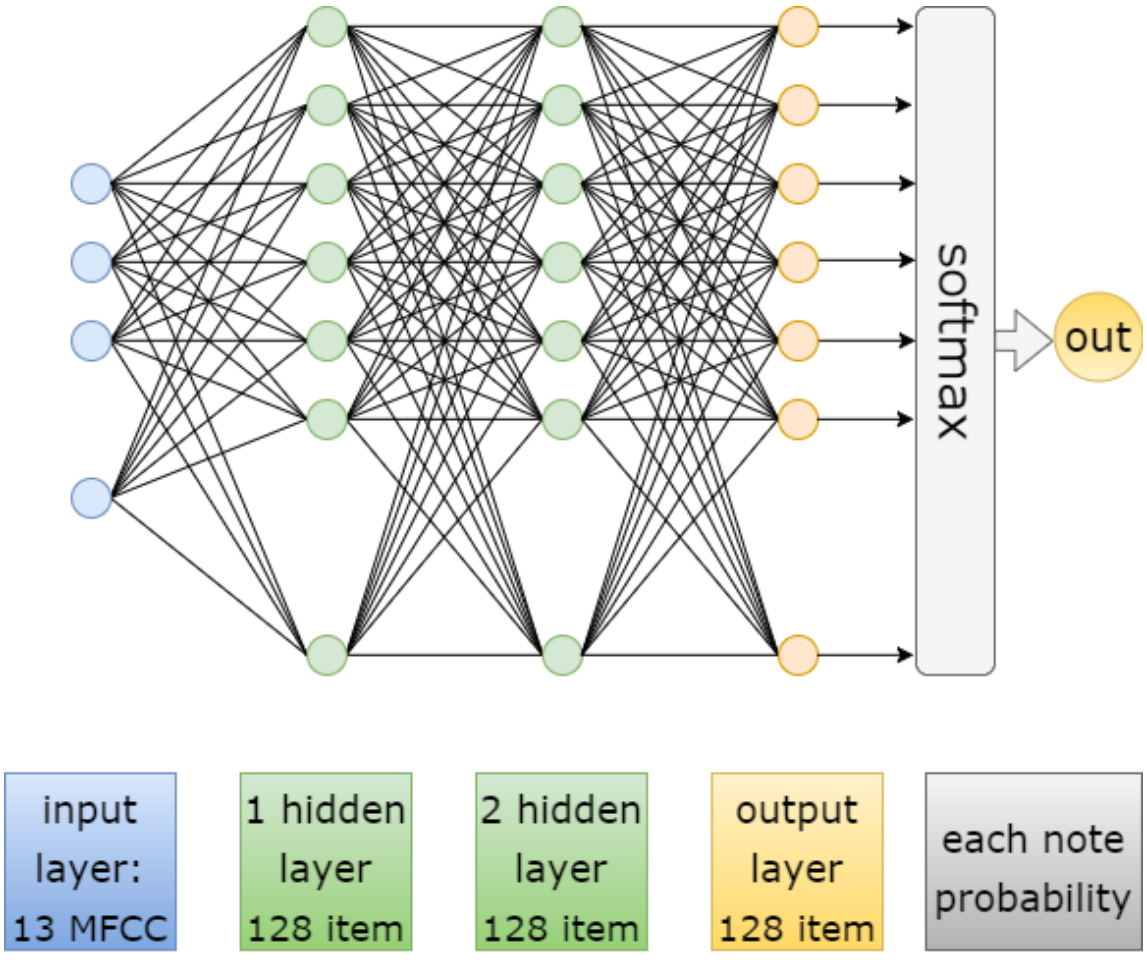


Рис. 8 – Архітектура багатошарового персептрона

Дана модель на виході розраховує ймовірність належності до кожного із 128 класів (по висоті звуку) за допомогою методу софтмакс регресії а також мінімізує похибки за допомогою методу градієнтного спуску та зворотного поширення помилки.

Висновок

Для навчання нейронної мережі обраний датасет Nsynth. Він містить інформацію про висоту звуку яка є унікальною для кожної ноти, а отже чудово підходить для вирішення поставленого завдання. Окрім піаніно він містить десятки інших інструментів тому перед навчанням датасет має бути відфільтрований таким чином щоб містити тільки необхідну інформацію, адже невалідність даних негативно впливає на процес навчання.

Спектр аудіосигналу являє собою великий набір даних, які недоцільно використовувати в початковому вигляді для вирішення задачі розпізнавання. У зв'язку з цим необхідно визначити значущий набір ознак. В якості такого набору було вирішено використовувати мел-частотні кепстральні коефіцієнти (MFCC). Дані коефіцієнти були визначені як кращі ознаки для розпізнавання музичних інструментів в роботі [9]. Широке застосування саме цих ознак спричинено тим, що вони апроксимують систему слуху людського вуха завдяки зглажуванню різкого сприйняття зміни у нижньому діапазоні частот та навпаки підсиленні сприйняття до змін високочастотних сигналів.

Параметри нейронної мережі для класифікації аудіо: вхідний шар містить 13 нейронів, далі два прихованих шари по 128 нейронів і вихідних шар для логістичної регресій також містить 128 нейронів. Модель на виході розраховує ймовірність належності до кожного із 128 класів (по висоті звуку) за допомогою методу софтмакс регресії а також мінімізує похибки за допомогою методу градієнтного спуску та зворотного поширення помилки.

3 РЕАЛІЗАЦІЯ АЛГОРИТМУ ТА ОГЛЯД ОТРИМАНИХ РЕЗУЛЬТАТІВ

Алгоритм обробки даних побудований таким чином що може аналізувати аудіо послідовність в режимі реального часу або в фоновому режимі обробляючи весь файл за один прохід. На початку іде обробка так званих «сирих даних», для приближення до реальних умов нейронна мережа навчалась на вибірці .wav файлів.

Далі алгоритм розбиває вхідну послідовність на інтервали довжиною в 0.2 сек. для подальшої побудови мел-частотних кепстральних характеристик (MFCC) отриманого інтервалу. Це дає можливість стиснути сигнал до декількох числових значень, зберігши при цьому швидкість обчислень та унікальність отриманих даних по відношенню до нот піаніно. Після обчислення MFCC коефіцієнти, які є «відбитком» аудіосигналу, подаються на вхід нейронної мережі. Вона організована в якості повнозв'язного багатошарового перецептрону. В якості класифікатора на виході мережі використовується софтмакс регресія. Фінальна стадія – перетворення виходу нейронної мережі (число в діапазоні від 0 до 127 яке характеризує висоту звуку в форматі MIDI) в ноту піаніно, зідно до таблиці 2:

3.1 Використання MFCC для створення «відбитку» звуку.

