

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки
Кафедра технічної кібернетики

«До захисту допущено»

Завідувач кафедри

_____ Ігор ПАРХОМЕЙ

«___» _____ 2020 р.

Дипломний проєкт

на здобуття ступеня бакалавра

**за освітньо-професійною програмою «Програмне забезпечення
інтелектуальних та робототехнічних систем»**

спеціальність 121 «Інженерія програмного забезпечення»

**на тему: «Інтелектуальна система підбору ударної партії для аранжування
композицій»**

Виконала:

студентка IV курсу, групи ІТ-62

Каширіна Ольга Юріївна _____

Керівник:

доцент, к.т.н.

Цьопа Н. В. _____

Консультант з норм. контролю:

доцент, к.т.н.

Пасько В. П. _____

Рецензент:

професор КБЗІ КНУ ім. Т. Шевченка

професор, д.т.н.

Толюпа С. В. _____

Засвідчую, що у цьому дипломному
проєкті немає запозичень з праць інших
авторів без відповідних посилань.

Студентка _____

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра технічної кібернетики

Рівень вищої освіти – перший (бакалаврський)

Спеціальність 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма «Програмне забезпечення інтелектуальних та робототехнічних систем»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Ігор ПАРХОМЕЙ

«__» _____ 2020 р.

ЗАВДАННЯ
на дипломний проєкт студентки
Каширіної Ольги Юріївни

1. Тема проєкту «Інтелектуальна система підбору ударної партії для аранжування композицій», керівник проєкту Цьопа Наталія Володимирівна, к.т.н, доцент, затверджені наказом по університету від «07» травня 2020р. № 1081-с

2. Термін подання студентом проєкту: 27.05.2020 р.

3. Вихідні дані до проєкту: Системи штучного інтелекту для роботи з музикою, набір числових даних для навчання нейронної мережі.

4. Зміст пояснювальної записки:

Вступ

1. Огляд предметної області

2. Аналіз проблем при реалізації системи підбору ударної партії для аранжування композицій

3. Розробка архітектури системи підбору ударної партії для аранжування композицій

4. Аналіз результатів роботи системи підбору ударної партії для аранжування композицій

Висновки

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо): Блок-схема алгоритмів конвертації файлів між числовим та MIDI-форматом (А3); Загальна схема роботи нейронних мереж (А3); Результати оцінки якості роботи програми (А3).

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Норм. контроль	доцент Пасько В. П.		
Перевірка на співпадіння	доцент Лісовиченко О. І.		

7. Дата видачі завдання «01» жовтня 2019р.

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1	Аналіз існуючих рішень серед систем штучного інтелекту для роботи з музикою.	01.10.19 — 30.10.19	
2	Постановка задачі та формулювання функціональних вимог до розроблюваної інтелектуальної системи.	01.11.19 — 29.12.19	
3	Проектування, створення та навчання нейронної мережі для підбору ударних партій.	30.12.19 — 23.02.20	
4	Проектування та розробка системи підбору ударних партій.	24.02.20 — 12.04.20	
5	Тестування якості розробленої системи.	13.04.20 — 17.05.20	
6	Оформлення дипломної роботи.	18.05.20 — 26.05.20	
7	Попередній захист.	27.05.20	
8	Норм. контроль.	28.05.20	
9	Перевірка на співпадіння.	05.06.20	
10	Захист.	15.06.20	

Студентка

Ольга КАШИРІНА

Керівник

Наталія ЦЬОПА

АНОТАЦІЯ

У роботі розглянуто системи штучного інтелекту для роботи з музикою в цілому та системи і методи для роботи з ритмами у музиці зокрема, показано основні особливості існуючих рішень інтелектуальних систем для підбору ударних партій, їх переваги та недоліки.

Розроблено інтелектуальну систему підбору ударної партії для аранжування композицій, що підбирає барабанні партії до заданих мелодійних, керуючись принципами контрапункту та співвідношенням акцентів у мелодійних та ударних партіях. Дана система може бути використана композиторами-початківцями, що не мають досвіду аранжування композицій, не працювали з барабанними установками та не вивчали принципів контрапункту та аранжування. Крім того система може стати у нагоді музичним гуртам, що займаються аранжуванням власних пісень.

Ключові слова: нейронні мережі, штучний інтелект, музика, аранжування, генерування ударної партії, Java, R.

Розмір пояснювальної записки — 55 аркушів, містить 14 ілюстрацій, 6 таблиць, 6 додатків.

ABSTRACT

The research paper examines artificial intelligence systems working with music in general, as well as systems and methods working with rhythms in music in particular, shows the main features of existing solutions for intelligent beat generation systems, their advantages and disadvantages.

An intelligent beat generation system for song arrangement has been developed, which generates beat to the given melody, guided by the principles of counterpoint and the correlation between accents in melody and beat. This system can be used by novice composers who have no experience in song arrangement, have not worked with drum installations and have not studied the principles of counterpoint and arrangement. In addition, the system can be useful for music bands that arrange their own songs.

Keywords: neural networks, artificial intelligence, music, song arrangement, beat generation, Java, R.

Explanatory note size — 55 pages, contains 14 illustrations, 6 tables, 6 appendices.

**Пояснювальна записка
до дипломного проєкту
на тему: «Інтелектуальна система підбору ударної
партії для аранжування композицій»**

Київ – 2020 року

ЗМІСТ

ПЕРЕЛІК ВАЖЛИВИХ СКОРОЧЕНЬ І ТЕРМІНІВ	8
ВСТУП	9
РОЗДІЛ 1. ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ	
1.1. Загальний огляд систем штучного інтелекту для роботи з музикою	11
1.1.1. Історія	11
1.1.2. Огляд алгоритмів для написання музики штучним інтелектом	12
1.2. Ритми у комп'ютерній музиці	14
1.3. Огляд існуючих рішень проблеми підбору ритмічної партії	15
1.3.1. Алгоритм Евкліда для генерації ритмів	15
1.3.2. Генерація ритмів за допомогою рекурентних нейронних мереж	16
1.4. Постановка задачі	17
ВИСНОВКИ ДО РОЗДІЛУ	18
РОЗДІЛ 2. АНАЛІЗ ПРОБЛЕМ ПРИ РЕАЛІЗАЦІЇ СИСТЕМИ ПІДБОРУ УДАРНОЇ ПАРТІЇ ДЛЯ АРАНЖУВАННЯ КОМПОЗИЦІЙ	
2.1. Складності роботи з контрапунктом	20
2.2. Розміри барабанної установки	21
2.3. Обґрунтування вибору формату MIDI для роботи з композиціями	23
2.4. Обґрунтування вибору нейронної мережі для аналізу існуючих композицій	25
2.5. Проблема аналізу вихідних даних	27
2.6. Деталізація постановки задачі	29
ВИСНОВКИ ДО РОЗДІЛУ	30
РОЗДІЛ 3. РОЗРОБКА АРХІТЕКТУРИ СИСТЕМИ ПІДБОРУ УДАРНОЇ ПАРТІЇ ДЛЯ АРАНЖУВАННЯ КОМПОЗИЦІЙ	
3.1. Кодування вихідних даних	31
3.2. Проєктування нейронної мережі	34
3.3. Конвертація даних між числовим форматом та MIDI	37
3.3.1. Опис пакета javax.sound.midi	37
3.3.2. Конвертація MIDI-файлу в числовий формат	39
3.3.3. Конвертація числових результатів в MIDI-формат	40
3.4. Загальна структура проєкту	41
ВИСНОВКИ ДО РОЗДІЛУ	44
РОЗДІЛ 4. РОЗДІЛ 4. АНАЛІЗ РЕЗУЛЬТАТІВ РОБОТИ СИСТЕМИ ПІДБОРУ УДАРНОЇ ПАРТІЇ ДЛЯ АРАНЖУВАННЯ КОМПОЗИЦІЙ	
4.1. Дослідження якості роботи нейронної мережі	45
4.2. Результати опитування слухачів	49
ВИСНОВКИ ДО РОЗДІЛУ	51
ВИСНОВКИ	53
ПЕРЕЛІК ПОСИЛАНЬ	54
ДОДАТКИ	56

					ІТ-62.08.1081.01 ПЗ						
Змн.	Арк.	№ докум.	Підпис	Дат.							
Розроб.		Каширіна О.Ю.			Інтелектуальна система підбору ударної партії для аранжування композицій			Літ.	Арк.	Аркушів	
Перевір.		Цьопа Н.В.								7	55
Н. Контр.		Пасько В.П.									
Затверд.		Пархомей І.Р.			Пояснювальна записка			КПІ ім. Ігоря Сікорського Каф. ТК Гр. ІТ-62			

ПЕРЕЛІК ВАЖЛИВИХ СКОРОЧЕНЬ І ТЕРМІНІВ

MIDI (Musical Instrument Digital Interface) — цифровий інтерфейс музичних інструментів

ANN (Artificial Neural Network) — штучна нейронна мережа

ГА — генетичний алгоритм

PSO (Particle Swarm Optimization) — метод рою часток

RHM (Рекурентні нейронні мережі) — клас штучних нейронних мереж, у якому з'єднання між вузлами утворюють граф, орієнтований у часі.

LSTM (Long Short-Term Memory) — довга короткочасна пам'ять

Контрапункт — одночасне поєднання двох і більше мелодій в різних голосах

Бас-барабан (англ. bass drum) — барабан стандартної ударної установки з найнижчим регістром

Робочий барабан (англ. snare drum) — основний інструмент ударної установки

Хай-хет (англ. hi-hat) — дві тарілки, встановлені на одному стрижні і керовані педаллю

					ІТ-62.08.1081.01 ПЗ	Арк.
						8
Змн.	Арк.	№ докум.	Підпис	Дат.		

ВСТУП

У сучасному світі системи штучного інтелекту все швидше набирають обертів. Нейронні мережі застосовуються в багатьох сферах людського життя, як із серйозною науковою, так і з розважальною метою. Зокрема штучний інтелект все частіше використовується в мистецтві. Роботи малюють картини, пишуть сценарії та вірші, допомагають з дизайном різноманітних речей, пишуть музику тощо.

Однак очевидно, що комп'ютер не в змозі повністю замінити людину, особливо коли мова йде про мистецтво. Так, ті ж музичні твори, написані штучним інтелектом, далеко не завжди звучать гармонійно, та й для завершення більш вдалих потрібна допомога людини. Але якщо не ставити на меті створення чогось нового, а запрограмувати комп'ютер лише для допомоги, можна добитися досить непоганих результатів.

Дана робота спрямована на те, щоб розробити інтелектуальну систему, яка допомагатиме в аранжуванні композицій. Майже вся сучасна музика має чітку ударну партію, складність та звучання якої залежать від жанру. Вдало підібравши ударну партію до свого твору, музикант збільшує потенційну кількість слухачів, адже якою б гармонійною не видавалась мелодія, слабка ритмічна партія може зруйнувати весь ефект. До того ж далеко не всі стандартні ритми можна успішно застосувати до новонаписаного твору через співпадіння ритмів, невідповідність жанрів, порушення поліфонії тощо. Саме тому композиторам-початківцям, які не вивчали принципи аранжування, необхідна допомога при підборі цікавого і захоплюючого слухачів ритму.

Метою даної роботи є розробка та створення інтелектуальної системи, яка підбиратиме ударну партію до записаної користувачем композиції на основі проаналізованих існуючих пісень.

Для досягнення поставленої мети були визначені такі задачі:

					ІТ-62.08.1081.01 ПЗ	Арк.
						9
Змн.	Арк.	№ докум.	Підпис	Дат.		

- аналіз існуючих рішень інтелектуальних систем для роботи з музичними творами та виявлення їхніх недоліків;
- дослідження можливостей створення інтелектуальної системи для підбору ударної партії до заданої музичної композиції;
- розробка архітектури обраного рішення;
- проектування штучної нейронної мережі для аналізу існуючих ритмічних патернів;
- реалізація інтелектуальної системи підбору ударної партії для аранжування пісень;
- тестування розробленої системи та аналіз якості її роботи шляхом опитування слухачів.

Об'єктом дослідження є системи штучного інтелекту для роботи з музикою.

Предметом дослідження є залежність ритмічних патернів барабанів від ритміки інших інструментів.

Практична цінність роботи полягає у тому, що вихідну програму зможуть використовувати композитори-початківці, які не вивчали правила аранжування. Крім того, програма може бути корисною для музичних груп, які підбирають ритмічні партії для своїх пісень.

					ІТ-62.08.1081.01 ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дат.		10

РОЗДІЛ 1. ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Загальний огляд систем штучного інтелекту для роботи з музикою

1.1.1. Історія

Першим музичним твором, написаним комп'ютером, вважається «Сюїта Ілліака для струнного квартету», написана комп'ютером ILLIAC, що є однією з першим ЕОМ у світі, в 1957 році. Комп'ютер запрограмували Л. Гіллер та Л. Айзексон, які описують сюїту як хронологічний запис експериментів. Загальна ідея полягала у використанні правил екранування для прийняття або відхилення випадково створених пітчів та ритмів. Також у сюїті використовувалися розподіл ймовірностей та процеси Маркова.

Прикладами ранньої комп'ютерної музики можуть також слугувати «Стохастичний струнний квартет» Дж. Тенні 1963 року та «Серенада» Т. ДеЛіо, створена з використанням ланцюгів Маркова [1].

У 70-х процес алгоритмізації створення музики зацікавив також поп-виконавців. Одним з перших, хто почав думати в цьому напрямку, став Девід Боуї. Разом з Т. Робертсом він розробив Verbasizer – програму, в яку можна було завантажити слова і фрази, а вона переставляла їх місцями, створюючи нові форми. Результати роботи цієї програми можна почути в треках «Берлінської трилогії». Боуї продовжував використовувати її і в майбутньому. Вона допомогла йому написати, наприклад, текст пісні «Outside».

Вивчення можливості писати музику за допомогою комп'ютерів продовжувалося, і реальний результат з'явився в 1980 році в Університеті Каліфорнії. Професор і композитор Девід Коуп розробив систему, яка навчилася копіювати його стиль та стилі інших композиторів і створювала на цій основі власні твори. Систему назвали ЕМІ (читається як «Еммі»), і вона була досить успішною. Але Коуп не зупинився на досягнутому. Пізніше він

					ІТ-62.08.1081.01 ПЗ	Арк. 11
Змн.	Арк.	№ докум.	Підпис	Дат.		

створив Емілі Хоуелл (продовження роботи ЕМІ) – віртуальну композиторку, яка розробила власний стиль музики.

З появою сучасних комп'ютерів і технологій до процесу створення систем написання музики підключається все більше людей і компаній. Співачка Тарін Саузерн, наприклад, вже випустила альбом, повністю написаний штучним інтелектом [2].

1.1.2. Огляд алгоритмів для написання музики штучним інтелектом

Для написання музики штучним інтелектом використовуються різноманітні алгоритми та підходи.

1) Генетичний алгоритм для акомпанементу.

