Хмельницький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему *Метод створення музичних композицій на основі технологій генеративного штучного інтелекту*

Галузь знань *12 – Інформаційні технології*
Шифр і назва галузі знань

Спеціальність *122 – Комп'ютерні науки*
Шифр і назва спеціальності

Освітня програма *Комп'ютерні науки*
Назва освітньої програми

Виконав: *студент групи КН-21-2* (Група виконавця) _____ (Підпис) *Андрій ПРИЙМА* (Ім'я, ПРІЗВИЩЕ)

Керівник: *д.т.н.проф.каф.КН* (Науковий ступінь, посада) _____ (Підпис) *Едуард МАНЗЮК* (Ім'я, ПРІЗВИЩЕ)

Нормоконтроль: *к.т.н., доц. каф. КН* (Науковий ступінь, посада) _____ (Підпис) *Руслан БАГРІЙ* (Ім'я, ПРІЗВИЩЕ)

До захисту допускаю:
зав. кафедри КН, д.т.н., професор _____ (Підпис) *Олександр БАРМАК* (Ім'я, ПРІЗВИЩЕ)
09 06 2025 р.

Хмельницький 2025

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет *інформаційних технологій*
Кафедра *комп'ютерних наук*
Освітній ступінь *бакалавр*
Галузь знань *12 – Інформаційні технології*
Спеціальність *122 – Комп'ютерні науки*

ЗАТВЕРДЖУЮ
Завідувач кафедри *комп'ютерних наук*

_____
(підпис)
д.т.н., професор Олександр БАРМАК
«*10*» *02* *2025* року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА

1. Тема кваліфікаційної роботи бакалавра: *«Метод створення музичних композицій на основі технологій генеративного штучного інтелекту»*

2. Завдання видано студенту *Андрію Приймі*
(ім`я, прізвище)

3. Керівник роботи *д.т.н.проф.каф.КН Едуард МАНЗЮК*
(посада, ім`я, прізвище)

4. Затверджено наказом університету від «*07*» *02* *2025* р. № *23*

5. Дата видачі завдання студенту: «*10*» *02* *2025* р.

6. Зміст пояснювальної записки (перелік задач) та вихідні дані:

Мета кваліфікаційної роботи бакалавра – покращення якості та різноманітності генерованих музичних композицій шляхом застосування методів генеративного штучного інтелекту.

Відповідно до мети було поставлено наступні задачі: проаналізувати створення музичних, дослідити існуючі методи створення музичних композицій за допомогою генеративного штучного інтелекту, розробити метод відповідно до мети, розробити відповідну інформаційну систему, що використовує метод автоматизованого створення музичних композицій на основі технологій генеративного штучного інтелекту, провести тестування створеної інформаційної системи, дослідити ефективність інформаційної системи.

7. Календарний план виконання кваліфікаційної роботи бакалавра:

№	Назва етапів (розділів) кваліфікаційної роботи бакалавра	Термін виконання	Примітка
1	*Вибір напряму дослідження та узгодження тематики кваліфікаційної роботи бакалавра з керівником, складання календарного графіка виконання роботи*	*січень 2025*	
2	*Ознайомлення з предметною областю, формулювання мети та задач дослідження, визначення об'єкта та предмета дослідження*	*лютий 2025*	
3	*Проектування та розробка методу створення музичних композицій на основі генеративного штучного інтелекту*	*березень 2025*	
4	*Створення відповідної інформаційної системи, дослідження ефективності, проведення експериментів*	*квітень 2025*	
5	*Написання пояснювальної записки, урахування зауважень керівника, оформлення згідно вимог*	*травень 2025*	
6	*Розробка презентаційних матеріалів та попередній захист кваліфікаційної роботи*	*травень 2025*	
7	*Отримання відгуку керівника, рецензії, перевірка на плагіат, нормоконтроль*	*червень 2025*	
8	*Підготовка до захисту та захист кваліфікаційної роботи бакалавра*	*червень 2025*	

Виконавець: *студент групи КН-21-2* _____ *Андрій ПРИЙМА*
Група виконавця | Підпис | Ім'я, ПРІЗВИЩЕ

Керівник: *д.т.н.проф.каф.КН* _____ *Едуард МАНЗЮК*
Науковий ступінь, посада | Підпис | Ім'я, ПРІЗВИЩЕ

Анотація

Тема кваліфікаційної роботи бакалавра: *«Метод створення музичних композицій на основі технологій генеративного штучного інтелекту»*

Виконавець кваліфікаційної роботи бакалавра: *студент групи КН-21-2 Андрій ПРИЙМА*

Керівник кваліфікаційної роботи бакалавра: *д.т.н.проф.каф.КН Едуард МАНЗЮК*

Кваліфікаційна робота бакалавра містить:

Пояснювальна записка				Кількість додатків
Сторінок	Рисунків	Таблиць	Джерел інформації	
50	18	1	46	3

Мета кваліфікаційної роботи бакалавра – покращення якості та різноманітності генерованих музичних композицій шляхом застосування методів генеративного штучного інтелекту. Для розроблення методу було використано мову програмування Python, модель HuggingFace.

Розроблений метод дає змогу локально розгорнути штучний інтелект, що здатний створити музикальну композицію за запитом користувача, завдяки чому можна дослідити з якою якістю створюється музикальна композиція штучним інтелектом та дає можливість музикантам знайти натхнення для створення нових музикальних композицій.

Ключові слова: Штучний інтелект, нейронна мережа, музика, композиція, модель.

Виконавець: *студент групи КН-21-2* (Група виконавця) _____ (Підпис) *Андрій ПРИЙМА* (Ім'я, ПРІЗВИЩЕ)

Зміст

Перелік скорочень

Скорочення, термін, позначення	Пояснення
БД	База даних
ІНМ	Інформаційний навчальний матеріал
ІС	Інформаційна система
ІТ	Інформаційні технології
КРБ	Кваліфікаційна робота бакалавра
КН	Комп’ютерні науки
НК	Навчальний курс
НМ	Нейронна мережа
ПЗ	Пояснювальна записка
ПП	Програмний продукт
ТНМ	Тестовий навчальний матеріал
ХНУ	Хмельницький національний університет.
DAW	Цифрова аудіо-робоча станція
VST	Віртуальні студійні технології
LSTM	Long short-term memory
RNN	Рекурентна нейронна мережа
DDD	Доменно-орієнтоване проєктування
GAN	Generative Adversarial Network
VAE	Варіаційний автокодер

Вступ

Кваліфікаційна робота бакалавра присвячена розробці методу створення музичних композицій на основі технологій генеративного штучного інтелекту та реалізації відповідної системи, яка дозволяє автоматизовано генерувати музичний контент різного стилістичного та емоційного спрямування. Сучасний розвиток технологій штучного інтелекту, зокрема генеративних моделей, відкриває нові можливості для автоматизації творчих процесів, серед яких створення музики займає особливе місце.

Актуальність. Музика є однією з найпоширеніших форм творчого самовираження, що відіграє важливу роль у культурному, емоційному та соціальному житті людей. Традиційне створення музичних композицій вимагає високого рівня підготовки, натхнення та значних часових ресурсів. У той же час, стрімкий розвиток генеративного ШІ, зокрема моделей на базі нейронних мереж, дозволяє автоматизувати окремі етапи композиторської діяльності, відкриваючи нові горизонти для музикантів, продюсерів і розробників мультимедійного контенту. Актуальність теми зумовлена зростаючим попитом на інструменти, які дозволяють генерувати якісний музичний матеріал в автоматизованому режимі, а також потребою в нових підходах до творчої взаємодії людини з машиною. Генеративні технології, такі як GPT[1], Diffusion Models[2], трансформери та рекурентні нейронні мережі, дедалі частіше застосовуються для генерації текстів, зображень, а з нещодавнього часу – й музики. Це робить тему дослідження особливо актуальною в умовах цифрової трансформації творчих індустрій.

Об’єкт дослідження – процес створення музичних композицій за допомогою генеративних технологій.

Предмет дослідження – методи та технології генеративного штучного інтелекту для автоматизованого створення музичних композицій.

Мета кваліфікаційної роботи бакалавра – покращення якості та різноманітності генерованих музичних композицій шляхом застосування методів генеративного штучного інтелекту.

Завдання кваліфікаційної роботи бакалавра – провести аналіз предметної області автоматизованого створення музичних композицій за допомогою генеративного штучного інтелекту. Виконати аналіз існуючих методів генерації музики та дослідити можливості їх застосування у різних жанрових і стилістичних напрямках. Визначити особливості використання генеративних моделей, зокрема нейронних мереж, у процесі музичної генерації. Проаналізувати наявні програмні рішення та фреймворки, що реалізують генерацію музики з використанням ШІ. Розробити метод створення музичних композицій на основі генеративних моделей штучного інтелекту. Спроєктувати архітектуру інформаційної системи для генерації музики з урахуванням вибраного підходу. Виконати вибір засобів розробки системи генерації музики з використанням генеративного ШІ. Реалізувати програмний прототип системи для автоматизованого створення музичних композицій. Провести тестування розробленої системи генерації музики. Виконати дослідження якості та різноманітності створених музичних композицій за допомогою розробленого методу.