Датасет NSynth [15] містить аудіо дані в форматі розробленому спеціально для TensorFlow а також існує варіант завантаження даних де аудіо дані зберігається в .wav файлах а розмічені лейбли в окремому JSON файлі. Для пришвидшення роботи алгоритму прийнято рішення перебудувати базу даних таким чином щоб вона повністю зберігалась в JSON форматі. Це досягається записом додаткового поля «MFCC» як характеристику кожного семплу із тренувальної, валідаційної та тестової вибірок. Таким чином

TensorFlow не потрібно буде під час навчання витрачати час на тисячі процесів відкриття аудіо файлів та підрахунку частотних характеристик кожного файлу. Для вирішення поставленої задачі цей процес достатньо провести одноразово.

Розглянемо роботу алгоритму більш детально:

```
import json
import os
from python_speech_features import mfcc #бібліотека для обчислень MFCC
import scipy.io.wavfile as wav #бібліотека для роботи з .wav файлами

WORK_DIR = 'nsynth-train/' #шлях до директорії з датасетом

def getMFCC(wavfile, n):
  print("Analyzing " + str(n) + " wav: " + wavfile)
  (rate,sig) = wav.read(wavfile)
  mfcc_feat = mfcc(sig,rate) # обчислення mfcc характеристики вхідного .wav
  return mfcc_feat[1,:] # python_speech_features дозволяє обчислювати енергію
                        # сигналу, дельта коефіцієнти та нормавані дельта.
                        # В якості ознак для порівняння використовуються дельта.

def parseJsonFile():
  infile = WORK_DIR + 'examples.json' # вхідний файл для парсингу
  outfile = WORK_DIR + 'pianoMFCC.json' # результати парсингу
  with open(infile, 'r') as f:
    json_data = json.load(f)
    new_data = {}
    counter = 0
    for obj in json_data:
      if 'keyboard_acoustic' in obj:  # reduce unused objects;
        wavpath = WORK_DIR + "audio/" + obj + ".wav"
        counter += 1
```

```
        serialized_mfcc = list(getMFCC(wavpath, counter)) # серіалізація відповіді
        json_data[obj]["MFCC"]=serialized_mfcc # створення нового поля в  json
        tmp = {obj:json_data[obj]}
        new_data.update(tmp)

with open(outfile, 'w') as outf:
    json.dump(new_data, outf, indent=4) # оновлення вихідного файлу на основі
                                        # обчислених характеристик
```

Алгоритм обробки відсіює всі невикористані семпли із вибірки і паралельно замінює необхадність використання аудіо за допомогою додання нового поля в датасет з лейблами. В результаті отримали наступне розподілення даних для піаніно по всьому датасету:

Табл. 5 – Розподілення даних для піаніно

nsynth-test	119 items
nsynth-valid	321 items
nsynth-train	7351 items

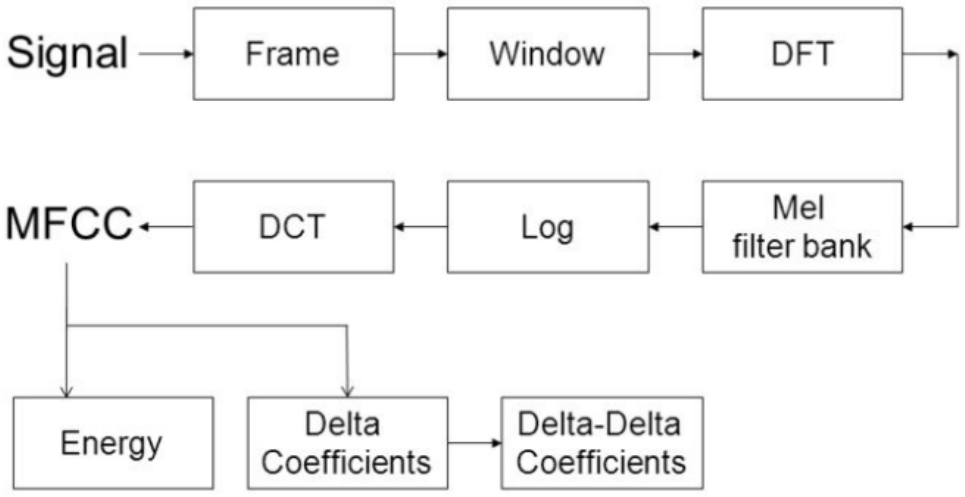


Рис. 9 – Алгоритм обчислення MFCC

Власне обчислення MFCC відбувається за допомогою бібліотеки python_speech_features. На Рис. 9 представлений алгоритм обчислення з якого видно що на виході отримуємо дані у трьох представленнях:

- енергія сигналу
- дельта коефіцієнти
- дельта-дельта коефіцієнти (нормовані дельта)

В якості характеристик аудіо в роботі використовувались дельта коефіцієнти.

Приклад вектора MFCC ознак для ноти *H(B):*

```
"MFCC": [
  18.456741311964741,
  22.559780411944082,
  -4.1885086703425065,
  -11.326731612118941,
  -4.2346220069489524,
  -0.93987034378969936,
  -14.567367380116758,
  5.4945554409177859,
  11.360135675607172,
  6.8177498936522625,
  -7.3702806360145825,
  -17.103638846308524,
  2.1466219884029929 ]
```

Побудова MFCC ознак базується на наступних параметрах:

```
python def mfcc (  signal,samplerate=16000,winlen=0.025,winstep=0.01,numcep=13,
                   nfilt=26,nfft=512,lowfreq=0,highfreq=None,preemph=0.97,
                   ceplifter=22,appendEnergy=True)
```

Основним тут є параметр numcep=13 - кількість кепстрів які ми отримуємо на виході як "відбитків" аудіосигналу.

3.2 Побудова моделі та навчання нейронної мережі за допомогою фреймворку TensorFlow

Як було зазначено у пункті 4 розділу 1 найбільш сприятливим фреймворком для виконання даної роботи вибрано TensorFlow. Він наділений всіма необхідними засобами для машинного навчання, має безліч оптимізацій, що пришвидшують процес навчання нейронних мереж. Завдяки легким моделям (мається на увазі пам'ять необхідна для збереження матриць вагів) TensorFlow створює production-ready рішення, можуть бути повторно використані і готові до інтеграції в додатки будь-якого класу.

Параметри нейронної мережі для класифікації аудіо: вхідний шар містить 13 нейронів, далі два прихованих шари по 128 нейронів і вихідних шар для логістичної регресій також містить 128 нейронів. Зв’язки нейронної мережі налаштовані по принципу fully-connected.

Після обробки датасет необхідно завантажити в TensorFlow. Оскільки дані зберігаються в форматі JSON, а не в готовому для TensorFlow вигляді то завантаження даних вимагає додаткової уваги (Додаток А).

Далі розглянемо деатльніше воркфлов роботи з засобами TensorFlow.