Генетичні алгоритми (ГА) показали свою ефективність для розв'язання різноманітних проблем. Загальний принцип ГА симулює механізми природної еволюції, такі як селекція, кросовер та мутація. ГА можна використати для генерації основного акомпанементу, басового та акордового акомпанементу.

Музичні такти представлені рядом чисел для хромосоми в ГА. Відповідно до 12 рівних темперментів Баха, кожна октава ділиться на 12 рівних нот, позначених символами C, #C, D, #D, E, F, #F, G, #G, A, #A та B. Для представлення у хромосомі кожна нота позначається відповідним числом.

По-перше, запропонований метод породжує цільову функцію, що базується на ритмі та висоті заданої домінуючої мелодії. ГА з цією функцією розвивається в основному акомпанементі. По-друге, він використовує інформацію акорда для встановлення другої цільової функції і запускає ГА для створення басового акомпанементу. По-третє, ми повторюємо вищевказану процедуру для створення третьої функції, створеної за допомогою інформації домінуючої мелодії та основного акомпанементу. Потім акорд може вироблятися через ГА з третьою цільовою функцією. Нарешті, ми додаємо прості перкусії для покращення структури створеної музики [3].

					ІТ-62.08.1081.01 ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дат.		12

2) Ройові алгоритми для написання музики.

Одним з методів генерації музичних уривків є використання так званого ройового інтелекту. Ройовий інтелект (англ. Swarm intelligence) — метод оптимізації, що використовується в теорії штучного інтелекту і описує колективну поведінку децентралізованої системи, що самоорганізовується. Існує велика кількість методів оптимізації, що спираються на колективний інтелект — метод рою частинок, бджолиний алгоритм, оптимізація пересуванням бактерій, світляковий алгоритм, алгоритм крапель води тощо.

За музикою, яку згенерували за допомогою цих методів, закріпилася назва Swarm Music (ройова музика). Одним з головних фахівців, які досліджували цей метод для формалізації музичних процесів, є Тім Блеквелл з університету Суссексу. Він застосував в своїх дослідженнях метод рою часток (Particle Swarm Optimization) — один з ефективних поліноміальних алгоритмів для вирішення задач оптимізації.

Ідея Блеквелла полягає в призначенні часткам характеристик, що описують музичні події: нота здатна проходити певний шлях в просторі музичних параметрів. Поки вона переміщається в цьому просторі, змінюються її характеристики: висота, гучність, тривалість (рис. 1.1). Вона проходить цей шлях не безцільно, а притягується до інших нот-частинок, і незабаром формуються групи нот, з яких зникають інші ноти, групи об'єднуються з іншими, що утворилися в іншій ділянці простору. Ці рої мелодій збираються з нот, які «не знають», що вони частина великого мотиву [4].

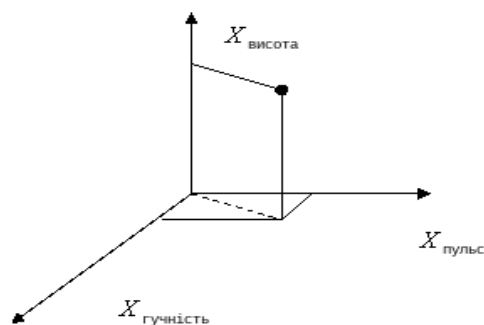


Рисунок 1.1. Представлення ноти у вигляді частки за Блеквеллом

3) Нейронні мережі.

Для написання музичних творів використовуються також нейронні мережі. Загальний принцип полягає в тому, що нейронну мережу навчають на існуючих творах композиторів. Мережа аналізує співвідношення висот, тривалостей та інших показників нот і пізніше може «передбачати» нотні послідовності, створюючи таким чином музику.

В основі мережі лежить не людське розуміння і сприйняття музики, а навик знаходити зразки гармонії, ритму і стилю в сотнях тисяч MIDI-файлів, а потім створювати власні твори. Прикладом такої мережі є нейронна мережа MuseNet, яка може передбачати найкращі послідовності як в нотному тексті, так і в аудіо. Знання безлічі стилів дозволяє мережі створювати абсолютно унікальні комбінації музики різних поколінь.

1.2. Ритми у комп'ютерній музиці

Якими б не були досягнення у сфері написання музики штучним інтелектом, більшість таких систем спрямована на підбір гармонії та акомпанементу або написання музики з нуля. Натомість молодим композиторам-початківцям потрібна допомога саме з аранжуванням мелодії, причому часто вони вже мають уявлення про те, як і в якому стилі має звучати композиція у фінальному варіанті. Від штучного інтелекту їм може знадобитися певний поштовх у потрібному напрямку, підказка про те, як заволодіти увагою слухача.

Одним з найважливіших елементів пісні, як і будь-якої музичної композиції, є ритмічна складова. Навіть якщо ударні не присутні у творі, завжди є певний розмір та прихована пульсація твору, які можна легко помітити, проаналізувавши композицію. Чітко та грамотно організованим ритмом можна перекрити недоліки мелодії та гармонії і привабити слухача.

					ІТ-62.08.1081.01 ПЗ	Арк.
						14
Змн.	Арк.	№ докум.	Підпис	Дат.		

Однак для цього ритміку ударних треба вдало пов'язати з ритмікою інших інструментів.

Коли штучний інтелект створює музику, він зазвичай використовує певні патерни для розташування барабанів у площині пісні. Але це не гарантує того, що ритміка ударних вдало поєднається з ритмікою мелодії у заданому музичному стилі.

Саме тому предметом дослідження даної роботи було обрано ритмічні патерни барабанів у залежності від ритміки інших інструментів на основі існуючих пісень.

1.3. Огляд існуючих рішень проблеми підбору ритмічної партії

1.3.1. Алгоритм Евкліда для генерації ритмів

Алгоритм Евкліда обчислює найбільший спільний дільник двох заданих цілих чисел. Однак структура алгоритму Евкліда може бути використана для генерації великого сімейства ритмів, що повторюються (остинати), африканської музики зокрема і світової музики в цілому.

В роботі [5] описується алгоритм, який максимально рівномірно розподіляє задану кількість акцентів на певну кількість долей. Наприклад, потрібно розподілити 5 бітів на 13 долей. У початковому вигляді розставимо акценти таким чином:

[x x x x x],

де «x» – удар, «.» – тиша. На наступному кроці розподіляємо неакцентовані біля акцентованих, а три долі тиші лишаємо на кінці:

[[x .] [x .] [x .] [x .] [x .] [.] [.] [.]].

Далі повторюємо операцію із залишковими неакцентованими долями:

[[x . .] [x . .] [x . .] [x .] [x .]].

Тепер маємо послідовність із трьох елементів «удар-тиша-тиша» і залишок з двох елементів «удар-тиша». Маємо розподілити залишок:

[[x . . x .] [x . . x .] [x . .]].

Таким чином, отримуємо ритм [x . . x . x . . x . .].

Такий підхід доволі ефективний і надає цікаві та доречні результати. Однак в даній роботі генерація ритмів не враховує накладання мелодії та її ритміки на існуючу ударну партію.

1.3.2. Генерація ритмів за допомогою рекурентних нейронних мереж

Рекурентні нейронні мережі — це клас штучних нейронних мереж, у якому з'єднання між вузлами утворюють граф, орієнтований у часі. Це створює внутрішній стан мережі, що дозволяє їй проявляти динамічну поведінку в часі. На відміну від нейронних мереж прямого поширення, РНМ можуть використовувати свою внутрішню пам'ять для обробки довільних послідовностей входів [6].

В роботі [7] РНМ використовується для генерації ритмів у стилі техно. Барабанні цикли MIDI різних жанрів зчитувалися, квантувалися до тридцять других нот і перетворювалися у матричну форму (рядки – час, стовпці – номери нот MIDI, комірки – тривалість). Використовувався розмір 4/4. Усі петлі були поєднані в тому порядку, в якому вони з'явилися у файловій системі. У кінцевому результаті в навчанні нейронної мережі брало участь близько 27000 32-х нот. Для кодування ударних партій було використано чотири різні типи барабанів (Kick, Snare, Closed Hihat, Open Hihat), і їхня бінарна присутність або відсутність кодувалась як число між 0 і 15.

Мережу було навчено передбачати, яка нота (або комбінація нот, або тиша) буде наступною, враховуючи попередні ноти. Для генерації ритму було використано рекурентну мережу з довгою короткочасною пам'яттю (LSTM).

Згенеровані таким чином ритми вийшли незвичними, різноманітними та цікавими для прослуховування. Однак, знову ж таки, вони генерувалися окремо від основної мелодії музичного твору. Такий підхід можна використовувати,

коли написання пісні починається не з мелодії, а з ритму, що відбувається досить рідко.

Ще одним прикладом генерації ритмів за допомогою РНМ з LSTM може слугувати робота [8].

1.4. Постановка задачі

Штучний інтелект на сьогоднішній день широко застосовується у сфері мистецтва в цілому та в музичній галузі зокрема. Інтелектуальні системи пишуть музику різних стилів для оркестру, сольних виконавців, музичних гуртів. Існує також чимало рішень в напрямку генерації ударної партії. Однак у підходах, розглянутих в розділі 1.3, було виявлено декілька недоліків, а саме:

- в існуючих рішеннях ритми генеруються за певними алгоритмами, але в них відсутній творчий елемент у вигляді інтелектуальності системи: усі результуючі ритми схожі між собою, і в них немає різноманіття;
- існуючі інтелектуальні системи генерують різноманітні ритми в різних жанрах, але не враховуючи при цьому партії інших інструментів: komponувати і знаходити вдалі комбінації мелодійних та ударних партій доводиться композиторам власноруч.

Таким чином, для усунення цих недоліків було вирішено створити інтелектуальну систему підбору ударної партії, яка б могла працювати з різними музичними жанрами та генерувала б барабанну партію таким чином, щоб вона вдало контрастувала з мелодійною. Для реалізації такої системи були поставлені наступні задачі:

- розглянути принципи роботи з контрапунктом (поєднання та протиставлення різних голосів і/або партій в музичних творах), дослідити роботу з ударною установкою;
- обрати та обґрунтувати використання певних технічних рішень;
- обрати та обґрунтувати архітектуру та структуру системи;

- реалізувати систему підбору ударної партії;
- експериментально перевірити якість роботи системи та зробити відповідні висновки.

ВИСНОВКИ ДО РОЗДІЛУ

В першому розділі роботи було досліджено предметну область, а саме:

1. Оглянуто історію використання штучного інтелекту у роботі з музикою.
2. Розглянуто підходи до аналізу музичних творів, які використовуються системами штучного інтелекту, а саме:
 - 2.1. Генетичний алгоритм для підбору гармонійного акомпанементу.
 - 2.2. Використання ройового інтелекту (swarm intelligence) для написання музичних композицій.
 - 2.3. Використання нейронних мереж.
3. Досліджено методи, які використовуються для генерації ритмів у різних системах для роботи з музикою, а саме:
 - 3.1. Алгоритм Евкліда для генерації ритмів.
 - 3.2. Використання рекурентних нейронних мереж.
4. Виявлено недоліки підходів до генерації ударних партій, а саме: недостатня жанрова різноманітність результатів та неврахування мелодійної партії інших інструментів при генерації барабанної партії.

На основі даних досліджень було вирішено розробити та реалізувати інтелектуальну систему підбору ритмічних партій для аранжування композицій. Система має орієнтуватись на музичну партію, до якої підбирається біт, та її акценти, а також працювати за принципами контрапункту. Система має бути гнучкою, тобто варіювати згенеровані ритми у різних жанрах, підбирати ударну партію для музичних уривків будь-якої довжини, надавати користувачеві можливість повторного підбору, якщо його не задовольнили результати.

					ІТ-62.08.1081.01 ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дат.		18

Для реалізації системи необхідно розглянути різні можливі технічні рішення та обґрунтувати вибір тих чи інших рішень та інструментів для створення системи.

Система має бути протестована шляхом проведення експериментів для обчислення точності її роботи, а також шляхом опитування слухачів.

Результуюча система стане в нагоді композиторам-початківцям, музичним гуртам, що працюють у різних жанрах та аранжують пісні, а також усім людям, яким цікаво працювати з музикою та програмами, що допомагають у її написанні.

					ІТ-62.08.1081.01 ПЗ	Арк.
						19
Змн.	Арк.	№ докум.	Підпис	Дат.		

РОЗДІЛ 2. АНАЛІЗ ПРОБЛЕМ ПРИ РЕАЛІЗАЦІЇ СИСТЕМИ ПІДБОРУ УДАРНОЇ ПАРТІЇ ДЛЯ АРАНЖУВАННЯ КОМПОЗИЦІЙ

2.1. Складності роботи з контрапунктом

Контрапункт (лат. *Punctus contra punctum* — точка навпроти точки) — одночасне поєднання двох і більше мелодій в різних голосах. Простим контрапунктом називається з'єднання мелодій, яке звучить тільки один раз. Якщо при повторі відбуваються які-небудь зміни в самих мелодіях або в їх співвідношенні відносно одна одної, то такий контрапункт буде називатися складним.

Рухомий контрапункт — вид складного контрапункту, в якому похідне з'єднання утворюється в результаті зміни співвідношення мелодій шляхом їх руху відносно одна одної по горизонталі, вертикалі і одночасно по обох координатах (рис. 2.1). Самі мелодії при цьому не змінюються. Різновиди рухомого контрапункту: вертикально-рухомий, горизонтально-рухомий, подвійно-рухомий. У перестановках можуть брати участь два голоси (подвійний контрапункт), три голоси (потрійний контрапункт), чотири і навіть п'ять голосів [9].

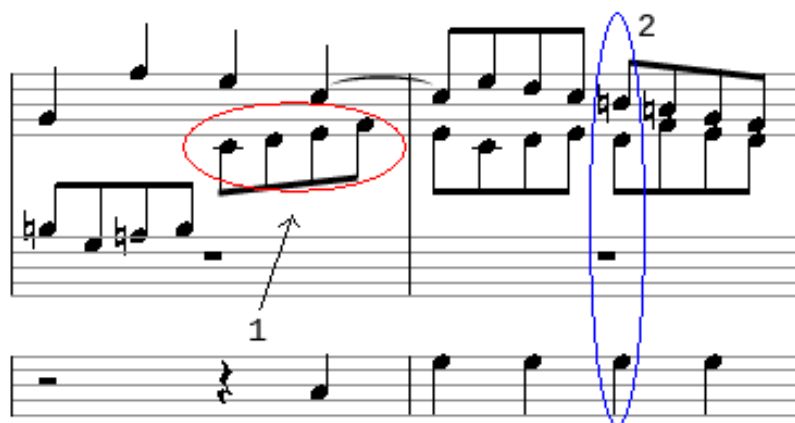


Рисунок 2.1. Види контрапункту: 1) горизонтальний; 2) вертикальний

У 1725 році австрійський композитор Йоганн Йозеф Фукс опублікував теоретичну працю «Gradus ad Parnassum» («Сходи до Парнасу»), де він описав п'ять видів контрапункту:

- нота навпроти ноти;
- дві ноти навпроти однієї;
- чотири ноти навпроти однієї;
- ноти мають зміщення відносно одна одної (синкопування);
- поєднання попередніх чотирьох підходів.