Розділ 1 Характеристика предметної області: аналіз моделей, методів та реалізацій

1.1 Аналіз інформаційних моделей генерації музики

Музика [3] – це одна з найдавніших форм мистецтва, що виникла ще в доісторичні часи й супроводжує людство впродовж усього його культурного розвитку. Вона базується на організованому звуковому матеріалі, який впорядковується у часі. Музика виконує різноманітні функції – емоційну, соціальну, естетичну, комунікативну – та має величезний вплив на свідомість і поведінку людини [4]. Музичне мистецтво охоплює широкий спектр жанрів, стилів і форм, від класичної симфонії до електронної композиції, від народної пісні до експериментального саунду [5].

З точки зору музичної теорії, будь-який музичний твір складається з елементів, таких як мелодія, гармонія, ритм, тембр та динаміка [6]. Мелодія[7] – це послідовність музичних звуків, організована у логічну й впізнавану структуру. Гармонія[8] визначає співзвуччя кількох нот, які утворюють акорди та гармонічні послідовності. Ритм[9] задає часову структуру музики, організовану у такти, метри й фрази. Тембр[10] надає кожному інструменту або голосу унікального звучання, а динаміка – варіативність у гучності виконання.

Створення музичного твору традиційно є результатом творчої діяльності композитора[11]. Цей процес включає в себе генерацію ідеї, формування основної музичної теми[12], її розвиток, гармонізацію[13], оркестровку[14], а також аранжування[15] та запис. У випадку пісні можуть додаватися текстові складові – лірика, яка також має власну художню структуру і вписується у загальну композицію [16].

З переходом людства у цифрову епоху, музика також зазнала значної трансформації. Сьогодні більшість музичних творів створюється, зберігається, обробляється та відтворюється в цифровому вигляді. Широке розповсюдження отримали цифрові аудіо-робочі станції (DAW – Digital Audio Workstation[17]), віртуальні інструменти (VST – Virtual Studio Technology[18]), MIDI-

редактори[19], цифрові синтезатори та інші програмні засоби [20]. Музичні твори можуть бути створені або змонтовані без використання фізичних інструментів, повністю у віртуальному середовищі, що робить музику доступнішою для більш широкого кола авторів.

Сучасні інформаційні технології, зокрема штучний інтелект, відкривають нові горизонти у сфері музичного мистецтва. Генеративні моделі ШІ[21], такі як нейронні мережі, можуть не тільки аналізувати існуючі твори, а й створювати нові композиції на основі заданих параметрів або прикладів [22]. Такі системи, як OpenAI MuseNet[23], Google Magenta[24] або AIVA (Artificial Intelligence Virtual Artist)[25], демонструють здатність генерувати музичні фрагменти у різних жанрах і стилях. Їхня робота базується на попередньому аналізі великих масивів музичних даних, вивченні закономірностей гармонії, мелодики, ритміки та інших параметрів, що дозволяє моделювати власну творчу поведінку [26].

Таким чином, предметна область створення музичних композицій, особливо в контексті використання генеративного ШІ, включає низку важливих понять, процесів і сутностей. До основних сутностей належать композиція, інструмент, тема, мотив, акорд, структура твору, жанр, темп, ритм, тональність тощо. До ключових процесів – створення мелодії, аранжування, гармонізація, структурування твору, додаткова обробка та збереження результату у цифровому форматі [27].

Актуальними питаннями, що можуть бути автоматизовані у цій галузі, є автоматичне створення гармонійно цілісних мелодій, підбір акомпанементу, аранжування твору в певному стилі, перетворення тексту в музику, а також генерація композицій відповідно до емоційного стану користувача або заданого настрою. Ці процеси можуть бути реалізовані на основі алгоритмів генеративного ШІ, що дозволить значно підвищити ефективність, швидкість і варіативність створення музичних творів у цифрову епоху.

1.2 Огляд теоретичних підходів до розв’язку подібних задач

В умовах стрімкого розвитку цифрових технологій та зростання обсягів даних все більш актуальним стає застосування штучного інтелекту (ШІ) та інформаційних технологій (ІТ) у сфері створення музичних композицій. Інтелектуальні системи мають потенціал не лише для автоматизації окремих етапів музичного виробництва, а й для повноцінної генерації нових творів, що відкриває нові можливості як для професійних композиторів, так і для широкого кола користувачів [28].

Серед сучасних напрямків використання ШІ в музиці можна виокремити генеративні моделі, розпізнавання стилю, класифікацію жанрів, адаптивне зведення треків, синтез звуків, автоматичне створення акомпанементу, пошук схожих мелодій та ритмів тощо [29]. Особливо важливим є напрям генеративного ШІ, де застосовуються моделі, здатні самостійно створювати музичні композиції на основі аналізу величезних обсягів навчальних даних [30].

Серед найбільш поширених методів штучного інтелекту у цій сфері – нейронні мережі (глибокі, рекурентні, згорткові), трансформери та моделі типу LSTM (Long Short-Term Memory). Рекурентні нейронні мережі (RNN) особливо ефективні для обробки послідовностей, таких як нотні ряди чи MIDI-події, що робить їх зручними для роботи з музикою [31]. Трансформери, такі як GPT або Music Transformer, демонструють високу якість у генерації складних композицій за рахунок здатності працювати з довгими залежностями у даних [32]. Алгоритми навчання нейромереж – такі як зворотне поширення помилки (backpropagation) – забезпечують поступове покращення якості результату під час тренування моделі [33].

Також у сфері музики застосовуються моделі подання знань, експертні системи, а також методи кластеризації й пошуку закономірностей для аналізу музичних вподобань користувачів [34]. Для визначення жанру або подібності композицій використовуються методи розпізнавання образів, де мелодії

перетворюються на спектрограми – зображення, що можуть бути оброблені згортковими нейромережами [35].

З боку програмування у реалізації інтелектуальних музичних систем часто використовуються принципи об’єктно-орієнтованого програмування (ООП [36]), що дозволяють організувати логічну й масштабовану структуру проєкту. Принципи SOLID та концепція доменно-орієнтованого проєктування (DDD [37]) сприяють створенню гнучких і легко підтримуваних систем.

1.3 Аналіз існуючих програмних засобів та наукових рішень

Одним із найвідоміших сучасних інструментів для генерації музики за допомогою штучного інтелекту є SUNO [38] – онлайн-сервіс, що дозволяє створювати повноцінні музичні композиції, включаючи інструментал і вокал, лише за допомогою текстового опису. Цей програмний продукт працює у вигляді вебплатформи, доступної через браузер, і не потребує встановлення програмного забезпечення чи спеціальних навичок роботи з музичними редакторами.

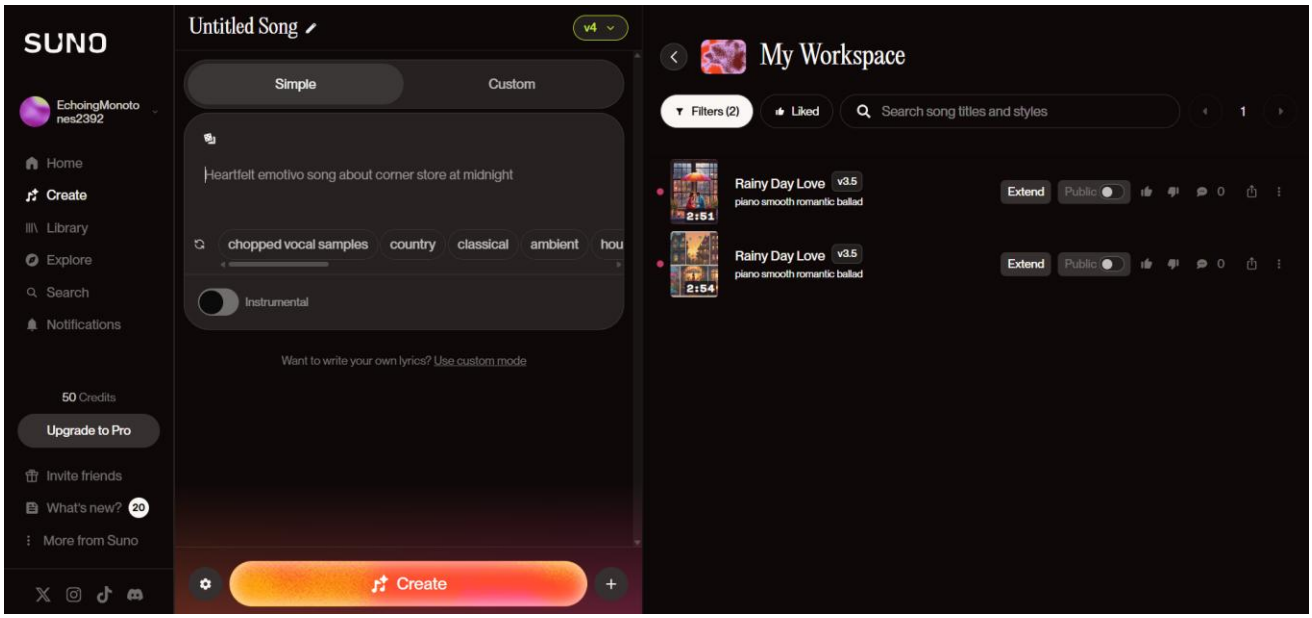


Рисунок 1.1 – Інтерфейс SUNO [38]

Головною особливістю SUNO є можливість створення пісні, яка звучить як справжній запис студійного треку, на основі короткої текстової інструкції, яку

користувач вводить у відповідне поле. Наприклад, користувач може ввести фразу типу "sad pop song about lost love" або "fast-paced metal track with dark vocals", після чого система згенерує мелодію, підбере відповідний стиль, створить текст пісні, а також синтезує вокал, що виконує цю пісню.