У placeholder_inputs () функція створює два tf.placeholder, які визначають форму входів, в тому числі batch_size – розмір чергової порції данних.

```
sounds_placeholder = tf.placeholder (tf.float32, shape = (batch_size,
                                                nsynth.CEPS_NUM))
labels_placeholder = tf.placeholder (tf.int32, shape = (batch_size))
```

Далі вниз, в тренувальному циклі, дані з датасету подаються за розміром batch_size для кожного кроку, а потім передаються в функцію sess.run () за допомогою параметра feed_dict.

Після створення наповнювачів для даних, будується граф даних який відповідає 3 етапам:

- inference() – за необхідності будує граф для запуску мережі вперед, щоб робити прогнози.
- loss() - додає графу дані, необхідниі для побудови функції втрат.
- training() - додає до графа дані, необхідні для застосовувати методу градієнтного спуску.

Кожен шар мережі створюється під унікальним tf.name_scope, який діє в якості префікса для елементів, створених в межах цього обсягу задач. У межах певного обсягу задач, ваги і зміщення, які будуть використовуватися в кожному шарі генеруються в tf.Variable, з їх бажаною формою. Дві основні функції графа: вибір функції активації tf.nn.relu та метода взаємодії між шарами tf.matmul Потім створюється шар за шаром створюється нейронна мережа:

```
hidden1 = tf.nn.relu(tf.matmul(images, weights) + biases)
hidden2 = tf.nn.relu(tf.matmul(hidden1, weights) + biases)
logits = tf.matmul(hidden2, weights) + biases
```

При побудові функції втрат в програмі додається метод tf.nn.sparse_softmax_cross_entropy_with_logits для порівняння даних вихідного шару нейронної мережі з розміченими даними з датасету. Далі використовується tf.reduce_mean для усереднення значень крос-ентропії в якості повної втрати по всьому набору чергової порції даних.

```
cross_entropy = tf.nn.sparse_softmax_cross_entropy_with_logits(
                    labels=labels, logits=logits, name='xentropy')
loss = tf.reduce_mean(cross_entropy, name='xentropy_mean')
```

Функція training() модифікує ваги мережі таким чином щоб мінімізувати похибку із функції loss() за допомогою методу градієнтного спуску.

```
tf.summary.scalar('loss', loss)
optimizer = tf.train.GradientDescentOptimizer(learning_rate)
global_step = tf.Variable(0, name='global_step', trainable=False)
train_op = optimizer.minimize(loss, global_step=global_step)
```

В результаті отриримуємо модель повнозв’язного багатошарового персептрона з двома прихованими і одним вихідним шарами по 128 нейронів кожен. Дана модель на виході розраховує ймовірність належності до кожного із 128 класів (по висоті звуку) за допомогою методу софтмакс регресії а також мінімізує похибки за допомогою методу градієнтного спуску.

3.3 Обробка даних на виході нейронної мережі

Завершальним етапом розробки алгоритму є побудова транслятора виходу нейронної мережі в зрозумілу ноту. Мережа на виході активує той нейрон який відповідає розпізнаній висоті звуку.

Табл.6 – Відповідність ноти піаніно до частоти та висоти MIDI звуку

Нота	Суб-контр-октава		Контр-октава		Большая октава		Малая октава		Первая октава	
До	12	16.35	24	32.7	36	65.41	48	130.82	60	261.63
До#	13	17.32	25	34.65	37	69.3	49	138.59	61	277.18
Ре	14	18.35	26	36.95	38	73.91	50	147.83	62	293.66
Ре#	15	19.44	27	38.88	39	77.78	51	155.56	63	311.13
Ми	16	20.61	28	41.21	40	82.41	52	164.81	64	329.63
Фа	17	21.82	29	43.65	41	87.31	53	174.62	65	349.23
Фа#	18	23.12	30	46.25	42	92.5	54	185	66	369.99
Соль	19	24.5	31	49	43	98	55	196	67	392
Соль#	20	25.95	32	51.9	44	103.8	56	207	68	415.3
Ля	21	27.5	33	55	45	110	57	220	69	440
Ля#	22	29.13	34	58.26	46	116.54	58	233.08	70	466.16
Си	23	30.87	35	61.74	47	123.48	59	246.96	71	493.88
Нота	Вторая октава		Третяя октава		Четвертая октава		Пятая октава			
До	72	523.25	84	1046.5	96	2093	108	4186		
До#	73	554.36	85	1108.7	97	2217.4	109	4434.8		
Ре	74	587.32	86	1174.6	98	2349.2	110	4698.4		
Ре#	75	622.26	87	1244.5	99	2489	111	4978		
Ми	76	659.26	88	1318.5	100	2637	112	5274		
Фа	77	698.46	89	1396.9	101	2793.8				
Фа#	78	739.98	90	1480	102	2960				
Соль	79	784	91	1568	103	3136				
Соль#	80	830.6	92	1661.2	104	3332.4				
Ля	81	880	93	1720	105	3440				
Ля#	82	932.32	94	1864.6	106	3729.2				
Си	83	987.75	95	1975.5	107	3951				

Тобто один із 128 виходів мережі не буде нульовим. Для коректної інтерпретації цих даних використана бібліотека Python MIDI. Вона дозволяє конвертувати MIDI базовану висоту звуку в частоту або ім’я ноти яка відповідає натисненій клавіші піаніно в момент розпізнавання у відповідності до таблиці 6.

3.4 Огляд результатів роботи алгоритму

Навчання нейронної мережі складається із 2000 епох в кожну з яких на вхід мережі іде по 100 випадкових наборів даних. Кожну соту епоху відбувається оцінка функції втрат для моніторингу похибки та процесу навчання в цілому. Кожну тисячну епоху відбувається оцінка точності нейронної мережі яка є відношенням кількості коректних спрацьовувань до загальної кількості спроб.

Рис.10 – Зміна похибки в процесі навчання нейронної мережі

Результати оцінки можна спостерігати під час навчання нейронної мережі у вигляді логу навчання (Додаток В) та на графіку навчання Рис. 10. Як бачимо під час навчання похибка поступово зменшується, але важливим залишається не перенавчити нейронну мережу так щоб вона давала хороші результати тільки на тестових даних. Зростання хибних спрацьовувань після 1700 кроку свідчить про вироблення якості узагальнення у нейронної мережі. Цей факт підтверджується Training Data Precision: 0.8961 < Validation Data Precision: 0.9070 тим, що точність коректних спрацьовувать на валідаційній вибірці датасету перевищує таку ж величину на тренувальному датасеті.

Висновок

Мел-частотні кепстральні коефіцієнти являють собою нелінійний спектр спектра, добре апроксимують слухову систему людини, а також успішно використовуються для вирішення завдань розпізнавання мови та жанрів музики. Як показує практика ці коефіцієнти доцільно використовувати і в області розпізнавання нот, а також вони дозволяють розширити задачу до класифікації окремих акордів які являють собою набір звуків подібних до нот. Дослідження класифікації акордів є підгрунтям для подальшої роботи над даною темою, а використання мел-частотних кепстральних коефіцієнтів спрощує задачу попередньої обробки аудіосигналу на вході такої системи.