Контрапунктичний стиль в музиці найяскравіше представлений в хорових творах Палестріні (бл. 1525-1594) і в інструментальних і хорових творах І. С. Баха (1685-1750) [10].

Коли мова йде про ритм, проблеми пошуку гармонійного співзвучання залишаються тими ж самими. Необхідно поєднати ритміку мелодії з ритмікою ударної партії.

2.2. Розміри барабанної установки

Стандартна ударна установка складається з барабанів та тарілок (рис. 2.2).



Рисунок 2.2. Стандартна ударна установка: 1) тарілки 2) підлоговий том-том 3) том-том 4) бас-барабан 5) робочий барабан 6) хай-хет

Серед тарілок виділяють наступні:

- креш (англ. crash) — тарілка з потужним шиплячим звуком;
- райд (англ. ride) — тарілка з дзвінким, але коротким звуком для акцентів;
- хай-хет (англ. hi-hat) — дві тарілки, встановлені на одному стрижні і керовані педаллю.

В установці присутні такі барабани:

- робочий барабан (англ. snare drum) – основний інструмент установки;
- три том-томи: високий том-том (англ. high tom-tom), низький том-том (англ. middle tom-tom), підлоговий том-том (або просто том, англ. floor tom-tom).
- бас-барабан («бочка», англ. bass drum).

Звісно, в залежності від жанру або виконавця склад барабанної установки може змінюватися.

Для підбору ударної партії до пісні в цій роботі використовуватиметься наступний склад барабанної установки: бас-барабан, робочий барабан та хай-хет. Ці три елементи є основними в ударній установці.

Якщо розглядати повну ударну установку, постає проблема занадто великої кількості даних для аналізу. Натомість три основних елементи установки легко можна почути в будь-якій пісні та проаналізувати ритм, який вони утворюють. Крім того такий поверхневий розгляд ритмічної партії дає більш узагальнений результат. Тобто, коли користувач вводитиме мелодійну партію і отримуватиме в результаті ударну партію з бас-барабана, робочого барабана і хай-хета, він легко зможе заграти запропоновану партію, додати до неї власні штрихи за допомогою інших елементів ударної установки та підлаштувати запропоновану партію таким чином до необхідного музичного жанру.

					ІТ-62.08.1081.01 ПЗ	Арк.
						22
Змн.	Арк.	№ докум.	Підпис	Дат.		

2.3. Обґрунтування вибору формату MIDI для роботи з композиціями

MIDI (англ. Musical Instrument Digital Interface, цифровий інтерфейс музичних інструментів) — стандарт передачі інформації між електронними музичними інструментами, розроблений 1983 року. Він надає можливість електронним музичним інструментам взаємодіяти з комп'ютером та іншим обладнанням, сумісним з MIDI, чи взаємодіяти між собою, здійснювати управління іншими інструментами з одного.

MIDI не передає та не генерує звук — натомість MIDI працює через «події», такі як Note On/Note Off (натискання/відпускання ноти), pitch (висота) та динаміка взятої ноти на інструменті; контрольні сигнали для таких параметрів як панорама, гучність, сигнали відліку часу для синхронізації темпу тощо. Музичний інструмент приймає повідомлення про такі події і генерує звук. Інструментом може бути як реальний пристрій, наприклад, синтезатор, так і віртуальний, наприклад, програма на комп'ютері. Як електронний протокол MIDI відзначається надзвичайно широким поширенням.

Запис та відтворення MIDI базується на пакетах даних, кожний з яких відповідає MIDI-події (англ. MIDI-events).

Пристрій для запису та редагування MIDI-повідомлень називається секвенсер (від англ. sequence — послідовність). MIDI-події вводяться через MIDI-клавіатуру (в покроковому режимі або в реальному часі) або ж різноманітними способами з клавіатури чи мишкою у програмі-секвенсері. Послідовність MIDI-подій може бути збережена як MIDI-файл (файл має розширення *.mid).

Переваги формату MIDI:

- компактність: ціла пісня може бути передана лише за допомогою декількох сотень MIDI-подій (в порівнянні з аудіо-даними, які семплуються кілька тисяч разів на секунду);

					ІТ-62.08.1081.01 ПЗ	Арк.
						23
Змн.	Арк.	№ докум.	Підпис	Дат.		

- легко маніпулювати: легко відредагувати або змінити ноти, їх тривалість, висоту без необхідності перезаписувати всю частину;
- зміна інструментів: оскільки MIDI просто описує, яку ноту грати, то як саме буде звучати ця нота — обирає автор. Змінюючи інструменти, можна змінити загальний звук композиції [11]:

Файли MIDI організовані у фрагменти даних. Кожен фрагмент має префікс із 8-байтним заголовком: 4-байтний рядок з ID, що використовується для ідентифікації типу фрагмента, а потім 4-байтний розмір, який вказує на довжину фрагмента, тобто кількість байтів, що слідують за заголовком цього фрагмента.

Заголовок містить інформацію про всю пісню, включаючи тип формату MIDI, кількість треків та розподіл часу. У стандартному MIDI-файлі є лише один заголовок, і він завжди стоїть на першому місці.

Фрагменти треків містять усю інформацію про окремий трек, включаючи назву трека та музичні події. Перелік MIDI-подій наведено у табл. 2.1.

Таблиця 2.1. MIDI-події

Тип події	Значення	Параметр 1	Параметр 2
Note Off	0x8	Номер ноти	Гучність
Note On	0x9	Номер ноти	Гучність
Note Aftertouch	0xA	Номер ноти	Сила натиснення
Controller	0xB	Номер контролера	Значення контролера
Program Change	0xC	Номер програми	—
Channel Aftertouch	0xD	Сила натиснення	—
Pitch Bend	0xE	Висота ноти (LSB)	Висота ноти (MSB)

Нумерація нот у файлах MIDI відбувається за їхньою висотою. Найнижча можлива нота — нота «до» -1-ої октави (на стандартному фортепіано, що має 88 клавіш, такої октави не існує, найнижча октава — нульова, субконтроктава) — має номер 0, найвища нота — нота «соль» 8-ої октави має номер 127.

Номери нот у файлах MIDI наведені в табл. 2.2.

Таблиця 2.2. Номери нот у файлах MIDI.

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

Таким чином, формат MIDI найкраще підходить для вирішення задач, поставлених у цій роботі.

Для аналізу вхідних даних користувача буде створено програму, написану мовою Java, із використанням пакету `javax.sound.midi`. Пакет містить усі необхідні інструменти для роботи з файлами формату MIDI, що допоможе легко конвертувати музичні дані в числові (за алгоритмом, описаним в розділі 3.1) та навпаки.

2.4. Обґрунтування вибору нейронної мережі для аналізу існуючих композицій

Оскільки для коректної та якісної роботи системи планується проаналізувати залежність ритмічних патернів від акцентів у мелодійних партіях, для такого дослідження можна використати регресійний аналіз.

Регресійний аналіз може допомогти змодельовати взаємозв'язок між залежною змінною, яку потрібно передбачити (у випадку даної роботи це ударна партія), та однією або кількома незалежними змінними (акценти мелодійної партії). Регресійний аналіз може показати, чи існує значна залежність між незалежними змінними та залежною змінною. Крім того, на основі результатів аналізу можна передбачити значення залежної змінної для певних нових незалежних змінних.

Такий принцип частково нагадує метод навчання «з учителем» для штучних нейронних мереж. Штучні нейронні мережі (англ. Artificial Neural Networks, ANN) складаються з простих елементів, що називають нейронами, кожен з яких може приймати прості математичні рішення. Разом нейрони можуть аналізувати складні проблеми, імітувати майже будь-яку функцію, включаючи дуже складні, та давати точні відповіді. Неглибока нейронна мережа має три шари нейронів: вхідний шар, прихований шар та вихідний шар. Глибока нейронна мережа має більше ніж один прихований шар, що збільшує складність моделі та може значно підвищити потужність прогнозування.

Нейронні мережі можна представити у якості регресійних моделей. Наприклад, на рис. 2.3 зображена дуже проста нейронна мережа, що має лише один вхідний нейрон, один прихований нейрон і один вихідний нейрон. Така мережа еквівалентна логістичній регресії. Вона приймає кілька залежних змінних (вхідні параметри), помножує їх на їх коефіцієнти (ваги) та пропускає їх через сигмоїдальну функцію активації та функцію Хевісайда [12].

Для простої регресійної моделі проєктувати нейронну мережу недоречно і ресурсозатратно. Однак використання нейронних мереж для більш складних регресійних задач може дати цікаві та цінні результати. Регресія працює коректно лише за умови, що вдасться вдало підібрати функцію, яка б описувала залежність набору даних. Нейронні мережі більш гнучкі у цьому плані.

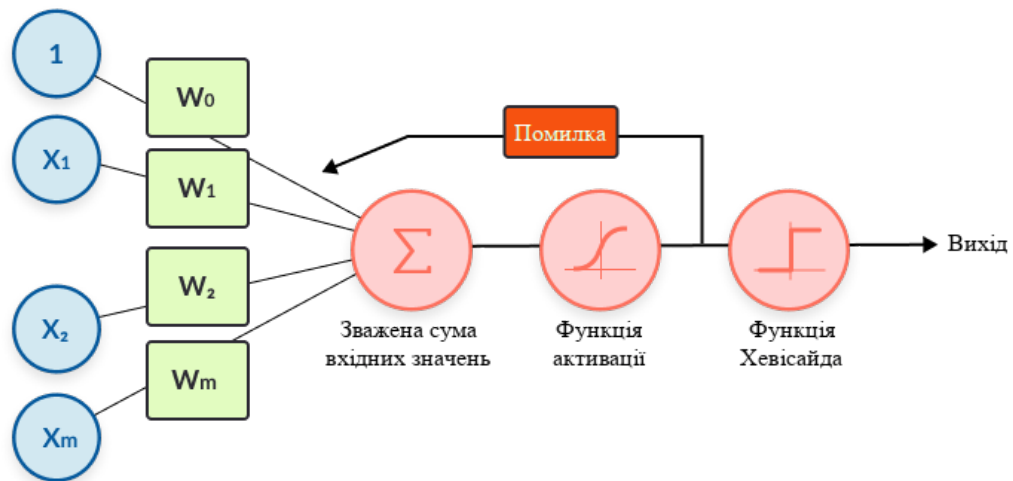


Рисунок 2.3. Регресійна модель у вигляді нейронної мережі

Змінюючи кількість прихованих шарів і/або кількість нейронів у них, можна отримати більш точні результати та збільшити точність передбачень.

Саме тому для системи підбору ударної партії було вирішено спроектувати та навчити нейронну мережу, яка реалізувала б певну регресійну модель. У якості вхідних параметрів виступатимуть акценти мелодійної партії, у якості вихідних параметрів — партії бас-барабану, робочого барабану та хай-хета.

2.5. Проблема аналізу вихідних даних

Для навчання нейронної мережі необхідна досить велика вибірка даних. Для даної роботи було вирішено проаналізувати ритміку пісень розміру 4/4 різних жанрів. Оскільки головною метою роботи є підбір ударної партії згідно з правилами контрапункту, аналіз пісень має проходити в два етапи:

1. Аналіз мелодійної партії та її ритміки.
2. Аналіз відповідного ритму в ударній партії.

Однак постає проблема в тому, щоб знайти велику кількість пісень MIDI-формату, які можна було б описати простими математичними структурами, з якими пізніше матиме змогу працювати нейронна мережа. До того ж, якщо аналізувати кожну пісню повністю, на це витрачатиметься багато часу та ресурсів.

Тому для даної роботи процес аналізу та кодування даних було вирішено максимально спростити.

Ідея полягає в тому, щоб аналізувати не всю пісню цілком, а окремі її такти. Наприклад, маємо кілька тактів куплету з більш спокійною мелодійною лінією, яка підкреслена ненав'язливою ударною партією. Такі такти аналізуються окремо і вважаються одним зразком для подальшого навчання нейронної мережі. Далі аналізуються такти приспіву, де мелодія стає більш насиченою і в ударній партії додається більша кількість барабанів. Ці такти також аналізуються окремо і вважаються одним зразком для подальшого навчання нейронної мережі.

Таким чином фінальна вибірка для навчання нейронної мережі складатиметься з математично описаних тактів. На вхід подаватиметься ритміка мелодійної партії, на виході отримуватиметься відповідна ударна партія.

Така схема є макимально простою та доречною. Оскільки основних ритмічних патернів є не так багато, для навчання мережі знадобиться відносно невелика кількість вихідних даних. Крім того, коли користувач записуватиме власну мелодійну партію, до якої розроблювана система матиме підібрати ритм, цю партію достатньо розбити на такти і підібрати найбільш підходящий ритм для кожного з них.

					ІТ-62.08.1081.01 ПЗ	Арк.
						28
Змн.	Арк.	№ докум.	Підпис	Дат.		

2.6. Деталізація постановки задачі

1. Формати даних.

1.1. Дані користувача вводяться у форматі MIDI.

1.2. Нейронна мережа працює з числовими форматами даних: векторами та масивами цифр.

1.3. Результат роботи нейронної мережі подається у числовому форматі даних.

1.4. Результат роботи програми подається користувачу у форматі MIDI.

1.5. Робота з форматом MIDI відбувається мовою програмування Java за допомогою пакета `javax.sound.midi`.

2. Нейронна мережа.

2.1. Нейронна мережа має реалізовувати регресійну модель (взаємозв'язок між акцентами мелодійної партії та барабанною партією).

2.2. Навчання нейронної мережі відбувається «з учителем» за допомогою методу зворотнього поширення помилки.

2.3. Нейронна мережа написана мовою програмування R за допомогою бібліотеки `neuralnet`.

3. Алгоритми конвертації даних.

3.1. Розробити алгоритм конвертації формату MIDI у числовий формат для навчання нейронної мережі та подальшого її використання.

3.2. Розробити алгоритм конвертації числового результату роботи нейронної мережі у формат MIDI.

4. Вимоги до якості продукту.

4.1. Помилка роботи нейронної мережі має бути мінімальною.

4.2. Виключні ситуації при роботі з файлами MIDI мають бути передбачені та оброблені.

4.3. Результати опитування слухачів мають бути задовільними.

					ІТ-62.08.1081.01 ПЗ	Арк. 29
Змн.	Арк.	№ докум.	Підпис	Дат.		

ВИСНОВКИ ДО РОЗДІЛУ

В другому розділі роботи було проаналізовано проблеми, які виникають при реалізації інтелектуальних систем для роботи з музикою, та розроблено функціональні вимоги до системи підбору ударних партій.

Для спрощення аналізу існуючих співвідношень барабанних та мелодійних акцентів було вирішено враховувати лише три основних елементи барабанної установки: бас-барабан, робочий барабан та хай-хет. Розроблювана система має враховувати принципи контрапункту.