Система використовує сучасні мовні та аудіогенеративні моделі, що працюють на основі глибокого навчання. В основі SUNO лежить гібридний підхід, що поєднує великі мовні моделі для створення текстового наповнення пісні та спеціалізовані аудіомоделі для синтезу звуку та вокалу. Це дозволяє створювати композиції, які мають логічну структуру, гармонійну мелодію, стилістичну цілісність і реалістичний вокал, що може звучати різними голосами і навіть на різних мовах. Користувачі можуть вводити як власний текст пісні, так і дозволяти системі згенерувати його автоматично на основі заданої тематики чи емоційного тону. SUNO також дозволяє редагувати назву пісні, обрати зображення для обкладинки альбому, зберегти трек у вигляді аудіофайлу та поділитися ним за посиланням.

Одним з найбільших досягнень SUNO є якість синтезованого вокалу: голос звучить виразно, має інтонації, наголоси, динаміку та навіть стилістичні особливості співу, властиві людському голосу. Такий рівень деталізації дозволяє створювати композиції, які на слух не відрізняються від пісень, записаних реальними виконавцями. Сервіс підтримує велику кількість музичних жанрів – від попу і року до електроніки, хіп-хопу, джазу чи навіть експериментальної музики. SUNO активно використовується як аматорами, що хочуть створити музику для TikTok[39] чи YouTube[40], так і професійними продюсерами для швидкого створення драфтів, демонстраційних треків або концептів. Попри високу якість і простоту використання, сервіс має низку обмежень. У безкоштовній версії користувач отримує обмежену кількість генерацій на день, а доступ до розширених функцій (наприклад, збереження у високій якості чи комерційне використання) можливий лише за підпискою. Крім того, хоча генерація музики є вражаючою, наразі користувач не має можливості глибоко редагувати створену композицію – наприклад, змінити акордову послідовність, структуру пісні,

тривалість куплетів або темп. Також важливою особливістю є закритість системи: SUNO не має відкритого API чи SDK, тому неможливо інтегрувати його у сторонні додатки або використовувати у власних програмних продуктах. Ще одним обмеженням є відсутність підтримки багатьох мов – система добре працює переважно з англійською, тоді як генерація українською або іншими мовами може бути нестабільною або взагалі недоступною.

Ще одним відомим проєктом у сфері генеративної музики є Magenta[41] – відкритий дослідницький проєкт, розроблений компанією Google на базі Google Brain. Його основна мета – дослідити, як штучний інтелект може допомогти людям у процесі творчості, зокрема в музиці та образотворчому мистецтві. На відміну від SUNO, який є закритим і орієнтованим на кінцевого користувача, Magenta – це ціла екосистема інструментів, моделей і бібліотек, які можна використовувати для побудови власних систем генерації музики.

Рисунок 1.2 – Інтерфейс застосунку на базі Magenta [41]

Цей проєкт є повністю відкритим і доступним для розробників, дослідників і музикантів, які хочуть глибше зрозуміти принципи роботи генеративного ШІ або створити власні творчі інструменти. Magenta базується на технологіях машинного навчання, зокрема на нейронних мережах типу RNN , VAE (варіаційні автокодери) та трансформерах. У рамках проєкту створено кілька моделей для генерації мелодій, гармоній, барабанних партій, акомпанементу та навіть повноцінних композицій. Наприклад, одна з найвідоміших моделей – MusicVAE – здатна не лише створювати нові мелодії, а й трансформувати одну мелодію в іншу, зберігаючи гармонійну послідовність і музичну логіку. Інша

модель, Performance RNN, була навчена на виконаннях фортепіанної музики і здатна генерувати віртуозні, насичені деталями фортепіанні композиції, враховуючи динаміку, тривалість нот і емоційну виразність виконання.

Однією з головних переваг Magenta є його інтеграція з платформами для роботи з музикою. Magenta підтримує інтеграцію з TensorFlow – популярною бібліотекою для машинного навчання, а також має плагіни для цифрових аудіо-робочих станцій, таких як Ableton Live (плагін Magenta Studio). Завдяки цьому користувачі можуть генерувати музичні ідеї прямо у своєму музичному середовищі, комбінуючи ШІ-генеровані партії з власною творчістю. Крім того, Magenta має вебінтерфейс, де можна протестувати деякі з моделей без необхідності програмування, що робить проєкт доступним також для музикантів без технічної підготовки.

Magenta має багату документацію, приклади коду, навчальні матеріали, GitHub-репозиторії та активну спільноту. Завдяки цьому платформа є ідеальним середовищем для навчання та експериментів. Важливо також зазначити, що Magenta – це не лише інструмент, але й наукова ініціатива: команда розробників активно публікує статті у сфері машинного навчання для музики, досліджуючи нові підходи до навчання моделей, аналізу структури музики, стилістичної адаптації та емоційної виразності.

Разом з тим, проєкт має свої обмеження. По-перше, для повноцінної роботи з Magenta часто потрібні технічні знання – як у програмуванні (Python, TensorFlow), так і у теорії музики. По-друге, результат генерації потребує інтерпретації: моделі не завжди створюють "готовий продукт", а радше пропонують ідеї чи заготовки, які потім музикант повинен обробити. Це робить Magenta потужним інструментом для підтримки творчого процесу, але не завжди кінцевим рішенням для створення повноцінних треків без участі людини.

Ще одним прикладом програмного забезпечення у сфері генеративної музики є Soundraw[42] – онлайн-сервіс, який використовує штучний інтелект для створення унікальних музичних треків на основі параметрів, заданих користувачем.

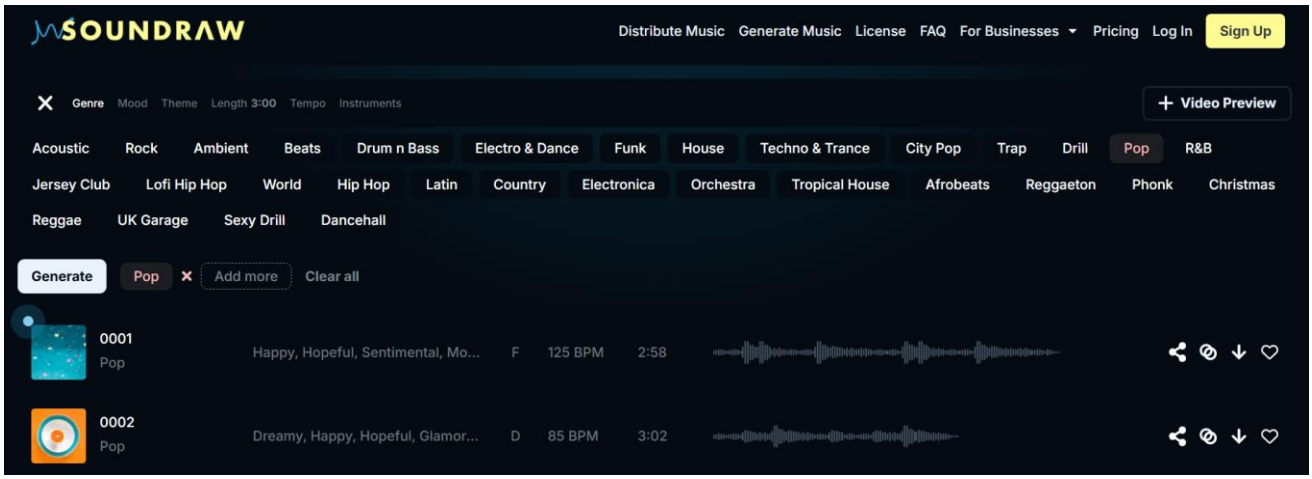


Рисунок 1.3 – Інтерфейс Soundraw [42]

Сервіс орієнтований на людей, яким потрібна швидка генерація музики для супроводу відео, презентацій, ігор або інших медіапроєктів. Основною перевагою Soundraw є надзвичайна простота у використанні: користувач обирає жанр, настрій, тривалість композиції, а система миттєво генерує декілька варіантів треків. Окремо можна налаштувати структуру пісні – змінити порядок і тривалість частин (куплет, приспів, бридж тощо). Для зручності всі варіанти миттєво програються у браузері, а фінальний результат можна експортувати у форматі .mp3.