В результаті побудована модель повнозв’язного багатошарового персептрона з двома прихованими і одним вихідним шарами по 128 нейронів кожен. Дана модель на виході розраховує ймовірність належності до кожного із 128 класів (по висоті звуку) за допомогою методу софтмакс регресії а також мінімізує похибки за допомогою методу градієнтного спуску.

Процес навчання моделі виконаний успішно адже вона показала близько 90% точності на будь-якому наборі даних. Слід зазначити, що необхідною умовою для коректного спрацьовування даної моделі є правильно налаштований музичний інструмент. В той же час наведений алгоритм алгоритм з мінімальними модифікаціями можна використовувати для настройки музичного інструменту.

У випадку натискання декількох клавіш піаніно алгоритм видасть найближчу по висоту ноту, до зіграного акорду. Ця проблема є підгрунтям для подальших досліджень в напрямку розпізнавання звуків клавішних музичних інструментів.

4 РОЗРОБКА СТАРТАП-ПРОЕКТУ

Описана технологія може буди реалізована в якості веб-додатку таким чином ним можна буде користуватись в будь-який зручний спосіб при наявності підключення до мережі Internet. Цей розділ ставить перед собою мету реалізації технології розпізнавання звучання нот піаніно. Процес включає в себе:

- імплементацію веб-сервісу з використанням технології розпізнавання з даної роботи;
- розробка стратегії виходу конкурентоспроможного продукту на ринок та подальший розвиток стартапу.

4.1 Інформаційна карта проекту

1. Назва проекту	Piano easy

2. Автори проекту	Галатенко Дмитро Володимирович

3. Анотація	Створення веб-додатку для навчання грі на піаніно без початкової бази знань. Навчання передбачається шляхом побудови часової діаграми натискання клавіш. Все, що необхідно додатку це лише дати “послухати” композицію і він відразу ж надасть можливість швидкого та ефективного навчання новим композиціям. Це надасть можливість користувачам не просто навчитись грати нові пісня а з блискавичною швидкістю збільшити репертуар мелодій навіть вишуканими новинками сучасних геніїв гри на піаніно.

4. Термін реалізації проекту	6 місяців
	Тривалість проекту (в місяцях)

5. Необхідні ресурси	3 людини, 3 персональних комп’ютери, 705000 грн.
	Перелік усіх необхідних ресурсів (фінансових, матеріальних інтелектуальних та ін.)
6. Опис проблеми, яку вирішує проект	Об’єкт допомагає любителям та спеціалістам гри на піаніно вивчати нові композиції без додаткових зусиль. Програма сама побудує з них часову діаграму мелодії.
7. Головні цілі та завдання проекту	Розробити веб додаток з подальшою портацією під мобільні пристрої для побудови часових діаграм натискання клавіш по готовій композиції.

8. Очікувані результати
Людям які бажають навчитись новинкам від майстрів гри а піаніно більше не треба буде чекати доки вийде навчальне відео з часовими діаграмами з детальним розбором композиції, програма сама буде в режимі реального часу будувати такі діаграми.

4.2 Команда стартап-проекту

Керівник проекту	Галатенко Дмитро Володимирович, 2 роки досвіду створення комерційних програмних продуктів, більше року досвіду в управлінні IT-проектами
Член команди 1	Суржан Антон Олегович, інженер-програміст з 4 роками досвіду
Член команди 2	Федченко Євген Антонович, фінансист, маркетолог з 5 роками досвіду

4.3 Маркетингова стратегія та маркетинговий план стартапу

Табл. 7 – Опис ідеї стартап-проекту

Зміст ідеї	*Напрямки застосування*	*Вигоди для користувача*
Створення веб-додатку здатного синтезувати часову діаграму натискання клавіш піаніно за допомогою записаної композиції. Користувач на вхід дає мелодію пісні у вигляді аудіо файлу. Веб-сервіс самостійно створює часову діаграму натискання клавіш піаніно, що спрощує вивчення нових пісень.	1. Для любителів шанс навчитись без додаткових зусиль грати ті композиції які їм подобаються незважаючи ні на що.	Користувач має вказати лише вхідні дані та те, що він хоче отримати, а система все зробить сама. Це пришвидшить роботу наукових установ та компаній, у яких дослідження пов’язані з цифровою обробкою.
	2. Для професіоналів це скоротить час який витрачається на обробку існуючої композиції – програма робитиме це автоматично.	Користувачу не треба встановлювати додаткове програмне забезпечення на комп’ютер, потрібен лише веб-браузер.

Табл. 8 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	*Техніко-економічні характеристики ідеї*	*(потенційні) товари/концепції конкурентів*				*W (слабка сторона)*	*N (нейтральна сторона)*	*S (сильна сторона)*
		Мій проект	*Конкурент1*	*Конкурент2*	*Конку-рент3*			
1.	Форма виконання	Веб-сервіс+ моб	Веб-сервіс	Програма	Програма			+
2.	Собівартість	Низька	Низька	Низька	Висока		+	
3.	Наявність адміністратора для налаштування	Не треба, автоматично	Треба	Треба	Треба			+
4.	Наявність інтернету	Необхідно для веб додатку	Необхідно	Не треба	Не треба		+	
5.	Крос-платформенність	Так	Так	Ні	Ні			+

Визначений перелік слабких, сильних та нейтральних характеристик та властивостей ідеї потенційного товару є підґрунтям для формування його конкурентоспроможності.

Табл. 9 – Технологічна здійсненність ідеї проекту

n/n	*Ідея проекту*	*Технології її реалізації*	*Наявність технологій*	*Доступність технологій*
1.	Створення додатку	Flash	Наявна	Платна, недоступна
		Java-applets	Наявна	Безкоштовна, доступна
		ASP .NET	Наявна	Безкоштовна, доступна
Обрана технологія реалізації ідеї проекту: для створення веб-додатку обрана технологія ASP .NET, яка є безкоштовною та з якою мають досвід роботи члени проекту.				

Табл. 10 – Попередня характеристика потенційного ринку стартап-проекту

№п/п	*Показники стану ринку (найменування)*	*Характеристика*
1	Кількість головних гравців, од	3
2	Загальний обсяг продаж, грн/ум.од	20000 грн./ум.од
3	Динаміка ринку (якісна оцінка)	Зростає/спадає/стагнує
4	Наявність обмежень для входу (вказати характер обмежень)	Немає
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі (або по ринку), %	R = (3000000 * 100) / (1000000 * 12) = 25%

Табл. 11 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	*Потреба, що формує ринок*	*Цільова аудиторія (цільові сегменти ринку)*	*Відмінності у поведінці різних потенційних цільових груп клієнтів*	*Вимоги споживачів до товару*
1.	Необхідно програмне забезпечення для навчання гри на піаніно	Потенційними цільовими групами є будь яка людина з бажанням навчитись грати нові композиції на піаніно	Цільова група не має вікових обмежень – повинная мати бажання розвивати музикальні таланти на піаніно	Рішення має бути крос-платформним, необхідно реалізувати зружний графічний вивід на екран з можливістю стоп/пауза перемотка.