Існуючі пісні аналізуватиме нейронна мережа, що реалізує регресійну модель, оскільки головною метою навчання нейронної мережі є встановлення залежності між вхідними (мелодійна партія) та вихідними (ударна партія) параметрами. Аналіз творів відбуватиметься не цілком, а по тактах, що має на меті зменшити обсяг навчальної вибірки та спростити структуру нейронної мережі. Для зручності та гнучкості форматом вводу-виводу музичних даних було обрано формат MIDI.

Для реалізації системи використовуватимуться наступні технічні засоби:

- мова програмування Java та пакет `javax.sound.midi`;
- мова програмування R та бібліотека `neuralnet`;
- середовище розробки IntelliJ IDEA.

					ІТ-62.08.1081.01 ПЗ	Арк.
						30
Змн.	Арк.	№ докум.	Підпис	Дат.		

РОЗДІЛ 3. РОЗРОБКА АРХІТЕКТУРИ СИСТЕМИ ПІДБОРУ УДАРНОЇ ПАРТІЇ ДЛЯ АРАНЖУВАННЯ КОМПОЗИЦІЙ

3.1. Кодування вихідних даних

Для навчання нейронної мережі було вирішено проаналізувати ритміку пісень розміру 4/4 жанрів рок, метал та поп-рок. В першу чергу аналізувалася інструментальна (гітарна) партія, потім — ударна партія.

Як було сказано в розділі 2.5, вихідними даними слугують такти, описані математично. У такт розміром 4/4 можна помістити різну кількість нот в залежності від їхніх тривалостей (рис. 3.1): одна ціла, дві половинні, чотири четвортні, вісім восьмих, шістнадцять шістнадцятих, тридцять дві тридцять других тощо. В даній роботі такт розміром 4/4 буде вважатися набором з шістнадцяти долей, кожна з яких триває одну шістнадцяту ноту.



Рисунок 3.1. Нотні тривалості

Оскільки основна мета роботи полягає в підборі ритміки, необхідно проаналізувати співвідношення акцентів мелодійної та ударної партій.

1) Кодування акцентів мелодійної партії.

Для мелодійної партії будемо представляти такт у вигляді одновимірного масиву з 16 елементів, що містить інформацію про акценти. Якщо доля неакцентована, відповідний елемент масиву містить 1, якщо доля акцентована – 2. Якщо на місці долі пауза або продовжує звучати попередня доля, в масив записується 0. Наприклад, закодуємо акценти для мелодії, наведеної на рис. 3.2.



Рисунок 3.2. Приклад мелодії для кодування

Етап 1. Запишемо початковий масив, що складається повністю з пауз (нулів): [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0].

Етап 2. Кожну наявну ноту на місці відповідної їй долі позначимо одиницею: [1 0 0 0 1 0 1 0 1 1 1 0 1 0 1 0].

Етап 3. Акцентовані ноти (символ «>») позначимо цифрою 2:

[2 0 0 0 1 0 2 0 1 1 1 0 2 0 1 0].

Таким чином отримуємо масив із шістнадцяти цифр, у якому закодовані акцентовані і неакцентовані ноти наведеного в прикладі такту.

2) Кодування ударної партії.

Припустімо, що до мелодії, наведеної на рис. 3.2, підібрана наступна ударна партія (рис. 3.3):



Рисунок 3.3. Приклад ударної партії для кодування

Ударна партія складається з трьох елементів барабанної установки: бас-барабан (base), робочий барабан (snare) та хай-хет (hi-hat). Кожен інструмент має свою нотну лінію, де символічно записані партії. Символами «>» позначено акцентований (сильний) удар.

Ударна партія кодується аналогічним до мелодійної чином. Але, враховуючи, що інструментів в цій партії три, масив буде не одновимірним, а двовимірним. Закодуємо ударну партію:

Етап 1. Закодуємо в одновимірний масив партію бас-барабану, ставлячи 0 замість паузи, 1 на неакцентований удар, 2 – на акцентований удар:

[0 0 2 0 0 0 0 0 0 0 1 0 0 0 0 0].

Етап 2. Аналогічним чином кодуємо партію робочого барабану:

[0 0 0 0 2 0 0 0 0 0 0 0 0 0 1 0].

Етап 3. Аналогічним чином кодуємо партію хай-хета:

[2 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0].

Етап 4. Зводимо отримані одновимірні масиви в один двовимірний. Отримуємо наступний результат:

$$\begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Таким чином, дані для навчання нейронної мережі будуть представлені у вигляді пар мелодійна партія — ударна партія, де мелодійна партія представлена у вигляді одновимірного масиву з 16 елементів, а ударна партія — з двовимірного масиву розміром 3х16.

3.2. Проектування нейронної мережі

Оскільки ударна партія, що буде підбирається розробленою в даній роботі системою, складається з партій трьох елементів (бас-барабан, робочий барабан та хай-хет), нейронних мереж для підбору барабанної партії також три (рис. 3.4). Перша мережа підбирає партію бас-барабану на основі мелодійної партії, друга підбирає партію робочого барабану на основі мелодійної партії та підібраного на попередньому кроці бас-барабану, третя підбирає партію хай-хета на основі мелодійної партії та підібраних на попередніх кроках партій бас-барабану та робочого барабану.

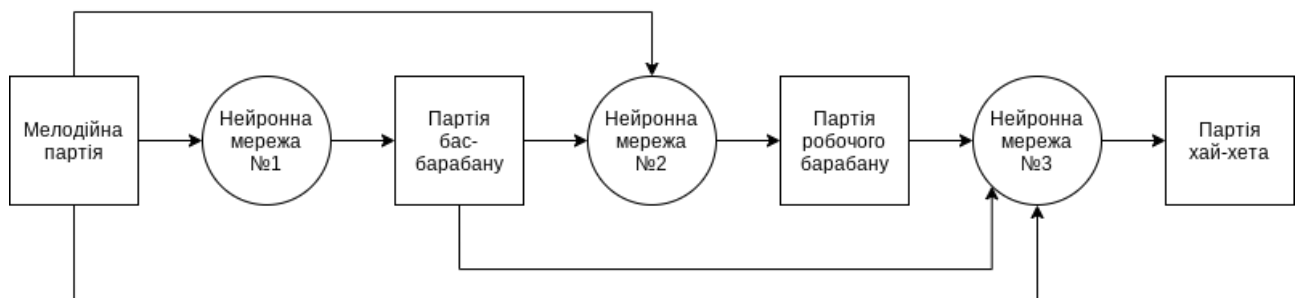


Рисунок 3.4. Схема роботи нейронних мереж для підбору ударної партії

1) Нейронна мережа для підбору партії бас-барабану.

Перша нейронна мережа складається з трьох шарів: вхідного, одного прихованого та вихідного. Вхідний шар містить 16 нейронів (по одному нейрону на кожну долю такту). Вихідний шар містить також 16 нейронів (по одному нейрону на кожну долю такту). Кількість нейронів прихованого шару обирається за принципом геометричної піраміди:

$$k = \sqrt{m \cdot n}, \quad (3.1)$$

де k — кількість нейронів прихованого шару, m — кількість нейронів вхідного шару, n — кількість нейронів вихідного шару.

Таким чином, в прихованому шарі першої нейронної мережі знаходиться 16 нейронів.

Нейронна мережа для підбору партії бас-барабану зображена на рис. 3.5.

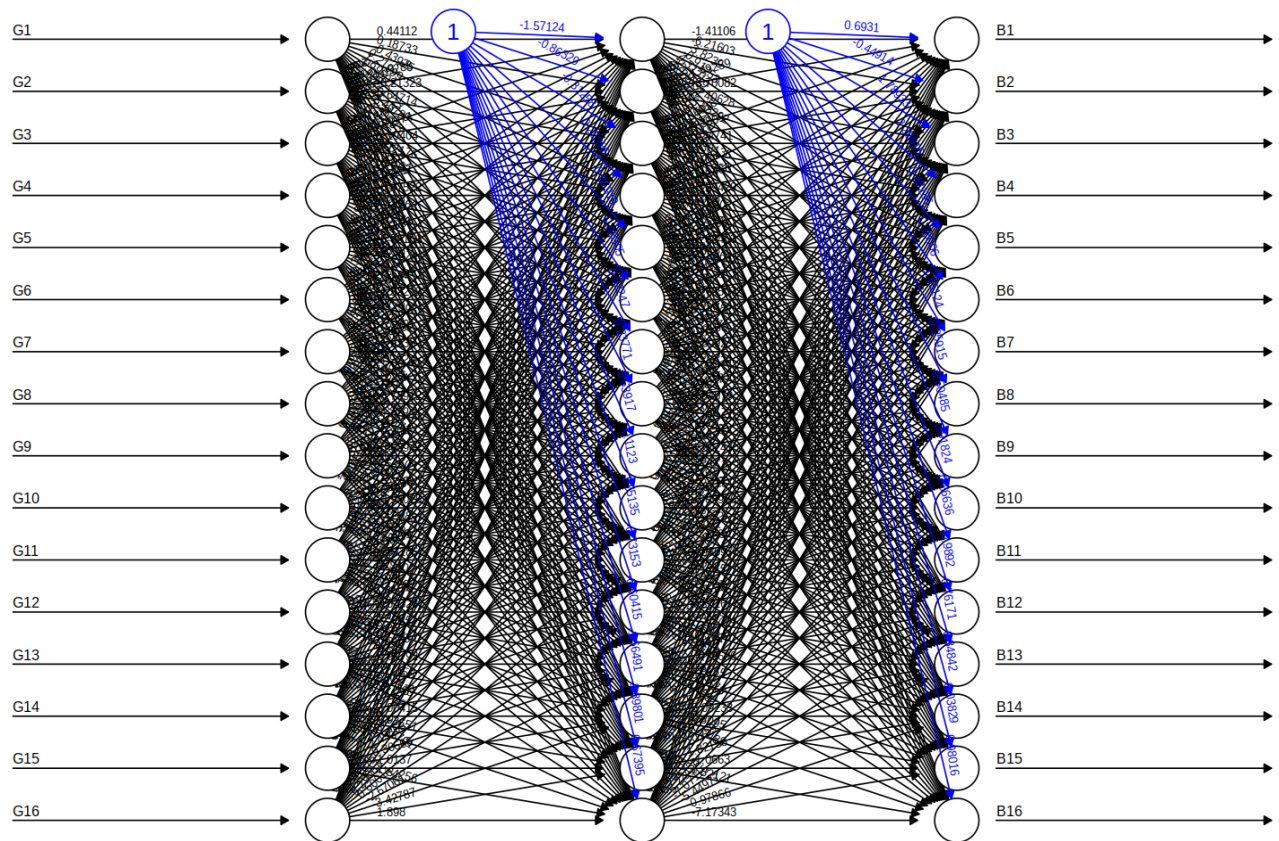


Рисунок 3.5. Нейронна мережа для підбору партії бас-барабану

2) Нейронна мережа для підбору партії робочого барабану.

Друга нейронна мережа аналогічна першій і також складається з трьох шарів: вхідного, одного прихованого та вихідного. Вхідний шар містить 32 нейрони (по 16 нейронів на мелодійну партію та партію бас-барабану).

Вихідний шар містить 16 нейронів (по одному нейрону на кожну долю такту партії робочого барабану). Кількість нейронів прихованого шару обирається за принципом геометричної піраміди: $k = \sqrt{m \cdot n} = \sqrt{32 \cdot 16} \approx 22.627 \approx 23$.

Нейронна мережа для підбору партії робочого барабану зображена на рис. 3.6.

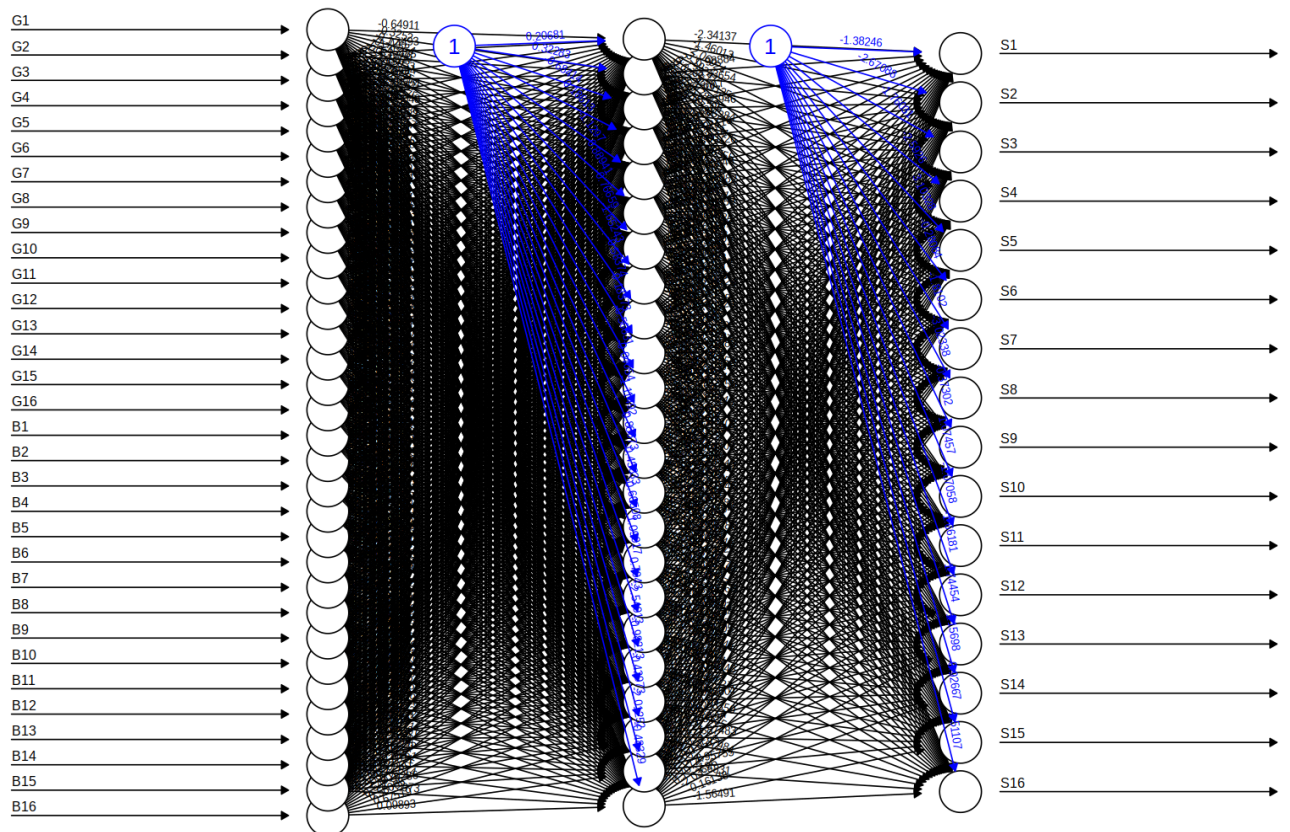


Рисунок 3.6. Нейронна мережа для підбору партії робочого барабану

3) Нейронна мережа для підбору партії хай-хета.