Soundraw надає також функцію "регеренерації" окремих частин треку – користувач може замінити лише одну частину композиції, зберігаючи решту без змін. Такий підхід дозволяє досягати більш точного результату без потреби створювати трек з нуля. Сервіс працює на основі власного пропрієтарного алгоритму, тож внутрішні технічні деталі генерації не розкриваються. Soundraw доступний за підпискою, має обмежений безкоштовний функціонал та повний доступ за умовами платного плану. Також, попри заявлену унікальність, іноді результати звучать шаблонно або однотипно.

Аналіз розглянутих рішень засвідчує активний розвиток технологій генеративного ШІ у сфері створення музики. Незважаючи на суттєві досягнення, більшість сучасних систем або закриті (як SUNO, Soundraw), або потребують технічної підготовки для використання (як Magenta). Це створює нішу для

розробки програмного забезпечення, яке було б одночасно інтуїтивно зрозумілим, гнучким у налаштуваннях і відкритим для творчих експериментів. У межах цієї тематики доцільно створити метод, орієнтований на генерацію музичних композицій за допомогою ШІ з можливістю взаємодії користувача на кожному етапі. Зважаючи на доступність та універсальність, найбільш доцільною є розробка віконного додатку для персонального комп'ютера, або додатку з вебінтерфейсом, який можна розгорнути локально. Такий формат дозволяє забезпечити повноцінний графічний інтерфейс, зручне керування параметрами генерації та інтеграцію з локальними ресурсами (наприклад, MIDI або DAW). Отже, метою КРБ є створення програмного засобу для генерації музичних композицій з використанням технологій генеративного ШІ, орієнтованого на зручність користувача, відкритість та гнучкість у творчому процесі.

1.4 Мета, задачі та вимоги до реалізації інформаційної системи

Метою кваліфікаційної роботи бакалавра є покращення якості та різноманітності генерованих музичних композицій шляхом застосування методів генеративного штучного інтелекту. Відповідно до мети було поставлено наступні задачі:

– проаналізувати сучасний стан створення музичних композицій за допомогою генеративного штучного інтелекту;

– дослідити існуючі методи створення музичних композицій за допомогою генеративного штучного інтелекту;

– розробити метод автоматизованого створення музичних композицій на основі технологій генеративного штучного інтелекту;

– розробити відповідну інформаційну систему, що використовує метод автоматизованого створення музичних композицій на основі технологій генеративного штучного інтелекту;

– провести тестування створеної інформаційної системи;

– дослідити ефективність інформаційної системи.

Розділ 2 Проєктування інформаційної системи

2.1 Моделі, методи, інформаційна технологія системи створення музичних композицій за допомогою генеративного штучного інтелекту

Метою кваліфікаційної роботи є покращення якості та різноманітності генерованих музичних композицій шляхом застосування методів генеративного штучного інтелекту. Планована система працюватиме на основі генеративної нейронної архітектури, зокрема моделі типу GAN, адаптованої для створення багатотрекових фрагментів у вигляді piano-roll. Розроблений метод передбачає поетапну генерацію музики у вигляді послідовності тактів. Кожен такт являє собою тривимірний тензор розмірності "час * висота ноти * трек", де трек – це окремий інструментальний канал як барабани, бас, гітара, фортепіано, струнні.

Для роботи методу, йому потрібні вхідні данні. Такими даними є шумові вектори (як загальні, так для кожного вектору, зі збереженням часової залежності) та, що є опціональним, реальний трек чи приклад треку, з яким буде працювати метод при умові, якщо це не з чистого листа, а також заздалегідь визначена кількість тактів та типи інструментів, які будуть грати.

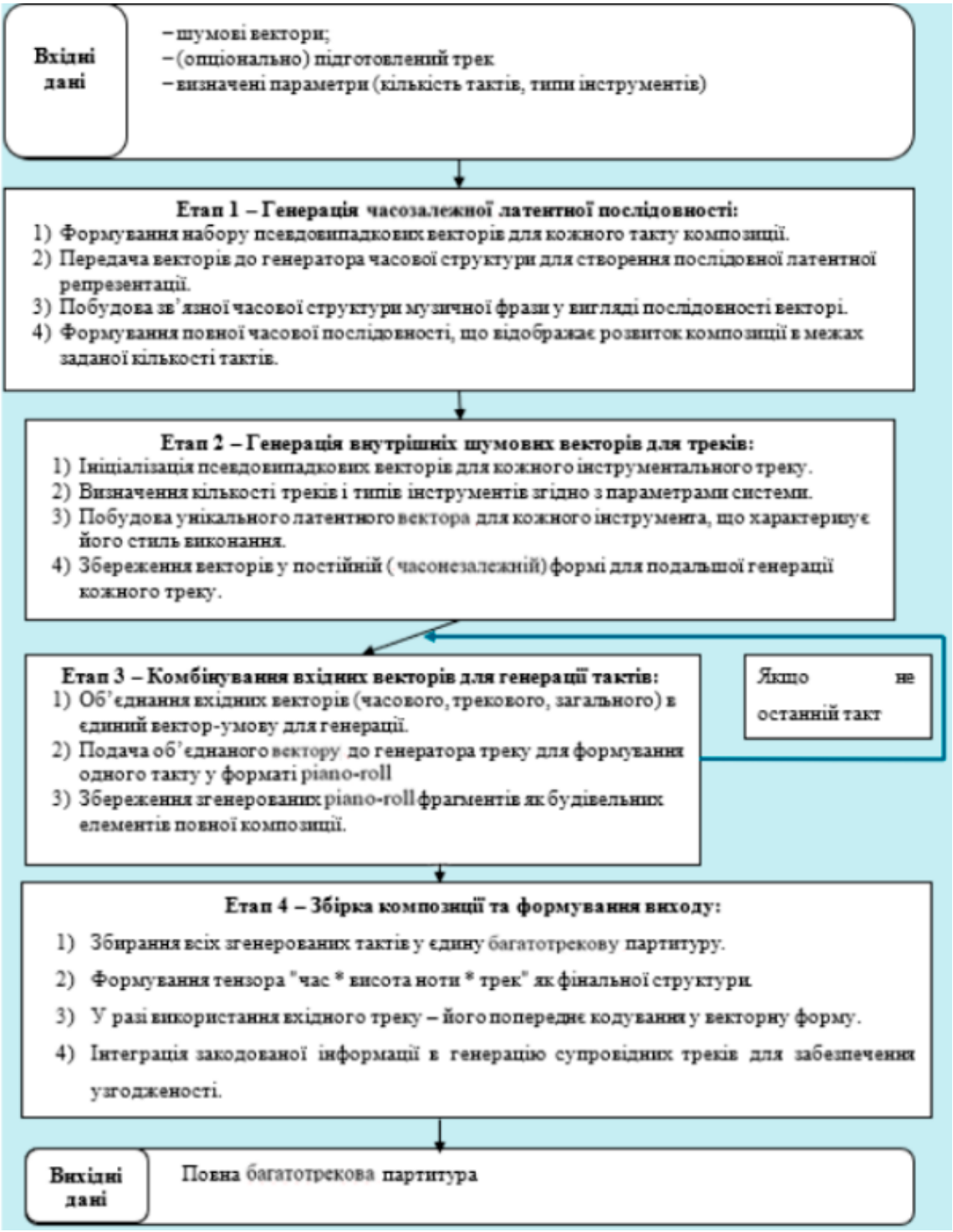


Рисунок 2.1 – Схема методу генерації музичних композицій

На першому кроці цього методу, окремий генератор формує послідовність латентних векторів для кожного такту з урахуванням музичної логіки розвитку. Такий генератор імітує музичну фразу, забезпечуючи зв'язність між окремими частинами композиції. В результаті отримується набір векторів, кожен з яких буде відповідати одному такту.

У другому етапі генеруються векторні представлення для кожного інструментального треку. Вони є незмінними в часі, але індивідуальними для кожного інструменту, що дозволяє задавати «стиль» або характер треку. Це може бути агресивніша бас партія чи більш м’яке піаніно.

На третьому етапі кожен генератор треку отримує комбінацію з загального шумового вектору, часового вектору та вектору треку. Ця комбінація подається на вхід генератору, який формує один такт одного інструментального треку у вигляді piano-roll. Повторення кроку для кожного треку і кодного такту формує повноцінну композицію.

Останній етап перед кінцевим результатом є четвертий етап – збірка композиції та формування виходу. В кінці, усі згенеровані фразменти по треках і тактах об’єднуються у єдиний масив, який інтерпретується як повна партитура композіції. У разі використання вхідного треку, його попередньо кодують у векторну форму і вводять у відповідні блоки генерації.

Завдяки цим крокам цього методу, у результаті отримується повна, закінчена партитура аудіокомпозиції.

2.2 Аналіз та автоматизація обробки потоків даних / Функціональна структура інформаційної системи

Функціонально система поділена на кілька логічних груп функцій, кожна з яких забезпечує виконання конкретного етапу процесу. Це є функція ініціалізації параметрів генерації, функція генерації структури композиції, функція побудови музичних тактів та фукнція збереження та виведення результату.

Рисунок 2.2 – Функціональна схема методу

Першою групою є ініціалізація параметрів генерації. Її функції:

– встановлення кількості тактів для майбутньої композиції;

– вибір кількості інструментальних треків;

– вибір типів інструментів для кожного треку;

– ініціалізація випадкових шумових векторів.