Табл. 12 – Фактори загроз

№ п/п	*Фактор*	*Зміст загрози*	*Можлива реакція компанії*
1.	Конкуренція	Вихід на ринок великої компанії	Вихід з ринку. Запропонувати великій компанії поглинути себе. Передбачити додаткові переваги власного ПЗ для того, щоб повідомити про них саме після виходу міжнародної компанії на ринок.
2.	Зміна потреб користувачів	Користувачам необхідне програмне забезпечення з іншим функціоналом	Передбачити можливість додавання нового функціоналу до створюваного ПЗ

Табл. 13 – Фактори можливостей

№ п/п	*Фактор*	*Зміст можливості*	*Можлива реакція компанії*
1.	Зростання можливостей потенційних покупців	Духовний розвиток населення у звязку з підняттям мінімалки до 3200грн	Продумати періоди безкоштовного користування додатком.
2.	Зниження довіри до конкурента 1	У ПЗ конкурента №1 нещодавно була знайдена помилка, завдяки якій дані досліджень усіх клієнтів стали доступні в інтернеті для всіх користувачів	При виході на ринок звертати увагу покупців на унікальність нашого ПЗ

Табл. 14 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	*В чому проявляється дана характеристика*	*Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)*
1. Тип конкуренції: - досконала	Існує 3 фірми-конкуренти на ринку	Врахувати ціни конкурентних компаній на початкових етапах створення бізнесу, реклама.

2. За рівнем конкурентної боротьби: - міжнародний	Одна з компаній – з ішної країни, дві – з України	Додати можливість вибору мови ПЗ, щоб легше було у майбутньому вийти на міжнародний ринок
3. За галузевою ознакою: - внутрішньогалузева	Конкуренти мають ПЗ, яке використовується лише всередині даної галузі	Створити основу ПЗ таким чином, щоб можна було легко переробити дане ПЗ для використання у інших галузях
4. Конкуренція за видами товарів: - товарно-видова	Види товарів є однаковими, а саме – програмне забезпечення	Створити ПЗ, враховуючи недоліки конкурентів
5. За характером конкурентних переваг: - неціновa	Вдосконалення технології створення ПЗ, щоб собівартість була нижчою	Використання менш дорогих технологій для розробки, ніж використовують конкуренти
6. За інтенсивністю: - не марочна	Бренди відсутні	-

Табл. 15 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	*Прямі конкуренти в галузі*	*Потенційні конкуренти*	*Постачальники*	*Клієнти*	*Товари-замінники*
	Навести перелік прямих конкурентів	*Визначити бар'єри входження в ринок*	*Визначити фактори сили постачальників*	*Визначити фактори сили споживачів*	*Фактори загроз з боку замінників*
Висновки	Існує 3 конкуренти на ринку. Найбільш схожим за виконанням є конкурент 1, так як його рішення також представлене у вигляді веб-додатку.	Так, можливості для входу на ринок є, бо наше рішення спрощує та пришвидшує роботу музиканта.	Постачальники відсутні.	Важливим для користувача є кросплатформеність ПЗ та швидкість його роботи.	Товари-замінники можуть використати більш дешеву технологію створення ПЗ та зменшити собівартість товару.

Табл. 16 – Обґрунтування факторів конкурентоспроможності

№ п/п	*Фактор конкурентоспроможності*	*Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)*
1.	Виконання ПЗ у вигляді веб-сервісу	Це рішення дозволяє використовувати ПЗ з будь-якого пристрою, яке підключене до інтернету та має веб-браузер
2.	Простота інтерфейсу користувача	Інтерфейс користувача зроблений таким чином, що користувачу необхідно лише задати вхідні. Проміжні ланки роботи відсутні, що пришвидшує отримання результату.

Табл. 17 – Порівняльний аналіз сильних та слабких сторін проекту “Piano Easy”

№ п/п	*Фактор конкурентоспроможності*	*Бали 1-20*	*Рейтинг товарів-конкурентів у порівнянні з нашим підприємством*						
			-3	*-2*	*-1*	*0*	*+1*	*+2*	*+3*
1.	Виконання ПЗ у вигляді веб-сервісу	15			+				
2.	Простота інтерфейсу користувача	20	+						

Табл. 18 – SWOT аналіз стартап-проекту

Сильні сторони: простий інтерфейс користувача, виконання ПЗ у вигляді веб-сервісу	Слабкі сторони: необхідний доступ до інтернету для роботи з ПЗ
Можливості: у конкурента 1 виявлена проблема із безпекою ПЗ, додаткове держфінансування для досліджень у підприємствах, які є потенційними покупцями	Загрози: конкуренція, зміна потреб користувачів

Табл. 19 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	*Альтернатива (орієнтовний комплекс заходів) ринкової поведінки*	*Ймовірність отримання ресурсів*	*Строки реалізації*
1.	Створення веб-сервісу за технологією ASP.NET, з використанням онтології в основі, яка дозволятиме спростити інтерфейс користувача	90%	6 місяців
2.	Створення веб-сервісу за допомогою технології ASP.NET, яка не має в основі онтологію, що є більш швидким рішенням	60%	4 місяці

Проаналізвавши наведені вище дані робимо висновок, що альтернатива 1 є найбільш сприятливою для ринкового впровадження старпап-проекту.

Табл. 20 – Вибір цільових груп потенційних споживачів

№ п/п	*Опис профілю цільової групи потенційних клієнтів*	*Готовність споживачів сприйняти продукт*	*Орієнтовний попит в межах цільової групи (сегменту)*	*Інтенсивність конкуренції в сегменті*	*Простота входу у сегмент*
1.	Домашнє використання (професіонали)	Можливе виникнення деяких зауважень з боку ідеології навчання гри на піаніно.	Цікаво спробувати новинку на ринку музикальної освіти.	Існує 3 конкуренти, які надають схожі, але менш швидкі рішення. До того ж – лише 1 конкурент надає крос-платформне рішення	У сегмент увійти непросто, бо фахівці гри на піаніно можуть мати зауваження щобо такої суті навчання
2.	Домашнє використання (любителі)	Простий графічний інтерфейс та зручне управління дозволять зацікавити потенційного клієнта з			Маючи перевагу у зручності інтерфесу користувача, що пришвидшує роботу та отримання

		першого застосування.			результату, вийти на ринок нескладно
3.	Освітні музичні заклади.	Пришвидшення часу виконання додає їм можливості виконати більшу кількість замовлень, до того ж в цьому році з держбюджету виділені додаткові кошти на цей напрям досліджень			Маючи перевагу у тому, що рішення є зручним для користування і крос-платформним, вийти на ринок не є складно
Які цільові групи обрано: обираємо домашнії користувачів, та працівників освітніх музичних закладів.					