Третя нейронна мережа аналогічна першим двом і також складається з трьох шарів: вхідного, одного прихованого та вихідного. Вхідний шар містить 48 нейронів (по 16 нейронів на мелодійну партію, партію бас-барабану та партію робочого барабану). Вихідний шар містить 16 нейронів (по одному нейрону на кожну долю такту партії хай-хета). Кількість нейронів прихованого шару: $k = \sqrt{m \cdot n} = \sqrt{48 \cdot 16} \approx 27.713 \approx 28$.

Нейронна мережа для підбору партії хай-хета зображена на рис. 3.7.

Усі три нейронні мережі написані мовою програмування R, навчаються «з учителем» за допомогою методу зворотного поширення помилки.

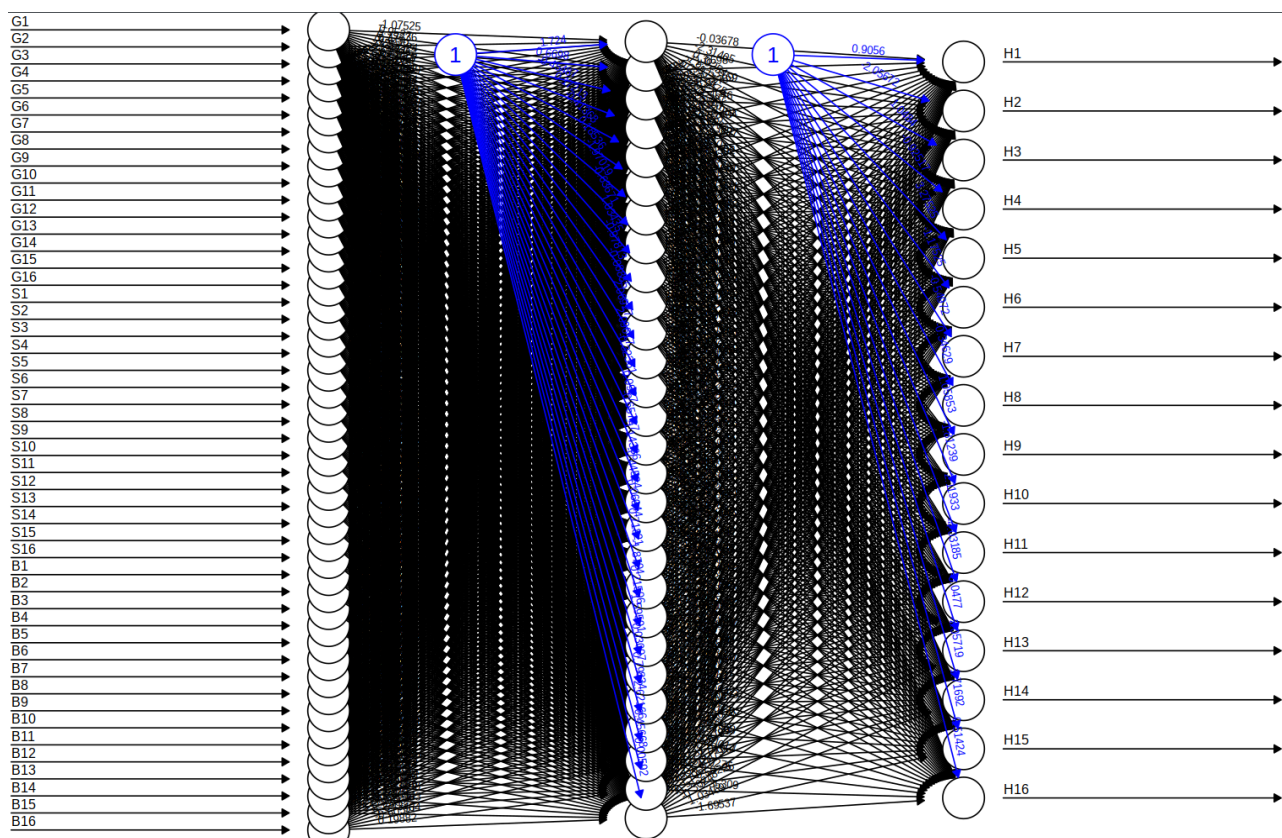


Рисунок 3.7. Нейронна мережа для підбору партії хай-хета

3.3. Конвертація даних між числовим форматом та MIDI

3.3.1. Опис пакета javax.sound.midi

Для роботи зі звуком та MIDI-форматом було обрано мову програмування Java та пакет javax.sound.midi. Цей пакет надає інтерфейси та класи для вводу/виводу даних MIDI, можливості створення послідовностей (sequence) та синтезу даних MIDI (Musical Instrument Digital Interface). За допомогою пакета javax.sound.midi можна зчитувати файли MIDI, програвати послідовності, записувати нові послідовності, відслідковувати MIDI-події тощо.

Основні класи та методи пакету наведені в табл. 3.1.

Таблиця 3.1. Основні класи та методи пакету javax.sound.midi

Назва	Опис
MidiSystem	Клас, що надає доступ до таких MIDI-ресурсів як секвенсори, синтезатори, порти введення/виведення. Усі методи цього класу є статичними, екземпляр класу не створюється.
getSequencer()	Метод класу MidiSystem, що повертає екземпляр секвенсора, під'єднаного до синтезатора.
sequencer.open()	Відкриває секвенсор для доступу до системних ресурсів.
sequencer.setSequence (Sequence sequence)	Встановлює поточну послідовність, з якою працюватиме секвенсор.
Sequencer.setTempoInBPM (float bpm)	Встановлює темп у бітах на хвилину.
Sequencer.start()	Починає програвання даних MIDI з поточно завантаженої послідовності.
Sequence	Клас, що представляє структуру даних, яка містить інформацію про трек або треки та часові характеристики.
Sequeunce.createTrack()	Створює пустий трек.
Track	Клас, що представляє трек і містить дані про MIDI-події у хронологічному порядку.
Track.add(MidiEvent event)	Додає нову подію до треку.
MidiEvent(MidiMessage message, long tick)	Об'єкт MIDI-події, що містить MIDI-повідомлення з поміткою про час.
ShortMessage()	Клас, що наслідує MidiMessage. Вміщає максимум 2 байти даних.
ShortMessage.setMessage(int command, int channel, int data1, int data2)	Встановлює коротке повідомлення з максимум двома байтами даних (data1 і data2).

3.3.2. Конвертація MIDI-файлу в числовий формат

Для конвертації мелодії, введеної користувачем у MIDI-форматі, використовуємо пакет `javax.sound.midi`.

Конвертація відбувається за наступним алгоритмом:

1. Вибір мелодійного треку за допомогою метода `getTrack()`.
2. Визначення кількості тактів у треці за формулою:

$$n = \frac{N_t - 1}{480}, \quad (3.2)$$

де n — кількість тактів, N_t — кількість тіків у доріжці. У стандартному MIDI-файлі один такт містить 480 тіків. Останній тік доріжки позначає її кінець, тому він не враховується.

3. Виконання для кожного такту наступних кроків:

3.1. Створення одномірного масиву *accents* на 16 елементів. Поділ такту на 16 долей. Виконання для кожної долі наступних кроків:

3.1.1. Пошук усіх подій типу Note On, що відбулися на тік даної долі.

3.1.2. Підрахування середнього арифметичного параметрів *velocity* кожної події, знайденої на попередньому кроці, за формулою:

$$v_{\text{сеп}} = \frac{\sum_{i=1}^m v_i}{m}, \quad (3.3)$$

де $v_{\text{сеп}}$ — середня гучність звучання мелодії на даній долі, m — кількість подій Note On, v_i — гучність звучання i -ої ноти на даній долі.

3.1.3. Якщо $v_i \leq 20$, записати в масив *accents* 0, якщо $20 < v_i < 100$, записати в масив 1, якщо $v_i \geq 100$, записати в масив 2.

3.1.4. Перехід на наступну долю.

3.2. Запис масиву *accents* до файлу *music.csv*.

3.3. Якщо такт не був останнім, перехід до наступного. Інакше кінець алгоритму.

3.3.2. Конвертація числових результатів в MIDI-формат

Результати роботи нейронної мережі записуються у файл *result.csv*. Для конвертації цих числових даних у MIDI-формат використаємо пакет *javax.sound.midi*.

Конвертація відбувається за наступним алгоритмом:

1. Зчитування файлу *result.csv*, запис числових барабаних партій в масив тактів.

2. Створення нового треку.

3. Виконання для *i*-го такту наступних кроків:

3.1. Запис у трек партії бас-барабану. Занесення значень з 17 по 32 *i*-го рядка файлу *result.csv* в масив *bass_beats*. Для кожного *j*-го елемента масиву:

3.1.1. Якщо *bass_beats[j] = 2*, додати MIDI-подію Note On з параметрами *key = 35*, *velocity = 127* на тік $480 * i + j$ та MIDI-подію Note Off з параметрами *key = 35*, *velocity = 127* на тік $480 * i + j + 15$.

3.1.2. Якщо *bass_beats[j] = 1*, додати MIDI-подію Note On з параметрами *key = 35*, *velocity = 80* на тік $480 * i + j$ та MIDI-подію Note Off з параметрами *key = 35*, *velocity = 80* на тік $480 * i + j + 15$.

3.1.3. Перейти до наступного елемента масиву.

3.2. Запис у трек партії робочого барабану. Занесення значень з 33 по 48 *i*-го рядка файлу *result.csv* в масив *snare_beats*. Для кожного *j*-го елемента масиву:

					ІТ-62.08.1081.01 ПЗ	Арк.
						40
Змн.	Арк.	№ докум.	Підпис	Дат.		

3.2.1. Якщо $snare_beats[j] = 2$, додати MIDI-подію Note On з параметрами $key = 38$, $velocity = 127$ на тік $480 * i + j$ та MIDI-подію Note Off з параметрами $key = 38$, $velocity = 127$ на тік $480 * i + j + 15$.

3.2.2. Якщо $snare_beats[j] = 1$, додати MIDI-подію Note On з параметрами $key = 38$, $velocity = 80$ на тік $480 * i + j$ та MIDI-подію Note Off з параметрами $key = 38$, $velocity = 80$ на тік $480 * i + j + 15$.

3.2.3. Перейти до наступного елемента масиву.

3.3. Запис у трек партії хай-хета. Занесення значень з 49 по 64 i -го рядка файлу result.csv в масив $hihat_beats$. Для кожного j -го елемента масиву:

3.3.1. Якщо $hihat_beats[j] = 2$, додати MIDI-подію Note On з параметрами $key = 46$, $velocity = 127$ на тік $480 * i + j$ та MIDI-подію Note Off з параметрами $key = 46$, $velocity = 127$ на тік $480 * i + j + 15$.

3.3.2. Якщо $hihat_beats[j] = 1$, додати MIDI-подію Note On з параметрами $key = 46$, $velocity = 80$ на тік $480 * i + j$ та MIDI-подію Note Off з параметрами $key = 46$, $velocity = 80$ на тік $480 * i + j + 15$.

3.3.3. Перейти до наступного елемента масиву.

3.4. Перейти до наступного такту.

4. Додавання створеного треку з ударними до раніше зчитаного з MIDI-файлу користувача об'єкту Sequence.

5. Збереження результуючої композиції у файл result.mid.

3.4. Загальна структура проекту

Проект складається з двох частин:

– нейронна мережа, описана в розділі 3.2;

					ІТ-62.08.1081.01 ПЗ	Арк. 41
Змн.	Арк.	№ докум.	Підпис	Дат.		

– програмна частина, написана мовою програмування Java, яка обробляє введені користувачем музичні дані.

Програмна частина проєкту працює за наступним алгоритмом:

Крок 1. Користувач вводить музичні дані у форматі MIDI. Програма зчитує файл з даними.

Крок 2. За допомогою пакета `javax.sound.midi` музичні дані користувача конвертуються у формат, описаний у розділі 3.1. Результат записується у форматі CSV у файл `music.csv`.

Крок 3. Програма викликає скрипт `neural_beats.r`, написаний мовою R, що містить нейронну мережу.

Крок 4. Скрипт `neural_beats.r` зчитує файл `music.csv`, дані з нього подаються на вхід нейронної мережі.

Крок 5. Нейронна мережа аналізує вхідні дані та видає результат у форматі, описаному в розділі 3.1.

Крок 6. Результат роботи нейронної мережі записується в файл `result.csv`.

Крок 7. Програма зчитує файл `result.csv` та переводить числові дані у формат MIDI за допомогою пакета `javax.sound.midi`.

Крок 8. Результат роботи програми надається користувачеві. Після прослуховування, якщо користувач не задоволений результатом, повторюються кроки 3-8. Якщо результат задовільний, програма завершує свою роботу.

Загальний алгоритм роботи системи наведений на блок-схемі (рис. 3.8).

					ІТ-62.08.1081.01 ПЗ	Арк.
						42
Змн.	Арк.	№ докум.	Підпис	Дат.		

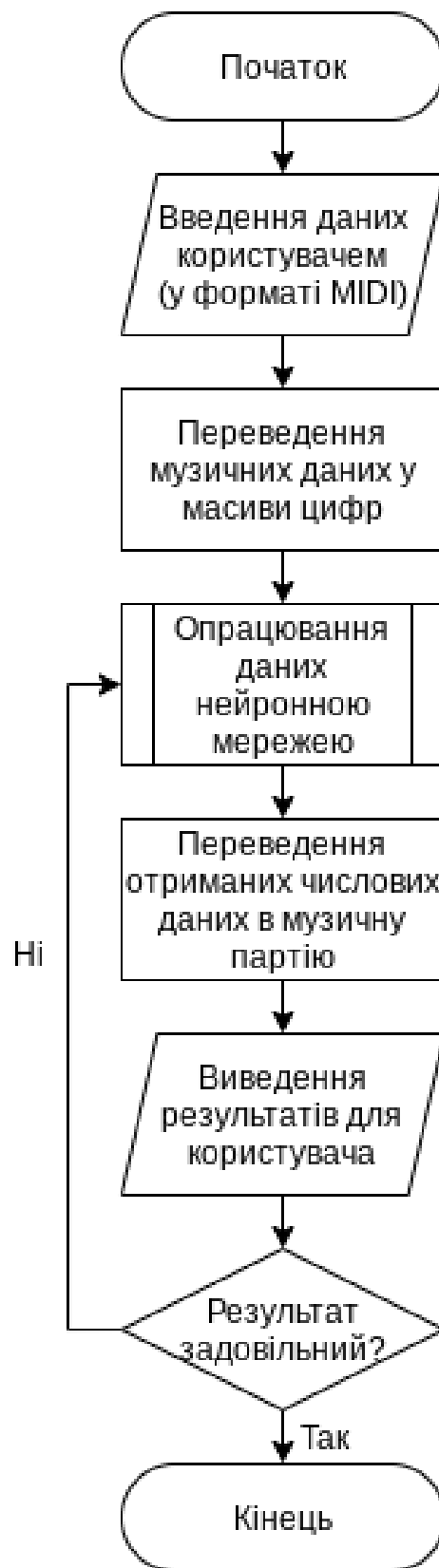


Рисунок 3.8. Блок-схема алгоритму роботи програми

ВИСНОВКИ ДО РОЗДІЛУ

У третьому розділі роботи було спроектовано та розроблено інтелектуальну систему підбору барабанної партії для аранжування композицій.