Дана функціональна група відповідає за формування вхідних параметрів системи, на основі яких буде здійснюватися подальша генерація. Користувач задає ключові характеристики композиції: скільки триватиме трек у тактах, скільки треків у ньому має бути та які інструменти використовуються (наприклад, бас, ударні, гітара, фортепіано). Після цього система формує набір випадкових латентних векторів, які служать стартовими умовами для генераторів. Ця група функцій реалізує вхідну частину методу, яка задає основні структурні параметри композиції та формує латентний простір для генерації. Від правильності й узгодженості цього блоку залежить якість та цілісність усього результату.

Другою групою функцій є генерація структури композицій. Її функціями є:

– побудова часової послідовності латентних векторів (один на такт);

– генерація незмінних шумових векторів для кожного інструментального треку;

– комбінування всіх вхідних векторів у структури для генерації тактів.

Цей процес відповідає за внутрішню логіку створення композиції. Він базується на алгоритмах глибокого навчання та моделює музичну фразу як послідовність пов’язаних між собою латентних представлень. На цьому етапі створюються як часозалежні вектори (які забезпечують логічну зв'язність між тактами), так і інструментозалежні (що відповідають за стиль і характер кожного треку). Особливістю цього блоку є його двокомпонентна побудова: генерація часової логіки та генерація стилістичної наповненості кожного інструмента. Їх комбінування дає змогу отримати складну і водночас впорядковану структуру композиції, подану у вигляді piano-roll для кожного інструмента окремо.

Третій набір функцій відповідає групі функцій побудови музичних тактів. Функціями цієї групи є:

– подача комбінованих векторів у генератор тактів;

– побудова piano-roll для кожного такту окремо;

– послідовне з’єднання тактів у межах одного треку;

– формування повної партитури за всіма треками.

Цей блок реалізує безпосередній процес створення музики, що базується на побудові кожного такту окремо, з урахуванням часової логіки та інструментальної специфіки. Для кожного інструмента формується повна лінія, яка складається з послідовно згенерованих тактів. Важливим моментом є те, що система працює ітеративно: для кожного інструмента й кожного такту створюється окрема piano-roll матриця, яка зберігається і згодом з'єднується з іншими. Це дозволяє зберігати узгодженість та гармонійність між треками.

Остання група функцій відповідає за збереження та вивід результату. Функціями цієї групи є:

– об’єднання всіх треків у фінальний багатотрековий масив;

– конвертація результату у формат MIDI;

– збереження отриманої композиції на диск;

– виведення результату у вигляді текстового подання (опційно: графічна візуалізація piano-roll).

Ця група функцій відповідає за завершення роботи методу і надання користувачеві фінального результату. Після того як композиція згенерована, система формує її фінальне представлення, суміщаючи всі інструментальні треки у єдину структуру. Далі ця структура експортується у формат, придатний для подальшого використання – наприклад, у музичному редакторі чи DAW.

2.3 Розробка архітектури нейронної мережі

У межах розробленого методу генерації багатотрекової музики використовується певний набір вхідних даних, які формують початковий простір для роботи нейронної архітектури. Дані вводяться у модель для подальшої генерації музичних композицій у форматі piano-roll. Кожен тип вхідних даних виконує свою окрему функцію, забезпечуючи як загальну креативність генерації, так і варіативність між окремими треками та тактами. Окрім вхідних даних, важливо визначити також форму вихідних даних, оскільки саме вони є кінцевим продуктом роботи методу та слугують основою для подальшого використання результатів у практичних цілях.

До вхідних даних моделі належать:

– загальний шумовий вектор;

– набір шумових векторів для кожного інструментального треку;

– набір часозалежних шумових векторів для генерації часової структури композиції.

Загальний шумовий вектор є початковим латентним вектором, який містить згорнуту інформацію про всі характеристики майбутньої композиції. Він має фіксовану розмірність і слугує джерелом випадковості для генератора часової структури. Набір шумових векторів для інструментальних треків формується окремо для кожного треку. Кожен вектор у цьому наборі визначає особливості стилю та виконання відповідного інструменту. Ці вектори не змінюються у часі і

залишаються сталими протягом усієї генерації композиції. Набір часозалежних шумових векторів створюється окремо для кожного такту майбутньої композиції. Ці вектори вводять до моделі змінність у часі, дозволяючи формувати розвиток музичної теми від одного такту до іншого. Таким чином, часозалежні вектори забезпечують динаміку та природність змін у композиції.

На виході модель генерує багатовимірний тензор, який представляє собою зведену символічну партитуру. Формат вихідних даних описується чотиривимірною структурою:

$$X \in \{0,1\}^{T \cdot R \cdot P \cdot M},$$

Де T – кількість тактів у згенерованій композиції, R – кількість часових поділів у межах одного такту (наприклад, 96 поділів для стандартної ритмічної сітки), P – кількість допустимих висот нот, що охоплюють типовий музичний діапазон (наприклад, 84 ноти від C1 до B7), M – кількість інструментальних треків, що беруть участь у генерації.

Кожен елемент вихідного тензора має значення 0 або 1, що вказує на відсутність або наявність певної ноти у відповідний момент часу для конкретного інструментального треку. У такий спосіб, сформована вихідна структура може бути інтерпретована як повна багатотрекова композиція, готова для подальшого конвертування у формат MIDI або синтезу в аудіофайл. Вхідні та вихідні дані мають узгоджену структуру, що дозволяє забезпечити цілісність роботи всієї архітектури та дає можливість ефективно контролювати процес генерації на різних етапах. Важливо, що така організація даних дає змогу як створювати композиції "з нуля", так і забезпечувати варіативність за рахунок змін параметрів вхідних векторів.

Дана нейронна архітектура для генерації багатотрекової музики побудована за принципами каскадної структури і складається з декількох ключових компонентів, кожен з яких виконує специфічну роль у загальному процесі генерації композиції. Композиція архітектури розділена на логічні блоки, що взаємодіють між собою послідовно, забезпечуючи створення музичних фраз, їх внутрішню гармонійну будову та цілісність усієї композиції в цілому.

Першим блоком є вхідний блок, у якому здійснюється підготовка латентного простору для подальшого перетворення у музичну структуру. Вхідні дані формуються у двох можливих режимах: або як випадкові шумові вектори, які ініціалізуються на початку генерації (у разі повністю автоматичної побудови композиції), або ж як приклад умовного треку, на основі якого модель має згенерувати супровід. У першому випадку формується три види латентних векторів: міжтрековий шумовий вектор, що забезпечує глобальну гармонійну узгодженість, вектори для кожного окремого треку, які вносять індивідуальні характеристики у звучання кожного інструменту, а також часозалежні вектори, що дають змогу моделі моделювати динамічний розвиток музики у часі. У другому режимі, який реалізує умовну генерацію, вхідними даними є заданий користувачем трек, що репрезентується у вигляді piano-roll. Цей трек далі передається до енкодера, який перетворює його у часову латентну послідовність, що відображає темп, структуру та музичний контекст вхідного матеріалу.

Енкодер умовного треку є функціональним аналогом генератора часової структури і використовується виключно в умовному режимі. Його основним завданням є витягнення високорівневих ознак із заданого треку та перетворення їх у послідовність латентних векторів. Результатом роботи енкодера є векторна репрезентація для кожного такту, яка надалі використовується у побудові інших треків.

Генератор часової структури виконує аналогічну функцію, але застосовується у випадку генерації з нуля. Він приймає на вхід міжтрековий часозалежний шумовий вектор і перетворює його у послідовність латентних векторів. Він виступає механізмом, який задає логіку розвитку композиції, створюючи основу, на якій будуватимуться окремі інструментальні партії.

Наступним етапом у процесі генерації є використання приватних генераторів часової структури, які відповідають кожному інструментальному треку окремо. Ці генератори працюють з внутрішньотрековими шумовими векторами та зберігають часову логіку, узгоджену з глобальним контекстом композиції. Кожен з них генерує окрему послідовність векторів, яка відображає

специфіку розвитку конкретного інструмента. Таке розділення дозволяє уникнути шаблонності та надати кожному треку власну динаміку розвитку, при цьому залишаючись у межах загального музичного контексту.

Ці часові вектори, разом із глобальним шумовим вектором та індивідуальними векторами треків, подаються на вхід до генераторів тактів – компонентів, відповідальних за формування piano-roll для кожного окремого інструментального треку в межах одного такту. У цьому модулі здійснюється об'єднання кількох джерел інформації у комбінований вектор, який потім проходить через кілька повнозв’язних шарів із нелінійною активацією. Це дає змогу моделі адаптивно витягнути значущі ознаки з латентного простору. Кінцева частина генератора тактів представлена шарами транспонованих згорток, які виконують функцію розгортання латентного вектору у двовимірну матрицю, що репрезентує нотну структуру – тобто, формує piano-roll із заданою кількістю нот і часових сегментів. Саме цей етап є найближчим до фактичної побудови звукового матеріалу, адже саме тут визначається, які ноти будуть присутні у кожному такті, і коли вони будуть звучати.