За результатами аналізу потенційних груп споживачів (сегментів) автори ідеї обирають цільові групи, для яких вони пропонуватимуть свій товар, та визначають стратегію охоплення ринку

Табл. 21 – Визначення базової стратегії розвитку

№ п/п	*Обрана альтернатива розвитку проекту*	*Стратегія охоплення ринку*	*Ключові конкурентоспроможні позиції відповідно до обраної альтернативи*	*Базова стратегія розвитку**
1.	Створення веб-сервісу за технологією ASP.NET, з використанням онтології в основі, яка дозволятиме спростити інтерфейс користувача	Ринкове позіціювання	Простота інтерфейсу, пришвидшення роботи, крос-платформність	Лідерство по витратам

Табл. 22 – Визначення базової стратегії конкурентної поведінки

№ п/п	*Чи є проект «першопрохідцем» на ринку?*	*Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?*	*Чи буде компанія копіювати основні характеристики товару конкурента, і які?*	*Стратегія конкурентної поведінки**
1.	Так	Шукати нових споживачів	Буде, а саме: основною задачею ПЗ є обробка цифрових сигналів (конкуренти 1, 2, 3), крос-платформність (як у конкурента 1)	Зайняття конкурентної ніші

Табл. 23 – Визначення стратегії позиціонування

№ п/п	*Вимоги до товару цільової аудиторії*	*Базова стратегія розвитку*	*Ключові конкурентоспроможні позиції власного стартап-проекту*	*Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)*
1.	Простота інтерфейсу, крос-платформність	Лідерство по витратам	Простота користувацького інтерфейсу, що дозволяє пришвидшити та спростити роботу працівників, крос-платформність, яка вимагає лише наявність веб-браузера	Швидкість, крос-платформність, простота

Табл. 24 – Визначення ключових переваг концепції потенційного товару

№ п/п	*Потреба*	*Вигода, яку пропонує товар*	*Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити*
1.	Крос-платформність	Використання будь-якої операційної системи	Рішення є крос-платформним
2.	Спрощення інтерфейсу користувача	Пришвидшення роботи з ПЗ	Користувачам не потрібно обробляти нову композицію вручну. Вони мають лише надати аудіофайл із композицію для аналізу та побудови діаграми.

4.4 Техніко-економічні характеристики товару

Табл. 25 – Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>І. Товар за задумом</td><td colspan="3">Об’єкт допомогає любителям та спеціалістам гри на піаніно вивчати нові композиції без додаткових зусиль. Програма сама побудує з них часову діаграму мелодії.</td></tr>
<tr><td rowspan="5">ІІ. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх /Тл/Е/Ор</td></tr>
<tr><td>1. Крос-платформне
2. Наявність-веб браузеру необхідне
3. Наявність інтернету необхідне
4. Простота у використанні</td><td>-</td><td>-</td></tr>
<tr><td colspan="3">Якість: згідно до стандарту ISO 4444 буде проведено тестування</td></tr>
<tr><td colspan="3">Маркування відсутнє.</td></tr>
<tr><td colspan="3">Моя компанія. «Piano easy»</td></tr>
<tr><td rowspan="2">ІІІ. Товар із підкріпленням</td><td colspan="3">1-місячна пробна безкоштовна версія</td></tr>
<tr><td colspan="3">Постійна підтримка для користувачів</td></tr>
<tr><td colspan="4">За рахунок чого потенційний товар буде захищено від копіювання: ноу-хау.</td></tr>
</table>

Табл. 26 – Визначення меж встановлення ціни

№ п/п	*Рівень цін на товари-замінники*	*Рівень цін на товари-аналоги*	*Рівень доходів цільової групи споживачів*	*Верхня та нижня межі встановлення ціни на товар/послугу*
1.	25000	30000	200000	20000

Табл. 27 – Формування системи збуту

№ п/п	*Специфіка закупівельної поведінки цільових клієнтів*	*Функції збуту, які має виконувати постачальник товару*	*Глибина каналу збуту*	*Оптимальна система збуту*
1.	Купують ПЗ та роблять щорічні внески для подовження ліцензії	Продаж	0(напряму), 1(через одного посередника)	Власна та через посередників

Табл. 28 – Концепція маркетингових комунікацій

№ п/п	*Специфіка поведінки цільових клієнтів*	*Канали комунікацій, якими користуються цільові клієнти*	*Ключові позиції, обрані для позиціонування*	*Завдання рекламного повідомлення*	*Концепція рекламного звернення*
1.	Купівля ПЗ через інтернет, робота з ПЗ на комп’ютерах з різними ОС	Інтернет, жива мова.	Швидкодія, простота використання, крос-платформність	Показати переваги ПЗ, у тому числі і перед конкурентами	Демо-ролик із використання, реклама

4.5 Елементи фінансової моделі стартапу

Табл. 29 – Сукупні інвестиційні витрати на реалізацію стартап-проекту

№ з/п	*Стаття витрат*	*Сукупні витрати, тис. грн.*
	Загальні початкові витрати	405
1.1.	Проведення пошукових та прикладних досліджень	5
1.2.	Розробка проектних матеріалів і ТЕО	10
1.3.	Робоче проектування і прив'язка проекту	10
1.4.	Витрати на придбання обладнання та устаткування та пристроїв	100
1.5.	Витрати на придбання нематеріальних активів	40
1.6.	Витрати на утримання обладнання та приміщень	40
1.7.	Витрати на передвиробничі маркетингові дослідження	50
1.8.	Витрати на створення збутової мережі	50
1.9.	Витрати на просування та рекламу	50
1.10	Оплата юридичних послуг	50
2.	***Витрати на матеріальні ресурси***	0
2.1.	матеріали	0
2.3.	комплектуючі	0
3.3.	сировина	0
3.	***Витрати на оплату праці команди стартапу***	300
Разом:		705

Табл. 30 – Визначення основних фінансово-економічних показників проекту

№ з/п	*Стаття витрат*	*Сукупні витрати, тис. грн.*
1.	Обсяг виробництва продукції в натуральних показниках	*1*
2.	Собівартість одиниці продукції, тис. грн.	*705000*
3.	Собівартість виробництва продукції, тис. грн. *(3 = 1 · 2)*	*705000*
4.	Обсяг реалізації продукції в натуральних показниках	*100*
5.	Ціна реалізації продукції без ПДВ, тис. грн.	*20000*
6.	Виручка від реалізації продукції без ПДВ, тис. грн. *(6 = 4 · 5)*	*2000000*
7.	Податок на додану вартість (ПДВ), тис. грн.	*200000*
8.	Валовий прибуток *(8 = 6 – 3)*	*1295000*
9.	Податок на прибуток	*259000*
10.	Чистий прибуток *(10 = 8 – 9)*	*1036000*

Рентабельність продажів (або маржа прибутку) показує, скільки прибутку приносить кожна гривня з обсягу реалізації. Маржу прибутку, як правило, визначають окремо за кожним видом діяльності або за кожною групою реалізованої продукції

Rs = 51.8%

Період окупності проекту відображає час, який потрібен для того, щоб сума надходжень від реалізації проекту відшкодувала суму витрат на його впровадження. Період окупності звичайно вимірюється в роках або місяцях

PBP = 0.35 (роки)

Рентабельність довгострокових інвестицій – коефіцієнт повернення інвестицій, показник рентабельності вкладень, що у відсотковому співвідношенні демонструє прибутковість (у разі, коли його значення більше 100%) або збитковість (у разі, коли його значення менше 100%) інвестицій в проект. Якщо розрахований показник менший 100%, то інвестиційні вкладення не окупуються.