Було розроблено алгоритм кодування музичних партій та їх акцентів у числові дані для подальшого використання для навчання нейронної мережі. Музичні композиції кодуються не цілком, а потактно, що дає змогу зменшити розміри початкового датасету та спростити роботу нейронної мережі.

Результуюча програма складається з двох частин:

- нейронна мережа, що аналізує існуючі пісні та підбирає партії до нових;
- власне програма, що конвертує MIDI-формат у числовий та навпаки і взаємодіє з користувачем.

Нейронна мережа реалізує регресійну модель, навчається «з учителем» за допомогою методу зворотного поширення помилки. Початковий датасет для навчання складається з двадцяти пар мелодійна партія — ударна партія.

Були розроблені наступні алгоритми конвертації даних:

- алгоритм конвертації файлів MIDI, що містять мелодійну партію, в числовий формат, що подається на вхід нейронної мережі;
- алгоритм конвертації числових результатів роботи нейронної мережі в файли MIDI, тобто власне музику.

Алгоритми запрограмовані мовою Java з використанням пакета `javax.sound.midi`.

					ІТ-62.08.1081.01 ПЗ	Арк. 44
Змн.	Арк.	№ докум.	Підпис	Дат.		

РОЗДІЛ 4. АНАЛІЗ РЕЗУЛЬТАТІВ РОБОТИ СИСТЕМИ ПІДБОРУ УДАРНОЇ ПАРТІЇ ДЛЯ АРАНЖУВАННЯ КОМПОЗИЦІЙ

4.1. Дослідження якості роботи нейронної мережі

Процес навчання нейронної мережі полягає в тому, аби підлаштувати її внутрішні параметри під конкретну задачу.

Початковий набір даних (датасет) для навчання нейронної мережі поділяють на дві підмножини. Більша підмножина (80% від початкового датасету) є навчальною множиною, менша підмножина (20% від початкового датасету) використовується для подальшого тестування та обчислення точності прогнозування нейронної мережі.

Алгоритм навчання нейронної мережі є ітеративним, а його кроки називають епохами або циклами. За одну епоху через нейронну мережу пропускаються усі приклади з навчальної множини. Процес навчання здійснюється на навчальній вибірці. Навчальна вибірка включає в себе вхідні значення і відповідні їм вихідні значення набору даних. В ході навчання нейронна мережа знаходить якісь залежності вихідних значень від вхідних.

Помилка навчання для побудованої нейронної мережі обчислюється шляхом порівняння вихідних і цільових (бажаних) значень. З отриманих різниць формується функція помилок. Функція помилок — це цільова функція, що вимагає мінімізації в процесі керованого навчання нейронної мережі. За допомогою функції помилок можна оцінити якість роботи нейронної мережі під час навчання [13].

В даній роботі було створено три нейронні мережі для підбору партій трьох елементів ударної установки. Було проведено ряд досліджень, щоб виявити оптимальну кількість нейронів у прихованих шарах, кількість епох навчання та розмір навчальної вибірки. Результати досліджень представлені у таблицях 4.1 — 4.3.

					ІТ-62.08.1081.01 ПЗ	Арк. 45
Змн.	Арк.	№ докум.	Підпис	Дат.		

Таблиця 4.1. Дослідження впливу параметрів нейронної мережі для підбору партії бас-барабану на точність прогнозування

К-сть нейронів у прихованому шарі	Розмір вибірки для навчання	Кількість епох навчання	Час навчання, с	Точність прогнозування, %
8	10	1	0,003	17,532
		10	0,247	32,342
		100	2,179	31,265
	15	1	0,019	22,332
		10	0,257	63,156
		100	2,351	65,094
	20	1	0,029	40,223
		10	0,218	76,232
		100	2,093	77,863
16	10	1	0,016	19,423
		10	0,141	75,343
		100	1,144	78,422
	15	1	0,015	28,984
		10	0,143	94,822
		100	1,145	96,145
	20	1	0,014	51,722
		10	0,127	95,774
		100	1,166	98,115
32	10	1	0,012	33,634
		10	0,137	72,745
		100	1,367	71,534
	15	1	0,016	41,537
		10	0,141	95,431
		100	1,366	97,341
	20	1	0,014	60,343
		10	0,148	96,324
		100	1,337	98,998

Таблиця 4.2. Дослідження впливу параметрів нейронної мережі для підбору партії робочого барабану на точність прогнозування

К-сть нейронів у прихованому шарі	Розмір вибірки для навчання	Кількість епох навчання	Час навчання, с	Точність прогнозування, %
11	10	1	0,009	28,532
		10	0,104	60,348
		100	0,932	59,168
	15	1	0,012	34,532
		10	0,114	78,154
		100	0,896	80,032
	20	1	0,008	41,223
		10	0,104	88,232
		100	0,932	89,863
23	10	1	0,008	30,421
		10	0,081	80,322
		100	0,638	81,250
	15	1	0,007	42,342
		10	0,069	95,822
		100	0,583	97,725
	20	1	0,008	52,572
		10	0,066	97,734
		100	0,604	98,915
46	10	1	0,014	33,634
		10	0,078	74,745
		100	0,731	78,365
	15	1	0,009	46,153
		10	0,084	91,446
		100	0,763	92,073
	20	1	0,010	60,382
		10	0,087	94,294
		100	0,731	95,204

Таблиця 4.3. Дослідження впливу параметрів нейронної мережі для підбору партії хай-хета на точність прогнозування

К-сть нейронів у прихованому шарі	Розмір вибірки для навчання	Кількість епох навчання	Час навчання, с	Точність прогнозування, %
14	10	1	0,014	21,512
		10	0,112	39,542
		100	1,011	58,265
	15	1	0,012	27,332
		10	0,135	70,156
		100	1,014	69,993
	20	1	0,015	47,323
		10	0,110	81,232
		100	1,031	80,863
28	10	1	0,012	50,423
		10	0,114	77,321
		100	0,948	79,432
	15	1	0,013	31,984
		10	0,110	94,372
		100	0,954	98,141
	20	1	0,008	52,723
		10	0,098	96,774
		100	0,923	99,115
56	10	1	0,012	35,634
		10	0,144	77,745
		100	1,276	80,532
	15	1	0,020	43,536
		10	0,137	97,432
		100	1,313	98,342
	20	1	0,014	59,343
		10	0,141	97,324
		100	1,267	99,121

Найкращі результати у таблицях виділені жирним шрифтом.

Очевидно, що чим більшою є навчальна вибірка, тим кращі результати роботи мереж. Тому остаточне навчання мереж проводилося на вибірці розміром у 20 пар мелодійна партія — ударна партія.

Найкращі результати навчання показали нейронні мережі з найбільшою кількістю нейронів прихованого шару. Однак часові затрати при цьому були в рази більші, ніж при використанні кількості нейронів, підрахованої в розділі 3.2. Причому точність роботи мереж з більшою кількістю нейронів відрізнялась від точності роботи мереж з меншою кількістю нейронів в середньому на 0,982%. Тому для економії часових затрат остаточне навчання мереж для підбору партій бас-барабану, робочого барабану та хай-хета проводилося з кількістю нейронів прихованих шарів 16, 23 та 28 відповідно.

Часові затрати при 10 та 100 епохах навчання відрізнялися незначно, а точність роботи при 100 епохах виявилася більшою. Тому остаточне навчання нейронних мереж проводилося у 100 епох.

4.2. Результати опитування слухачів

Для оцінки якості системи було проведено опитування слухачів. Загалом було опитано 74 людини, серед яких 17 людей — музиканти, що займаються музикою професійно або у якості хоббі. Респондентам було запропоновано два види оцінювання.

Опитування проводилося за допомогою Google Forms.

У першому тесті слухачам було запропоновано послухати три музичні уривки, до яких розробленою в цій роботі системою було згенеровано ударні партії. Респонденти мали оцінити якість звучання за 5-бальною шкалою. Результати першого опитування наведені на рис. 4.1.

					ІТ-62.08.1081.01 ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дат.		49

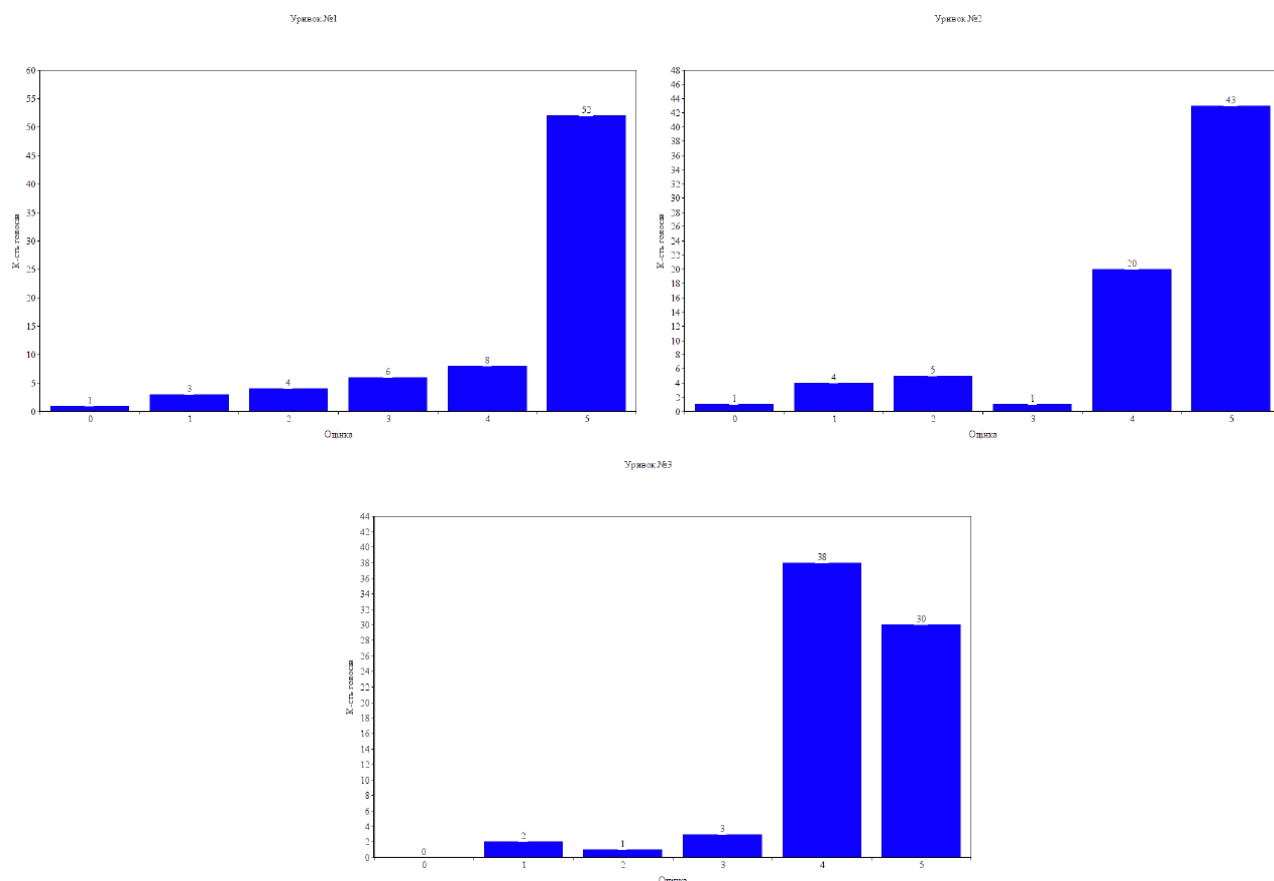


Рисунок 4.1. Результати першого опитування слухачів

У другому тесті слухачам було запропоновано послухати три пари музичних уривків. Мелодійна партія в кожній парі була одна й та сама, а ударна партія в одному з уривків була підібрана нейронною мережею, в другому — було обрано шаблонний ритм, який найкраще пасував до мелодійної партії, із запропонованих ударних партій програми WalkBand. Загальні результати опитування наведені на рис. 4.2. Детальні результати винесені в додаток Г.

За результатами першого оцінювання видно, що слухачам сподобалися музичні уривки, в яких ритм був згенерований нейронною мережею.

За результатами другого оцінювання можна сказати, що ритми, згенеровані нейронною мережею, подобаються слухачам більше, ніж шаблонні ритми. Однак різниця у відповідях невелика, з чого можна зробити висновок про те, що ритми, згенеровані нейронною мережею, настільки якісні, що



Рисунок 4.2. Результати другого опитування слухачів

слухачі не помічають різниці між музикою, написаною комп'ютером, та музикою, створеною людиною.

Цікаво також зазначити, що музиканти частіше обирали ритми, згенеровані нейронною мережею, ніж слухачі, що не займаються музикою. Це свідчить про те, що система працює за принципами контрапункту, і хоча для людей без музичної освіти це не помітно, професійні музиканти на таку дрібницю звертають увагу.

ВИСНОВКИ ДО РОЗДІЛУ

В четвертому розділі роботи було проведено декілька експериментів для перевірки якості роботи системи.

Експерименти з кількістю нейронів прихованого шару показали, що оптимальною кількістю нейронів для нейронної мережі, що підбирає партію бас-барабана, є 16 нейронів, для нейронної мережі, що підбирає партію

робочого барабана, — 23 нейрони, для нейронної мережі, що підбирає партію хай-хета, — 28 нейронів.

Оптимальною кількістю епох навчання є 100 епох. Найкращі результати дають нейронні мережі, навчені на вибірці розміром у 20 пар мелодійна партія — ударна партія.

Проведені експерименти дали змогу оптимально налаштувати параметри нейронних мереж та забезпечити точність передбачення 98,978%.

Також було проведено декілька опитувань слухачів як серед музикантів, так і серед людей, що не мають музичної освіти. Респондентам було запропоновано прослухати музичні уривки з барабанними партіями, які підбрала розроблена система, оцінити їх та порівняти з уривками із шаблонними ударними партіями. Результати опитувань показали, що робота системи є високої якості.

Таким чином, створена система підбору ударних партій для аранжування композицій виконує усі передбачені функції з найкращими результатами. Крім того, система має потенціал для подальшого розвитку, а саме: в майбутньому система може бути доповнена можливістю підбору не лише ударних партій, але й побічних мелодійних партій.

					ІТ-62.08.1081.01 ПЗ	Арк.
						52
Змн.	Арк.	№ докум.	Підпис	Дат.		

ВИСНОВКИ

У даній роботі було досліджено методи роботи з музикою у системах штучного інтелекту. Було проаналізовано існуючі рішення інтелектуальних систем для роботи з музичними творами та виявлено їхні недоліки. Було досліджено можливості створення інтелектуальної системи для підбору ударної партії до заданої музичної композиції.