Коли усі треки сформовано, відбувається етап об'єднання результатів у повноцінну композицію. Це технічний, проте критично важливий етап, адже він акумулює вихідні дані від усіх генераторів тактів і перетворює їх у багатовимірний тензор, що відповідає всій композиції. Цей формат є стандартним представленням у вигляді piano-roll і може бути збереженим у вигляді MIDI або використаним для подальшої обробки.

На етапі тренування до роботи долучається дискримінатор – ключовий компонент у структурі GAN, який відповідає за розрізнення між справжніми (реальними) музичними прикладами з датасету та згенерованими зразками. На завершальному етапі дискримінатор переходить до повнозв’язного шару з одним вихідним нейроном, активація якого відбувається через сигмоїдну функцію. Цей нейрон виводить ймовірність того, що композиція є справжньою. Результат використовується як функція втрат для генераторів, які навчаються таким чином,

щоб "обманювати" дискримінатор – тобто, створювати максимально реалістичні музичні структури.

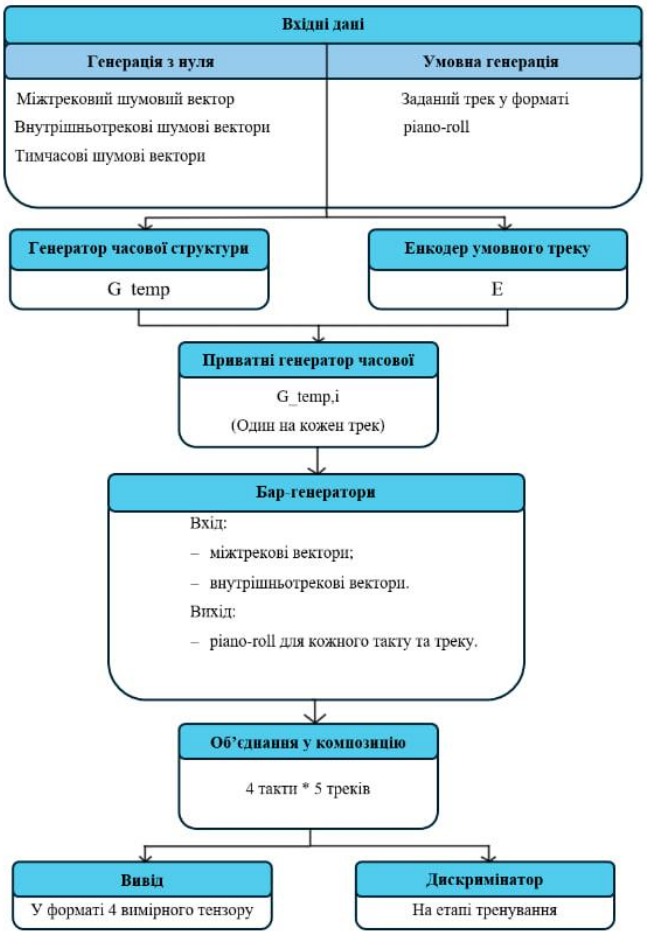


Рисунок 2.3 – Схема взаємодії компонентів нейронної мережі

Усі компоненти архітектури тісно взаємопов'язані між собою, утворюючи комплексну, але логічно послідовну структуру, яка дозволяє ефективно перетворювати випадкові латентні вектори у повноцінні музичні композиції. Така організація роботи нейромережевої архітектури забезпечує гнучкість, модульність та можливість подальшого розширення або адаптації методу до інших музичних стилів чи специфікацій.

2.4 Проектна архітектура системи та взаємозв’язок компонентів

Відповідно до розробленого методу генерації багатотрекових музичних композицій за допомогою технологій генеративного штучного інтелекту, було спроєктовано відповідну структуру програмної системи, що забезпечує виконання всіх основних функцій, визначених на етапах планування. Структура системи розроблена з урахуванням логічної послідовності обробки даних, простоти користувацької взаємодії та ефективності реалізації поставленого завдання. Загальна архітектура системи поділена на кілька функціональних підсистем, кожна з яких відповідає за окремий аспект роботи програми, а також передбачає наявність консолі користувача як основного інтерфейсу взаємодії.

Основними підсистемами програмної системи є: підсистема ініціалізації параметрів генерації, підсистема створення часової структури композиції, підсистема генерації музичних тактів та підсистема збереження згенерованого результату. Кожна підсистема виконує свої власні функції, однак вони тісно пов’язані між собою послідовністю обробки даних.

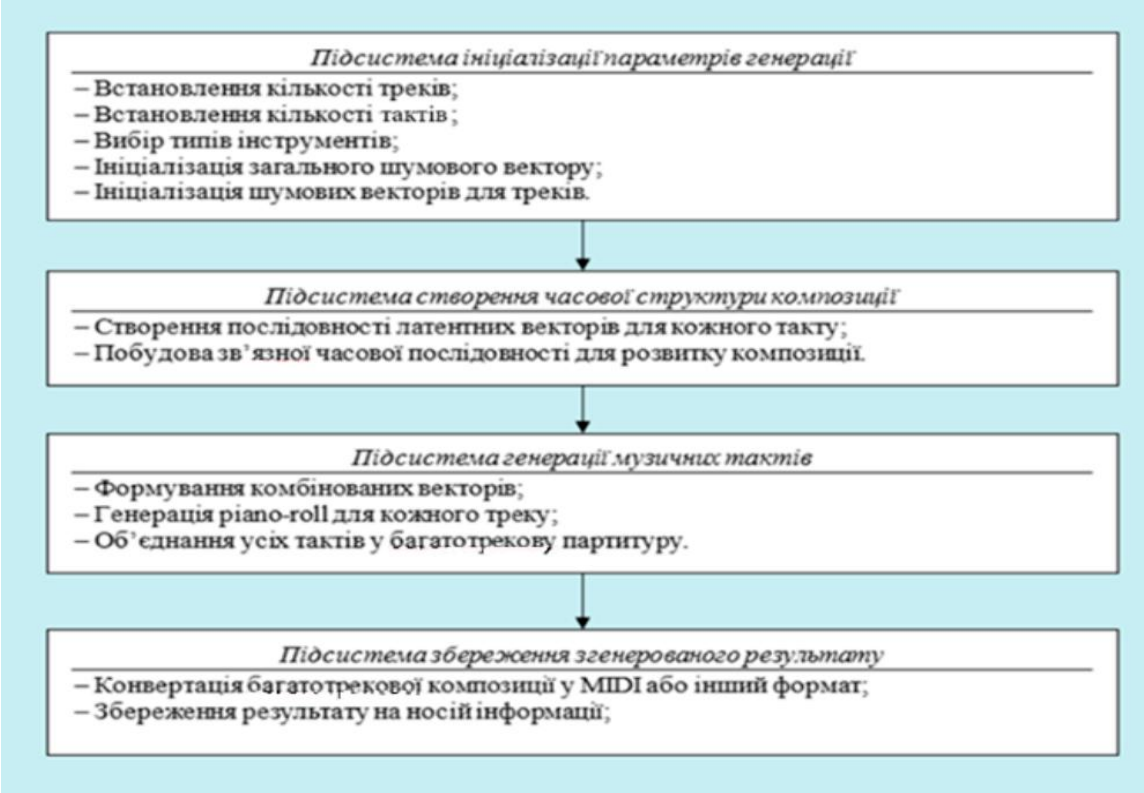


Рисунок 2.4 – Схема інформаційної системи створення музикальних композицій

Підсистема ініціалізації параметрів генерації забезпечує запит та обробку вхідної інформації від користувача, зокрема кількість тактів композиції, кількість інструментальних треків, вибір типів інструментів для кожного треку, а також формування початкових латентних шумових векторів. Ця підсистема є початковим етапом роботи всієї системи, оскільки її результатом є формування базових параметрів, необхідних для подальшої генерації музичних фрагментів.

Підсистема створення часової структури композиції відповідає за побудову послідовності латентних векторів для кожного такту на основі загального шумового вектора. Згенерована часова структура визначає логіку розвитку композиції у часі та забезпечує зв’язність між окремими частинами твору.

Підсистема генерації музичних тактів виконує безпосереднє створення окремих музичних фрагментів для кожного інструментального треку і кожного такту. На основі комбінованого векторного представлення (загальний вектор + вектор треку + вектор часу) система формує повну багатотрекову партитуру, що містить всю інформацію про ноти, їх висоти, тривалість та час початку.

Підсистема збереження згенерованого результату забезпечує конвертацію отриманих даних у зручний формат для подальшого використання. Згенерована композиція може бути збережена у форматі MIDI або у вигляді графічного зображення piano-roll. Також можливий вивід текстового подання композиції безпосередньо у консоль для попереднього перегляду.

На рівні логіки взаємодії компоненти системи пов’язані один із одним у строго визначеній послідовності. Завдання кожної підсистеми полягає у підготовці даних для наступного етапу обробки. Отже, процес починається з ініціалізації вхідних даних, продовжується побудовою часової структури, після чого відбувається генерація музичних тактів, і завершується процесом збереження згенерованої композиції.