ROI = (1036000 – 705000) / 705000 * 100% = 46.9%

4.6 Програма запобігання та реагування на ризики проекту

Табл. 31 – Програма запобігання та реагування на ризики проекту

Ризики проекту	***Заходи запобігання ризиків***	***Заходи реагування при виникненні ризиків***
Вихід з ладу БД	Збереження копій даних на інших серверах	Отримання копії даних з інших серверів
Звільнення члена команди	Детальна декомпозиція завдань, щоб зробити кожне з них максимально простим та незалежним. Використання систем контролю версій.	Продовження роботи без цієї людини
Збільшення собівартості через зміну ліцензії ASP.NET	Немає	Необхідне додаткове фінансування для продовження ліцензії
Зміна системного API ASP.NET	Приймати участь в обговореннях переваг та недоліків цієї технології на офіційному сайті компанії Microsoft, абстрагувати деякі рівні ПЗ для незалежності від системного API	Змінити код у тих місцях, де використовується системне API

Висновок

У даному розділі були досліджені основны аспекти виходу на ринок додатку розпізнавання нот ріаніно «Piano Easy». Описаний продукт є доцільний для користувачів, які бажають навчитись новинкам від майстрів гри а піаніно більше не треба буде чекати доки вийде навчальне відео з часовими діаграмами з детальним розбором композиції, програма сама буде в режимі реального часу будувати такі діаграми.

В рамках розділу було визначено перелік слабких, сильних та нейтральних характеристик та властивостей ідеї потенційного товару є підґрунтям для формування його конкурентоспроможності; обрана технологія реалізації ідеї проекту: для створення веб-додатку обрана технологія ASP .NET, яка є безкоштовною та з якою мають досвід роботи члени проекту; проведений ступеневий аналіз конкуренції на ринку, SWOT аналіз та обґрунтовані фактори конкурентоспроможності.

Отже відповідно до проведених досліджень:

- існує можливість ринкової комерціалізації проекту;
- існують перспективи впровадження з огляду на потенційні групи клієнтів, бар’єри входження не є високими, проект має дві значні переваги перед конкурентами: кросплатформність, а також простота та швидкість у використанні ПЗ;
- необхідно реалізувати веб-додаток із використанням технології ASP.NET;
- подальша імплементація є доцільною.

ВИСНОВКИ

В результаті проведеного дослідження в можна зробити наступні висновки.

На основі проведених досліджень виділені існуючі проблеми в процесі навчання та надані можливі рішення для їх усунення/мінімізації впливів на результат навчання. До таких проблем належать: вибір довжини кроків під час роботи стохастичного градієнтного спуску; боротьба із потраплянням в точку локального екстремуму; перенавчання мережі.

Спектр аудіосигналу являє собою великий набір даних, які недоцільно використовувати в початковому вигляді для вирішення задачі розпізнавання. У зв'язку з цим необхідно визначити значущий набір ознак. В якості такого набору було вирішено використовувати мел-частотні кепстральні коефіцієнти (MFCC). Дані коефіцієнти були визначені як кращі ознаки для розпізнавання музичних інструментів а також в вирішення задач NLP. Широке застосування саме цих ознак спричинено тим, що вони апроксимують систему слуху людського вуха завдяки зглажуванню різкого сприйняття зміни у нижньому діапазоні частот та навпаки підсиленні сприйняття до змін високочастотних сигналів.

Проведений порівняльний аналіз фреймворків для роботи з машинним навчанням, який показав показав сильні та слабкі сторони найбільш популярних продуктів використовуваних в сучасних рішеннях. Враховуючи особливості завдання поставленого в роботі прийнято рішення використовуати TensorFlow оскільки він є досить гнучким інструментом, володіє функцією побудови інтерактивних графів даних, які допомагають відлагодити модель треновану в Tensorflow та підібрати найбільш оптимальні параметри навчання.

Обраний датасет Nsynth, який містить інформацію про висоту звуку яка є унікальною для кожної ноти, а отже чудово підходить для вирішення поставленого завдання. Окрім піаніно він містить десятки інших інструментів тому перед навчанням датасет має бути відфільтрований таким чином щоб

містити тільки необхідну інформацію, адже невалідність даних негативно впливає на процес навчання.

Параметри нейронної мережі для класифікації аудіо: вхідний шар містить 13 нейронів, далі два прихованих шари по 128 нейронів і вихідних шар для логістичної регресій також містить 128 нейронів. Модель на виході розраховує ймовірність належності до кожного із 128 класів (по висоті звуку) за допомогою методу софтмакс регресії а також мінімізує похибки за допомогою методу градієнтного спуску та зворотного поширення помилки.

Реалізовано систему розпізнавання нот піаніно, в процесі навчання модель показала близько 90% точності на будь-якому наборі даних.

Слід зазначити, що необхідною умовою для коректного спрацьовування даної моделі є правильно налаштований музичний інструмент. В той же час наведений алгоритм алгоритм з мінімальними модифікаціями можна використовувати для настройки музичного інструменту.

У випадку натискання декількох клавіш піаніно алгоритм видасть найближчу по висоті ноту, до зіграного акорду. Ця проблема є підгрунтям для подальших досліджень в напрямку розпізнавання звуків клавішних музичних інструментів.

В рамках розробки стартап-проекту було визначено перелік слабких, сильних та нейтральних характеристик та властивостей ідеї потенційного товару, що є підґрунтям для формування його конкурентоспроможності; обрана технологія реалізації ідеї проекту: для створення веб-додатку обрана технологія ASP .NET, яка є безкоштовною та з якою мають досвід роботи члени проекту; проведений ступеневий аналіз конкуренції на ринку, SWOT аналіз та обґрунтувані фактори конкурентоспроможності.

Описаний продукт з використанням представленої технології є доцільним для користувачів, які бажають навчитись новинкам від майстрів гри піаніно; більше не треба буде чекати доки вийде навчальне відео з часовими діаграмами з детальним розбором композиції, програма зможе будувати такі діаграми без участі людини.