Результатом даної роботи є спроектована та реалізована система підбору ударної партії для аранжування композицій. Система використовує спроектовану та навчену нейронну мережу для підбору ударних партій до введених користувачем мелодійних партій. Система працює за принципами контрапункту та аналізує в основному акценти в музиці, тому є універсальною для підбору ударної партії у будь-якому музичному жанрі.

Система працює з форматом MIDI, що є універсальним форматом для роботи з музичними файлами.

Було досліджено якість роботи системи шляхом проведення експериментів та опитування слухачів як серед людей без музичної освіти, так і серед професійних музикантів. Система показала відмінний результат роботи. Штучний інтелект генерує різноманітні ритми, які вдало поєднуються із заданими мелодійними партіями.

Розроблена програма може стати у нагоді композиторам-початківцям, які прагнуть вдало аранжувати власну композицію, але не вивчали основ аранжування та принципів контрапункту. Крім того дана система може бути використана музичними гуртами для аранжування власних творів і просто зацікавленими людьми.

Розроблена система має потенціал для подальшого розвитку. В майбутньому планується додати можливість аранжування не лише додаванням ударної партії, але й генерацією побічних мелодійних партій.

					ІТ-62.08.1081.01 ПЗ	Арк. 53
Змн.	Арк.	№ докум.	Підпис	Дат.		

ПЕРЕЛІК ПОСИЛАНЬ

- 1 *Sandred, Örjan & Laurson, Mikael & Kuuskankare, Mika.* (2009). Revisiting the Illiac Suite — A rule-based approach to stochastic processes. *Sonic Ideas/Ideas Sonicas*. 2. 42-46.
- 2 *Бондаренко, Алексей.* Искусственный интеллект и музыка. Отберет ли компьютер работу у композиторов? [Электронный ресурс]. — 2018. — Режим доступа до ресурсу: https://karabas.live/ai_music/
- 3 *Chien-Hung, Liu & Ting, Chuan-Kang.* (2012). Polyphonic accompaniment using genetic algorithm with music theory. 2012 IEEE Congress on Evolutionary Computation, CEC 2012. 1-7. 10.1109/CEC.2012.6252869.
- 4 Алгоритмическая композиция. Естественные алгоритмы: роевой интеллект [Электронный ресурс]. — 2013. — Режим доступа до ресурсу: https://algorithmiccomposition.ru/article_entry_swarming.html
- 5 *Toussaint, Godfried.* (2005). The Euclidean Algorithm Generates Traditional Musical Rhythms. 47-56.
- 6 Рекуррентна нейронна мережа [Электронный ресурс]. — Режим доступа до ресурсу: https://uk.wikipedia.org/wiki/Рекуррентна_нейронна_мережа
- 7 *Nikolov, Stanislav.* NeuralBeats: Generative Techno with Recurrent Neural Networks [Электронный ресурс]. — 2016. — Режим доступа до ресурсу: <https://medium.com/@snikolov/neuralbeats-generative-techno-with-recurrent-neural-networks-3824d7ba7972>
- 8 *Huang, Tsunghao.* Neural Networks Generated Lamb of God Drum Tracks [Электронный ресурс]. — 2019. — Режим доступа до ресурсу: <https://towardsdatascience.com/neural-networks-generated-lamb-of-god-drum-tracks-45d3a235e13a>
- 9 Простой контрапункт. Сложный контрапункт [Электронный ресурс]. — Режим доступа до ресурсу:

					IT-62.08.1081.01 ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дат.		54

<https://ludustonalis.jimdofree.com/полифония/контрапункт-простой-и-сложный/>

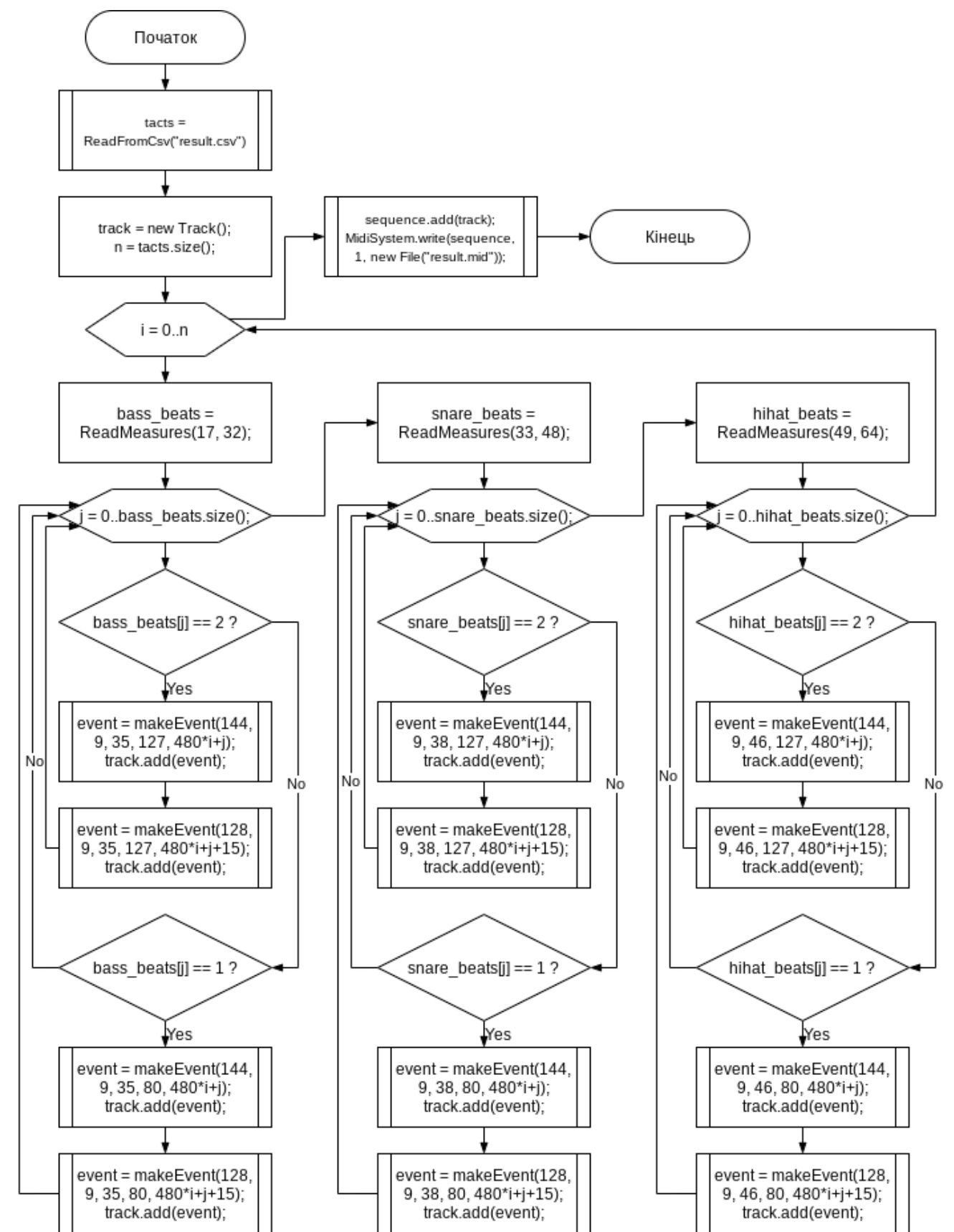
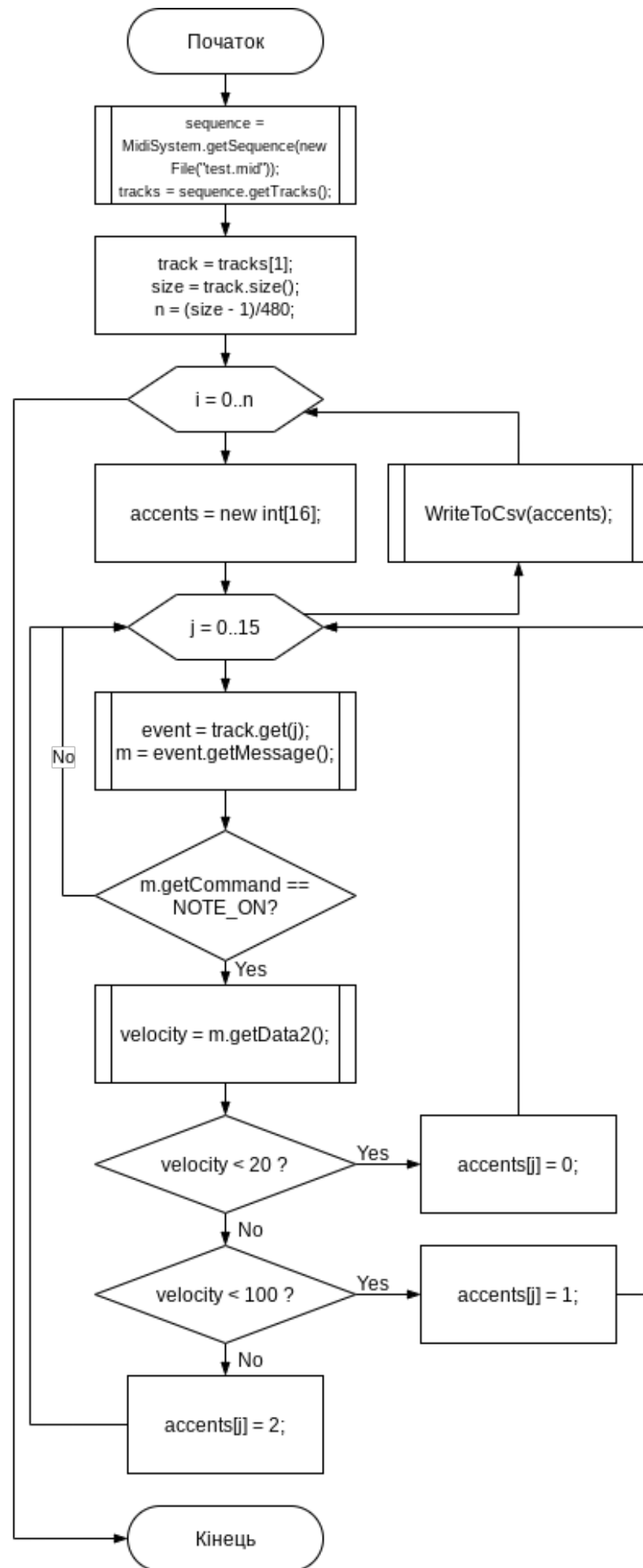
10 Контрапункт [Электронный ресурс]. — Режим доступа до ресурсу: <https://ru.wikipedia.org/wiki/Контрапункт>

11 MIDI [Электронный ресурс]. — Режим доступа до ресурсу: <https://uk.wikipedia.org/wiki/MIDI>

12 Neural Networks for Regression—Overkill or Opportunity? [Электронный ресурс] — Режим доступа до ресурсу: <https://missinglink.ai/guides/neural-network-concepts/neural-networks-regression-part-1-overkill-opportunity/>

13 Обучение нейронных сетей [Электронный ресурс]. — Режим доступа до ресурсу: <https://www.intuit.ru/studies/courses/6/6/lecture/178?page=4>

					ІТ-62.08.1081.01 ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дат.		55



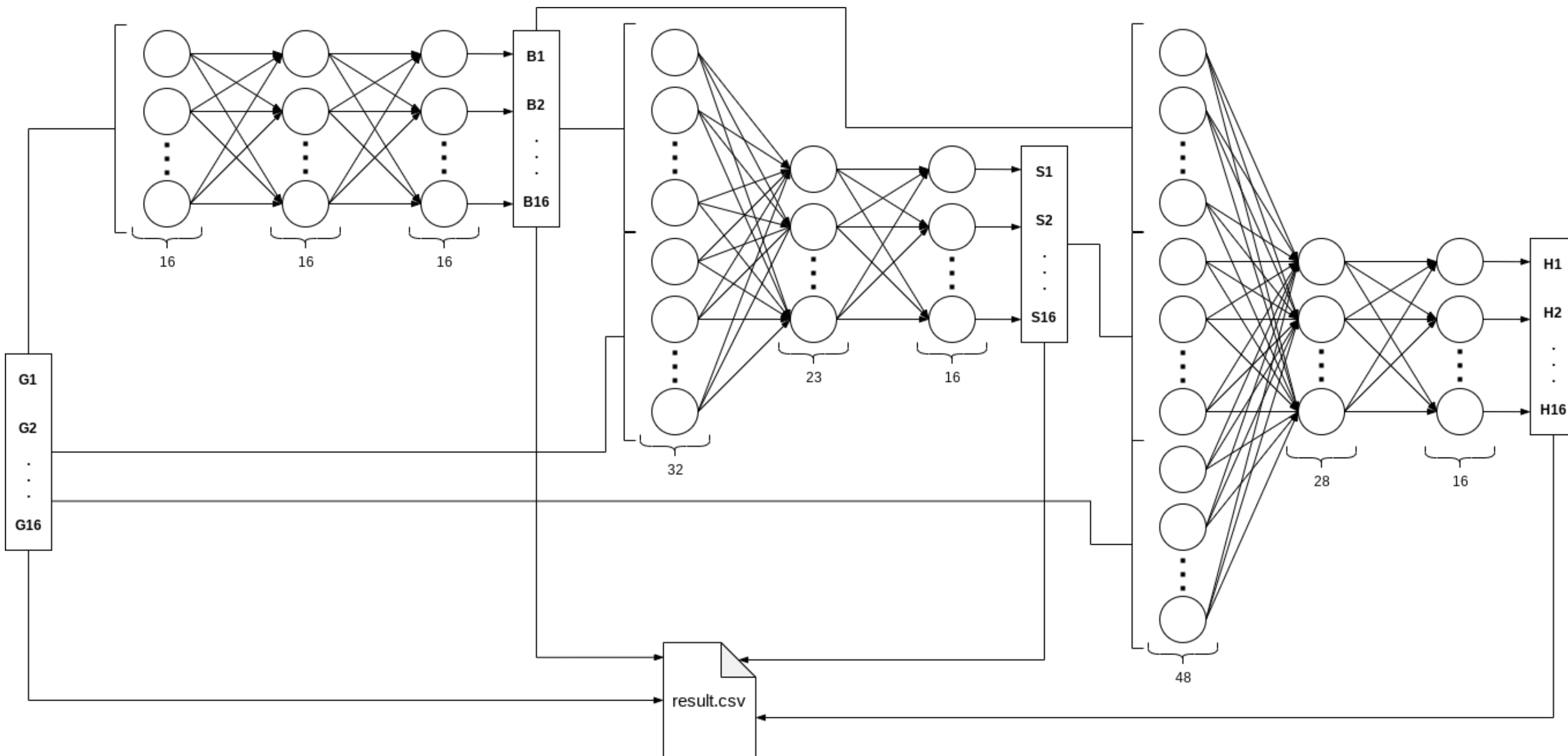
Блок-схема алгоритмів
конвертації файлів між
числовим та MIDI-
форматом

Кафедра
Технічної кібернетики

Літ.	Маса	Мірило
Лист 1	Листів 1	

Група IT-62

Підпис і дата	Інв. № дубл.	Взам. інв. №	Підпис і дата	Інв. № ориг.