Розроблена структура програмної системи дозволяє забезпечити чітку організацію всіх етапів генерації музики, спрощує процес управління даними та

робить систему гнучкою для можливого подальшого розширення її функціональності.

2.5 Структура датасету

Розроблений метод генерації багатотрекової музики базується на концепції обробки випадково згенерованих латентних векторів, які створюються безпосередньо в оперативній пам'яті під час виконання алгоритму. Вхідні дані для генерації композицій формуються на основі випадкових шумових векторів без використання попередньо збережених наборів даних або історії попередніх сесій. Генерація музичних творів відбувається на основі моделей машинного навчання, де кожна сесія є самостійним процесом, що починається з нуля, без необхідності звертання до зовнішніх сховищ чи збережених станів. Отримані результати у вигляді багатотрекових партитур одразу експортуються у зовнішні файли, такі як MIDI або графічні зображення piano-roll, що дозволяє уникнути необхідності у використанні баз даних для зберігання створених композицій, адже система функціонує в режимі "згенеровано – збережено", а всі проміжні етапи обробки виконуються динамічно у пам’яті програми. Відсутність потреби в підтримці складних транзакцій, унікальної ідентифікації об'єктів, забезпеченні зв’язків між різними сутностями або створенні запитів на вибірку даних робить використання бази даних у рамках даного проєкту недоцільним. З урахуванням цього, архітектура розроблюваної системи була спроєктована так, щоб забезпечити ефективне функціонування без інтеграції з будь-якими СКБД, що також сприяє спрощенню загальної структури застосунку, зменшенню вимог до ресурсів та підвищенню стабільності роботи програми під час виконання завдань генерації музичних композицій.

2.6 Підготовка робочих вхідних даних для системи

У межах реалізації даної кваліфікаційної роботи метод генерації багатотрекових музичних композицій базується переважно на обробці випадкових латентних векторів, які створюються безпосередньо в процесі роботи системи. Тому в базовій конфігурації система функціонує автономно, без обов'язкового використання зовнішніх датасетів. Це надає рішенню високу гнучкість, оскільки кожна сесія генерації є унікальною і не залежить від заздалегідь підготовлених наборів даних. Проте для забезпечення можливості навчання, тестування, а також експериментального порівняння результатів роботи методу за контрольними зразками допускається використання спеціалізованих датасетів або заздалегідь навчених моделей які містять структуровані приклади багатотрекової музики.

В якості джерела даних у дослідженні використовується датасет, який складається із символічних музичних композицій, представлених у форматі piano-roll. Piano-roll у цьому контексті є матричним представленням музичного твору, де вісь X відповідає часу (поділеному на дрібні кванти), вісь Y – висоті ноти, а значення елементів матриці вказують на наявність або відсутність звучання відповідної ноти у певний момент часу. Такий формат є зручним для обробки нейронними мережами та іншими алгоритмами машинного навчання, оскільки забезпечує просту бінарну структуру даних. Кожен елемент датасету має вигляд чотиривимірного тензора, який дозволяє гнучко оперувати даними на всіх етапах обробки: від тренування моделей до оцінювання результатів генерації.

Навчальний датасет складається з великої кількості MIDI-файлів у форматі piano-roll, а також розбитий на деякі інструменти (бас, гітара, барабани, піаніно, струнні). Кожна музична композиція конвертована у вигляді бінарних тензорів, де одиниця – нота звучить, нуль – нота мовчить у певний момент часу. Час розбитий на окремі кроки, а саме 96 кроків в одному музичному такті. Висота нот охоплює 84 ноти від C1(до першої октави) до C8(до 8 октави). Отже, структура одного

прикладу в датасеті – це чотири музичні такти, представлені як чотиривимірний тензор розміром 4 такти на 96 кроків на 84 ноти на 5 треків.

Структура даних у датасеті уніфікована та нормалізована: усі композиції мають однакову кількість тактів і часових поділів, що дозволяє зберігати однорідність навчальної вибірки. Також, у межах одного зразка всі інструментальні треки мають однакову тривалість і синхронізовану часову розгортку. Інструменти, що використовуються в датасеті, класифіковані за типами ,що дозволяє системі зберігати чітку логічну структуру композицій навіть при випадковій генерації нових треків. Для формування власного набору даних або експериментальної вибірки система також передбачає можливість збереження результатів генерації у форматі MIDI. Згенеровані композиції можуть бути перетворені у новий датасет для подальшого навчання, аналізу або розвитку моделі.

Щодо прикладів даних у датасеті, один зразок може містити, наприклад, композицію з 4 тактів, де кожен такт розділений на 96 часових поділів. При цьому для кожного інструментального треку, скажімо для басу, на кожен поділ часу може бути зафіксовано подію гри конкретної ноти, наприклад G2 або C3. Наявність події кодується одиницею, відсутність – нулем. Завдяки цьому, дані мають дуже компактне, але водночас інформативне подання, яке легко інтерпретувати як для машинної обробки, так і для подальшої конвертації у музичний формат.

Загалом, підхід до організації даних у межах запропонованого методу спрямований на забезпечення максимальної гнучкості: система здатна працювати як у режимі автономної генерації без зовнішніх даних, так і з використанням підготовлених датасетів для потреб дослідження або удосконалення моделей. Така архітектура даних дозволяє комбінувати різні сценарії використання без необхідності зміни основних алгоритмів роботи системи.

2.7 Особливості використання спеціалізованих програмних компонентів

Для реалізації розробленого методу генерації багатотрекових музичних композицій передбачається використання низки бібліотек, що забезпечують підтримку необхідних алгоритмів обробки даних, генерації латентних векторів, побудови нейронних мереж та роботи з музичними форматами. Основна задача підбору бібліотек полягає в тому, щоб оптимізувати процес розробки, скоротити час на реалізацію базових алгоритмів та забезпечити відповідність розроблюваної системи сучасним стандартам роботи з нейронними мережами та даними у форматі музичних партитур.

Однією з основних бібліотек, яка використовується у межах даного проєкту, є бібліотека TensorFlow [43]. Вона є однією з найпоширеніших та найпотужніших платформ для реалізації моделей глибокого навчання. TensorFlow забезпечує зручний інтерфейс для створення та тренування нейронних мереж, підтримує автоматичне диференціювання, ефективну роботу з тензорами та апаратне прискорення за допомогою графічних процесорів. Завдяки TensorFlow реалізовано архітектуру генеративно-змагальної мережі, що включає генератор часової структури, генератор тактів та дискримінатор. Основними перевагами використання TensorFlow є висока продуктивність, підтримка масштабованості, можливість розгортання моделей на різних платформах, а також велика кількість документації, навчальних матеріалів і прикладів, що значно полегшують процес розробки.

Для роботи із музичними форматами, зокрема для конвертації згенерованих piano-roll структур у формат MIDI, планується використання бібліотеки pretty_midi [44]. Ця бібліотека надає прості засоби для створення, збереження та обробки MIDI-файлів у Python. Вона дозволяє легко трансформувати масиви нотних подій у стандартний MIDI-формат, що забезпечить сумісність результатів роботи системи із популярними цифровими аудіо робочими станціями та програмами для редагування музики. Через

можливості pretty_midi планується реалізувати модуль збереження згенерованих композицій, а також опціонально – функцію візуалізації нотного вмісту.

Для обробки та базової роботи з багатовимірними масивами даних у системі передбачено використання бібліотеки NumPy [45]. NumPy є стандартною бібліотекою для роботи з числовими обчисленнями в Python. З її допомогою будуть реалізовані операції маніпуляції тензорами, підготовка латентних векторів для генерації, а також попередня обробка даних перед передачею їх у нейронну мережу.

Окрім цього, для створення графіків, візуалізації результатів роботи моделі та подання статистичних даних планується використовувати бібліотеку Matplotlib [46]. Вона надає можливості побудови як простих графіків (лінійних, гістограм, розподілів), так і більш складних візуалізацій, необхідних для аналізу роботи мережі, наприклад, для побудови графіків втрат під час навчання або для візуального представлення сіток piano-roll.

Отже, сукупність використаних бібліотек спрямована на забезпечення повного циклу обробки даних: від генерації латентних векторів до збереження готової музичної композиції у стандартних форматах. Застосування перевірених та надійних інструментів дозволяє зосередитися безпосередньо на науковій новизні методу та підвищити якість реалізації програмної системи загалом.

2.8 Висновки до розділу 2

На основі проведеної роботи в другому розділі було розроблено метод створення музичних композицій на основі генеративного штучного інтелекту, а також сформовано комплексне уявлення про архітектуру, функціональні можливості та реалізаційні особливості інформаційної системи, призначеної для генерації багатотрекових музичних композицій у символічному форматі. Метод, що ліг в основу системи, побудований на поетапному створенні музичної партитури з використанням генеративних моделей штучного інтелекту, а саме – генеративно-змагальної нейронної мережі. Ключова увага була зосереджена на

забезпеченні логічної послідовності етапів: від ініціалізації параметрів до збереження результату, що дозволило створити узгоджену архітектуру, в якій кожна підсистема чітко виконує свою роль і передає оброблені дані наступній.