Таким чином в роботі розкрита актуальна науково-прикладна проблема: створення підходу до розпізнавання звуків піаніно за допомогою систем штучного інтелекту для побудови часово-нотної діаграми натискання клавіш піаніно без участі людини.

Відповідно до поставленої мети в роботі:

- розроблено алгоритм перетворення аудіо сигналу який дозволить мінімізувати час обробки аудіосигналу;
- спроектовано модель на основі нейронної мережі для класифікації звучання нот піаніно;
- розроблено програмний додаток розпізнавання аудіо контенту;
- розроблено стратегію стартап проекту, який дозволить реалізувати описану технологію в якості конкурентоспроможного продукту.

Проблема натискання декількох клавіш піаніно одночасно являє собою іншу – більш глибоку задачу розпізнавння акордів. Таким чином вона може слугувати в якості вектора подальшого дослідження для захисту кандидатської дисертації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ле, Нгуен Виен, and Д. П. Панченко. "Распознавание речи на основе искусственных нейронных сетей." Международная научная конференция «Технические науки в России и за рубежом». \ Издательство «Молодой ученый», Ваш полиграфический партнер, 2011.
2. Сергиенко А.Б. Цифровая обработка сигналов [Текст] / А.Б. Сергиенко. – 2 е издание. – СПб.: Питер, 2006. – 752 с.
3. Системний аналіз та інформаційні технології: матеріали 19-ї Міжнародної науково-технічної конференції SAIT 2017, Київ, 22 – 25 травня 2017 р. / ННК “ІПСА” НТУУ “КПІ ім. Ігоря Сікорського”. – К.: ННК “ІПСА” НТУУ “КПІ ім. Ігоря Сікорського”, 2017. – 151 с.
4. Смоленцев Н. Основы теории вейвлетов. Вейвлеты в Matlab [Текст] / Н. Смоленцев. – М.: ДМК Пресс 2014. – 628 с.
5. Юр, Тетяна Василівна, Татьяна Васильевна Юр, and Tetiana V. Yur. "Метод визначення частотних характеристик сигналів за допомогою вейвлетів." (2015).
6. Baelde, Maxime, and Christophe Biernacki. "A mixture model-based real-time audio sources classification method." The 42nd IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP2017. 2017.
7. Eronen A. Comparison of features for musical instrument recognition. Workshop on Signal Processing for Audio and Acoustics, 2001. – P. 19–22.
8. Eronen, Antti, and Anssi Klapuri. "Musical instrument recognition using cepstral coefficients and temporal features." Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on. Vol. 2. IEEE, 2000.

9. Hinton, Geoffrey, et al. "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups." IEEE Signal Processing Magazine 29.6 (2012): 82-97.
10. Imai, Satoshi. "Cepstral analysis synthesis on the mel frequency scale." Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'83.. Vol. 8. IEEE, 1983.
11. Najmi A. The Continuous Wavelet Transform and Variable Resolution Time – Frequency Analysis [Text] / A. Najmi, J. Sadowsky // JOHNS HOPKINS APL TECHNICALDIGEST. – 1997. – Vol. 18, No. 1. – P. 134-140.
12. Scardapane, Simone, and Aurelio Uncini. "Semi-supervised echo state networks for audio classification." Cognitive Computation 9.1 (2017): 125-135.
13. Simon O. Haykin Neural networks: a comprehensive foundation, 2d ed, Person Education, 1999, 842 p.
14. Stanford U. CS231n: Convolutional Neural Networks for Visual Recognition [Електронний ресурс] / Univercity Stanford – Режим доступу до ресурсу: http://cs231n.github.io/.
15. The NSynth Dataset [Електронний ресурс] // Google Inc. – Режим доступу до ресурсу: https://magenta.tensorflow.org/datasets/nsynth.
16. Tsomokos, Dimitris I., Sahel Ashhab, and Franco Nori. "Fully connected network of superconducting qubits in a cavity." New Journal of Physics 10.11 (2008): 113020.
17. Wrzołek, Tomasz, and Zbigniew Remin. "Palaeontological modeling–classical recognition of genera and species in rugose corals vs. self-organizing Kohonen networks classification." 9th Paleontological Conference. Warszawa, Polska, Warszawa, Polska. 2008.
18. X. Huang, A. Acero, and H. Hon. Spoken Language Processing: A guide to theory, algorithm, and system development. Prentice Hall, 2001.

Додаток А Код обробки даних для завантаження JSON об’єктів

```
import json
import tensorflow as tf
import numpy as np
train_dataset = 'nsynth-train/pianoMFCC.json'

def get_mfcc(f):
        json_data = json.load(f)
        return np.array([obj["MCFF"] for obj in json_data], dtype='f')

def extract_sounds(f):
        mfcc = tf.placeholder(tf.float32, [None])
        [data] = tf.py_func(get_mfcc, [mfcc], [tf.float32])
        return data

json_path = train_dir + train_dataset
with open(json_path, 'r') as f:
        train_sounds = extract_sounds(f)
        train_labels = extract_pitches(f, one_hot=one_hot)
```

Додаток В Оцінка нейронної мережі під час навчання

Step 0: loss = 2.31 (0.387 sec)
Step 100: loss = 2.19 (0.013 sec)
Step 200: loss = 2.06 (0.013 sec)
Step 300: loss = 1.75 (0.013 sec)
Step 400: loss = 1.33 (0.012 sec)
Step 500: loss = 1.11 (0.013 sec)
Step 600: loss = 0.84 (0.013 sec)
Step 700: loss = 0.72 (0.013 sec)
Step 800: loss = 0.54 (0.013 sec)
Step 900: loss = 0.55 (0.013 sec)
Training Data Eval:
  Num examples: 55000 Num correct: 47697 Precision @ 1: 0.8672
Validation Data Eval:
  Num examples: 5000 Num correct: 4367 Precision @ 1: 0.8734
Test Data Eval:
  Num examples: 10000 Num correct: 8721 Precision @ 1: 0.8721
Step 1000: loss = 0.51 (0.015 sec)
Step 1100: loss = 0.39 (0.101 sec)
Step 1200: loss = 0.45 (0.013 sec)
Step 1300: loss = 0.50 (0.013 sec)
Step 1400: loss = 0.39 (0.013 sec)
Step 1500: loss = 0.46 (0.013 sec)
Step 1600: loss = 0.38 (0.012 sec)
Step 1700: loss = 0.46 (0.013 sec)
Step 1800: loss = 0.44 (0.013 sec)
Step 1900: loss = 0.47 (0.014 sec)
Training Data Eval:
  Num examples: 55000 Num correct: 49288 Precision @ 1: 0.8961
Validation Data Eval:
  Num examples: 5000 Num correct: 4525 Precision @ 1: 0.9070
Test Data Eval:
  Num examples: 10000 Num correct: 8998 Precision @ 1: 0.8998