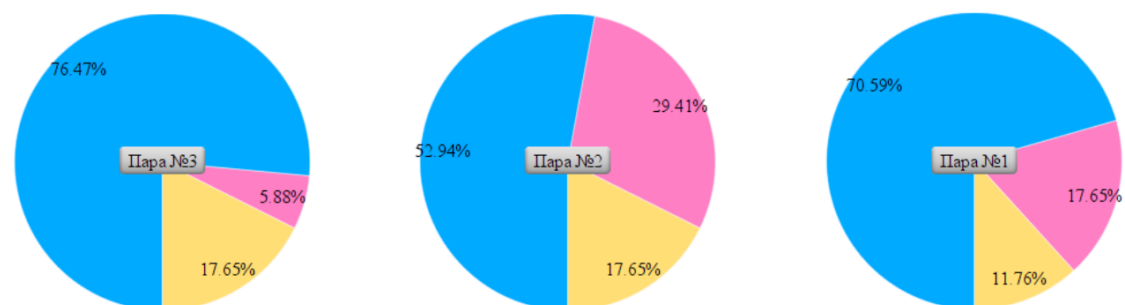
Інв. № дубл.	Підпис і дата
Взам. інв. №	
Підпис і дата	
Інв. № ориг.	

					ІТ-62.08.1081.04 ВЗ				
					Загальна схема роботи нейронних мереж	Літ.		Маса	Мірило
Зм.	Лист	№ докум	Підпис	Дата					
Розроб.	Каширіна О.Ю.								
Перев.	Цьопа Н.В.								
					Кафедра Технічної кібернетики	Лист 1		Листів 1	
Н. контр.	Пасько В.П.					Група ІТ-62			
Затв.	Пархомей І.Р.								

Результати оцінки якості роботи програми

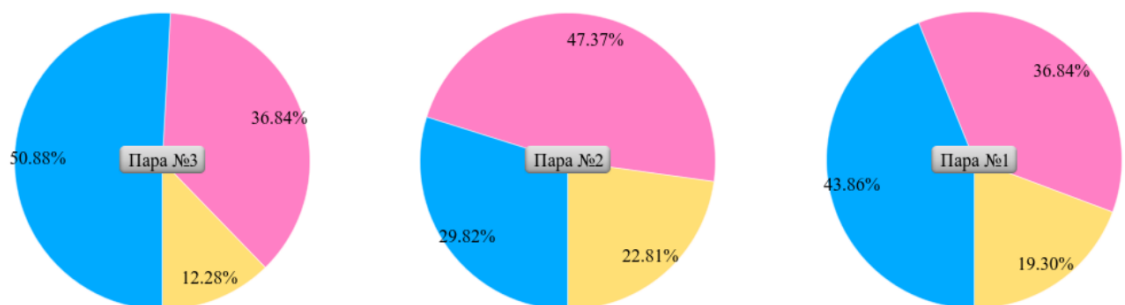
Вибір музикантів

■ Ритм, згенерований нейронною мережею ■ Шаблонний ритм ■ Не можу обрати



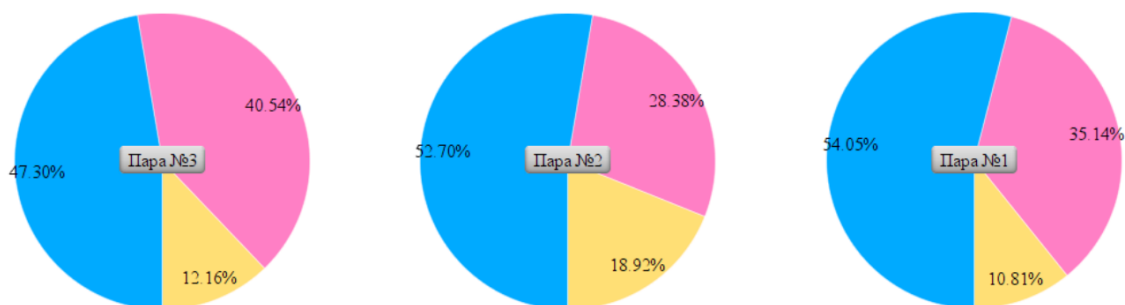
Вибір слухачів, що не займаються музикою

■ Ритм, згенерований нейронною мережею ■ Шаблонний ритм ■ Не можу обрати



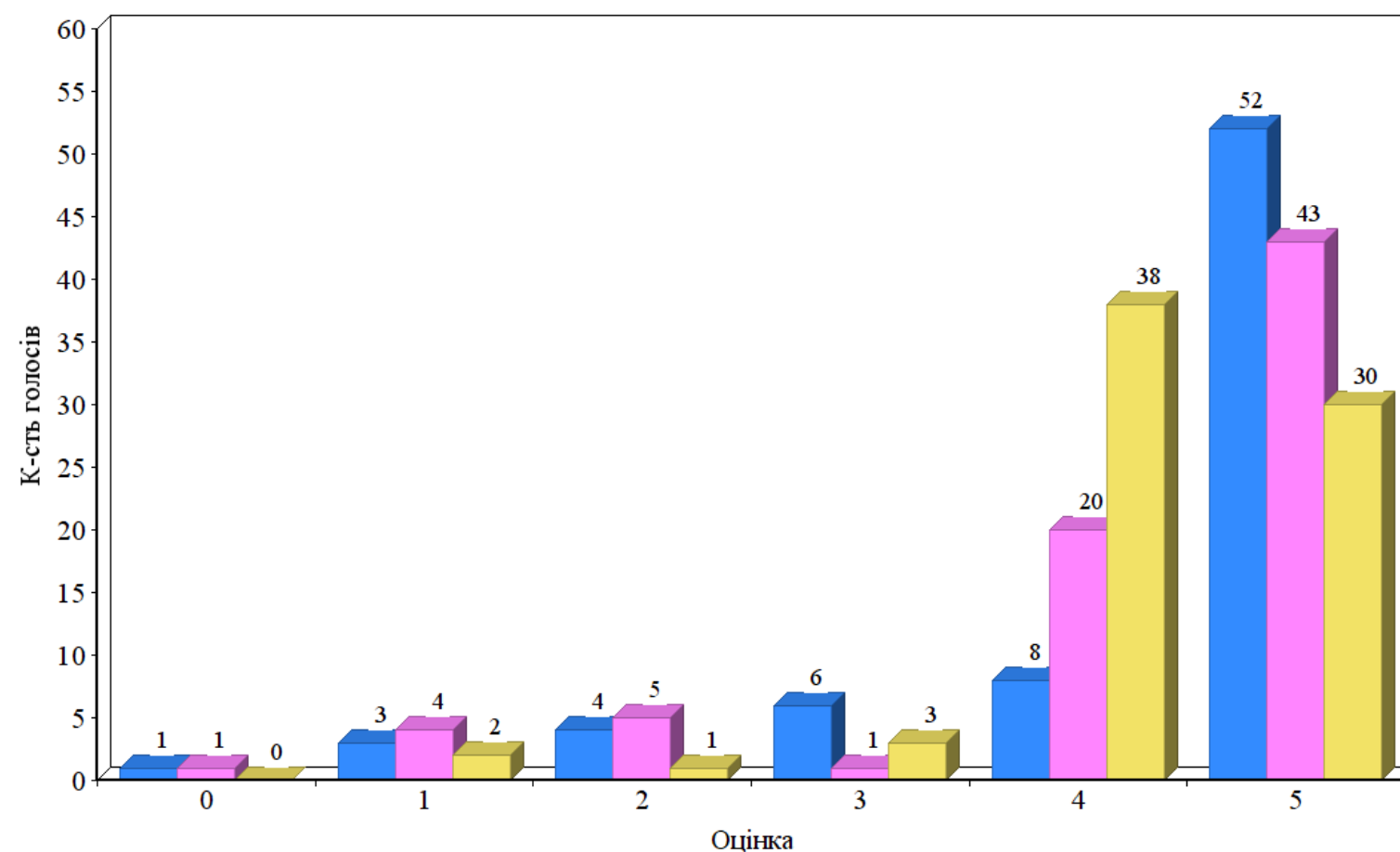
Загальні результати

■ Ритм, згенерований нейронною мережею ■ Шаблонний ритм ■ Не можу обрати



Оцінка звучання мелодій

■ Уривок №1 ■ Уривок №2 ■ Уривок №3



Демонстраційний плакат №1

до дипломного проєкту на тему

„Інтелектуальна система підбору ударної партії для аранжування композицій”

Розробила: студентка гр. ІТ-62 Каширіна О.Ю.

Прийняла: к.т.н., доцент Цьопа Н.В.

ДОДАТОК Г

```
require(neuralnet)
```

```
#функція переводу результатів роботи мережі в акценти
```

```
round <- function(n){  
  if(n<0.3) return (0)  
  else if(n >= 0.3 && n < 0.8) return (1)  
  else if(n >= 0.8) return (2)  
}
```

```
beats <- read.csv("beats.csv", header=TRUE)  
print(beats)
```

```
#навчання нейронної мережі для підбору партії бас-барабану
```

```
start_time <- Sys.time()  
for (i in 1:100)  
  base <-  
    neuralnet(B1+B2+B3+B4+B5+B6+B7+B8+B9+B10+B11+B12+B13+B14+B15+B16  
      ~G1+G2+G3+G4+G5+G6+G7+G8+G9+G10+G11+G12+G13+G14+G15+G16,  
      data=beats,  
      hidden=16,  
      act.fct = "logistic",  
      linear.output = FALSE)  
end_time <- Sys.time()  
end_time - start_time  
plot(base)
```

```
#навчання нейронної мережі для підбору партії робочого барабану
```

```
start_time <- Sys.time()  
for (i in 1:100)  
  snare <-  
    neuralnet(S1+S2+S3+S4+S5+S6+S7+S8+S9+S10+S11+S12+S13+S14+S15+S16
```

					IT-62.08.1081.06 12	Арк.
Змн.	Арк.	№ докум.	Підпис	Дат.		61

```

~G1+G2+G3+G4+G5+G6+G7+G8+G9+G10+G11+G12+G13+G14+G15+G16+
B1+B2+B3+B4+B5+B6+B7+B8+B9+B10+B11+B12+B13+B14+B15+B16,
data=beats,
hidden=23,
act.fct = "logistic",
linear.output = FALSE)
end_time <- Sys.time()
end_time - start_time
plot(snare)

#навчання нейронної мережі для підбору партії хай-хета
start_time <- Sys.time()
for (i in 1:100)
hihat <-
    neuralnet(H1+H2+H3+H4+H5+H6+H7+H8+H9+H10+H11+H12+H13+H14+H15+H16~.,
    data=beats, hidden=28,act.fct = "logistic",linear.output = FALSE)
end_time <- Sys.time()
end_time - start_time
plot (hihat)

#зчитування файлу з акцентами
test <- read.csv("music.csv", header=TRUE)
print(test)

#підбір партії бас-барабану
predict_base = compute(base,test)
predict_base$net.result
base_res <- predict_base$net.result
pred <- apply(base_res, 1:2, round)
pred

```

					IT-62.08.1081.06 12	Арк.
Змн.	Арк.	№ докум.	Підпис	Дат.		62

```

base_pred <-
  data.frame(B1=pred[,1],B2=pred[,2],B3=pred[,3],B4=pred[,4],B5=pred[,5]
    ,B6=pred[,6],B7=pred[,7],B8=pred[,8],B9=pred[,9],B10=pred[,10],B11=pre
    d[,11],B12=pred[,12],B13=pred[,13],B14=pred[,14],B15=pred[,15],B16=pre
    d[,16])
test2 <- cbind(test, base_pred)
test2

#підбір партії робочого барабану
predict_snare <- compute(snare,test2)
predict_snare$net.result
snare_res <- predict_snare$net.result
pred <- apply(snare_res, 1:2, round)
pred

snare_pred <-
  data.frame(S1=pred[,1],S2=pred[,2],S3=pred[,3],S4=pred[,4],S5=pred[,5]
    ,S6=pred[,6],S7=pred[,7],S8=pred[,8],S9=pred[,9],S10=pred[,10],S11=pre
    d[,11],S12=pred[,12],S13=pred[,13],S14=pred[,14],S15=pred[,15],S16=pre
    d[,16])
test3 <- cbind(test2, snare_pred)
test3

#підбір партії хай-хета
predict_hihat <- compute(hihat,test3)
predict_hihat$net.result
hihat_res <- predict_hihat$net.result
pred <- apply(hihat_res, 1:2, round)
pred

hihat_pred <-
  data.frame(H1=pred[,1],H2=pred[,2],H3=pred[,3],H4=pred[,4],H5=pred[,5]
    ,H6=pred[,6],H7=pred[,7],H8=pred[,8],H9=pred[,9],H10=pred[,10],H11=pre

```

					IT-62.08.1081.06 12	Арк.
Змн.	Арк.	№ докум.	Підпис	Дат.		63

```

d[,11],H12=pred[,12],H13=pred[,13],H14=pred[,14],H15=pred[,15],H16=pre
d[,16])
test4 <- cbind(test3, hihat_pred)
test4

#запис результатів у форматі csv
write.csv(test4, 'result.csv', row.names=FALSE)

//метод конвертації MIDI у числовий формат
public int[] MidiToNumbers(File file) throws InvalidMidiDataException,
IOException {
    Sequence sequence = MidiSystem.getSequence(file);
    Track track = sequence.getTracks()[1];
    int[] accents = new int[(track.size()-1)/480];
    for (int i = 0; i < track.size(); i++) {
        MidiEvent event = track.get(i);
        MidiMessage message = event.getMessage();
        if (message instanceof ShortMessage) {
            ShortMessage sm = (ShortMessage) message;
            if (sm.getCommand() == NOTE_ON) {
                int velocity = sm.getData2();
                if (velocity > 100)
                    accents[(int) (event.getTick() / 30)] = 2;
                else if (velocity > 20)
                    accents[(int) (event.getTick() / 30)] = 1;
                else accents[(int) (event.getTick() / 30)] = 0;
            }
        }
    }
    return accents;
}

```

Власник документу:
Лісовиченко Олег Іванович

ID перевірки:
1003791066

Дата перевірки:
04.06.2020 22:52:11 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
04.06.2020 23:03:45 EEST

ID користувача:
76913

Назва документу: IT-62_Kashyrina

ID файлу: 1003804950 Кількість сторінок: 44 Кількість слів: 8375 Кількість символів: 57617 Розмір файлу: 120.18 KB

6.59% Схожість

Найбільша схожість: 1.92% з джерело <https://www.cc.gatech.edu/~mark.guzdial/mediacomp/BETA-dsBook-7-22-09-first4c..>

6.5% Схожість з Інтернет джерелами

138

Page 46

4.35% Текстові збіги по Бібліотеці акаунту

439

Page 47

0% Цитат

Не знайдено жодних цитат

0% Вилучень

Вилучений текст відсутній

Підміна символів

Заміна символів

48