Функціональна структура системи передбачає наявність чотирьох основних груп задач: визначення вхідних параметрів генерації, побудова часової структури композиції, формування музичних тактів та збереження результату. Кожна з цих частин системи реалізована як окрема підсистема з чітко визначеними завданнями, що дозволяє легко масштабувати або модифікувати її при подальшому розвитку. Завдяки чітко продуманій послідовності та логіці взаємодії, система демонструє стабільність і передбачуваність результатів.

Важливою складовою підготовки до реалізації стала робота з даними. Було доведено, що система здатна працювати як у повністю автономному режимі, так і з використанням готових датасетів, побудованих на основі представлення музики у форматі piano-roll. Це забезпечує не лише гнучкість, а й потенціал до подальшого навчання або тонкого налаштування моделі під конкретні стилі чи задачі. Детальний аналіз структури датасетів, особливостей форматування та прикладів демонструє відповідність даних технічним вимогам нейронної архітектури, зокрема їх уніфікованість, нормалізацію та синхронізовану часову сітку.

З програмної точки зору, для реалізації методу було обрано перевірений стек технологій. Основним рушієм для роботи з нейронними мережами став фреймворк TensorFlow, який забезпечив гнучкість і продуктивність на всіх етапах моделювання. Для роботи з музичними даними використано бібліотеку pretty_midi, яка дозволила перетворювати piano-roll у MIDI-файли. Також до складу інструментів увійшли NumPy – для обчислень і тензорної арифметики, та Matplotlib – для графічної візуалізації результатів. Комбінація обраних бібліотек дозволила створити повноцінний функціональний цикл генерації, з урахуванням усіх вимог до якості, ефективності та зручності розробки.

Розділ 3 Експериментальне дослідження методу та програмна реалізація інформаційної системи

3.1 Визначення шляхів дослідження та засобів створення програмного забезпечення

Запропонований метод генерації музичних композицій на основі генеративного штучного інтелекту було реалізовано у вигляді програмної системи, що дозволяє генерувати музичні композиції як на основі заданого треку під час навчання, так і з чистого листа і дозволяє зберігати результат у форматі MIDI. Система дозволяє задавати базові параметри композиції (кількість тактів, кількість треків, типи інструментів), після чого на основі генеративної моделі формуються окремі музичні фрагменти, які потім об’єднуються у цілісну партитуру. Для оцінки ефективності методу буде виконано низку експериментів, спрямованих на вивчення характеристик згенерованих музичних творів. Генерація великої кількості прикладів (наприклад, 1000–5000 фрагментів) дозволяє аналізувати статистичні властивості результатів на основі формальних метрик. Серед основних параметрів дослідження передбачається оцінка: кількості порожніх тактів (Empty Bars), що дозволяє оцінити активність композицій; кількості унікальних висот (UPC – Used Pitch Classes), що показує різноманітність мелодії; відсотку правильних нот (QN – Qualified Notes), які тривають достатньо довго, щоб мати музичне значення; частки нот, що вкладаються в характерні ритмічні структури (DP – Drum Pattern); а також метрики міжтрекової узгодженості, зокрема тональної дистанції (TD – Tonal Distance), яка відображає ступінь гармонійної відповідності між інструментальними треками.

3.2 Вибір засобів розробки інформаційної системи

Для реалізації програмної системи генерації багатотрекових музичних композицій використано консольну архітектуру застосунку. Основною мовою програмування обрано Python, оскільки вона має розвинену екосистему бібліотек

для машинного навчання, обробки MIDI-даних та математичного аналізу. Python дозволяє ефективно реалізовувати генеративні моделі та обробляти великі масиви даних із мінімальними витратами часу на розробку.

В якості середовища розробки використано Visual Studio Code – кросплатформенний редактор програмного коду з розширеною підтримкою Python. VS Code забезпечує зручність налагодження, автодоповнення коду, інтеграцію з системою контролю версій Git, а також роботу з віртуальними середовищами. Цей редактор не потребує значних ресурсів, швидко завантажується та підтримує численні розширення, що робить його придатним для дослідницької роботи.

Генеративна модель реалізована з використанням фреймворку TensorFlow, який є одним із найпоширеніших засобів для розробки моделей глибокого навчання. TensorFlow забезпечує гнучкий та масштабований підхід до побудови обчислювальних графів, підтримує як імперативний (через API Keras), так і декларативний стилі програмування, що дозволяє ефективно реалізовувати як прототипи, так і повноцінні рішення. Крім того, TensorFlow забезпечує продуктивне виконання на різних апаратних платформах, включаючи CPU та GPU, що дає змогу адаптувати систему до обчислювальних ресурсів виконавця.

Для роботи з MIDI-файлами було обрано бібліотеку miditoolkit, яка надає зручний інтерфейс для роботи з музичними подіями, підтримку piano-roll представлення та повноцінне збереження композицій у форматі MIDI. У порівнянні з іншими бібліотеками (mido, pretty_midi), miditoolkit дозволяє швидше реалізувати обробку багатотрекових партитур, що є ключовою вимогою у межах цього дослідження.

Таким чином, для реалізації системи використано наступну комбінацію засобів:

- мова програмування: Python
- середовище розробки: Visual Studio Code
- фреймворк ШІ: Tensorflow
- бібліотека для роботи з MIDI: miditoolkit

Обрана конфігурація є достатньою для реалізації поставленої мети кваліфікаційної роботи: генерації, збереження та подальшого аналізу багатотрекових музичних фрагментів без залучення серверних рішень чи складної інфраструктури.

3.3 Структура та функціональне призначення програмних складових системи

Розроблена система генерації багатотрекових музичних композицій побудована за модульною архітектурою, що забезпечує розділення функціональних компонентів та спрощує супровід і тестування коду. Система має консольний інтерфейс і складається з низки окремих логічних модулів, кожен із яких відповідає за окремий етап генерації композиції – від ініціалізації до збереження. Основними модулями є config.py, latent_initializer.py, generator.py, merger.py, midi_saver.py, metrics.py,

Модуль config.py виконує роль централізованого конфігураційного блоку, де зберігаються всі основні параметри, що визначають поведінку системи. У цьому файлі зосереджено змінні, які визначають кількість тактів у композиції, кількість треків, перелік інструментів, а також шляхи до збереження результатів і завантаження моделі. Завдяки цьому модулю забезпечується гнучкість налаштувань без потреби вносити зміни в логіку інших модулів. Зокрема, користувач може швидко змінити кількість треків, обрати інші інструменти або змінити місце збереження згенерованих файлів, що особливо зручно на етапі експериментування та тестування.

Модуль latent_initializer.py відповідає за створення початкових латентних векторів, які є вхідними даними для генеративної моделі. Ці вектори формуються як випадкові тензори певної розмірності, що визначена в моделі. Для кожного треку генерується окремий латентний вектор, що дозволяє забезпечити варіативність та незалежність у структурі майбутньої композиції. Ініціалізація відбувається з використанням генератора випадкових чисел, і цей процес є цілком

автоматизованим. Завдяки цьому модуль забезпечує унікальність кожного згенерованого треку, але при цьому не враховує музичний контекст чи попередні зразки, що залишає простір для подальшого удосконалення.

Центральним усьому проєкту є модуль generator.py, який реалізує логіку взаємодії з навченою нейронною мережею. Саме тут здійснюється завантаження моделі з файлу, передача до неї латентних векторів і отримання у відповідь згенерованих фрагментів музики у вигляді piano-roll матриць. Модуль написано з використанням фреймворку TensorFlow, і він працює незалежно від конкретної архітектури моделі, що дозволяє легко оновлювати чи замінювати її за потреби. Згенеровані дані передаються на подальшу обробку, але вже на цьому етапі формується основа майбутньої композиції. У випадку зміни архітектури моделі або її параметрів, саме цей модуль найбільш чутливий до таких змін.

Модуль merger.py виконує важливу функцію синхронізації та структурування треків, які були згенеровані моделлю. Кожен трек окремо може мати різну довжину або початкову позицію, тому цей модуль виконує вирівнювання по тактах і узгоджує параметри, такі як темп та кількість нот у кожному інструменті. Крім того, модуль об'єднує треки в загальну композицію, при цьому відкидаючи порожні або надлишкові партії. Його роль полягає в упорядкуванні результатів генерації та формуванні їх у структуру, що відповідає форматам, прийнятим у цифровій музиці. Важливо зазначити, що цей етап суттєво впливає на сприйняття композиції, адже від нього залежить, наскільки треки будуть узгоджені між собою.

Модуль midi_saver.py відповідає за кінцевий етап – збереження композиції у вигляді MIDI-файлу. Цей модуль працює з бібліотекою miditoolkit і конвертує структури piano-roll у стандартний формат MIDI, який сумісний з більшістю музичних цифрових станцій. Для кожного інструменту створюється окремий MIDI-трек, у якому точно задаються висоти, тривалості нот, час початку звучання, а також канал і назва інструмента. Отже, результат генерації перетворюється у форму, що легко використовується музикантами та композиторами для подальшої

роботи. Хоча MIDI не містить реальних звуків, він зберігає всю необхідну інформацію для точної інтерпретації композиції.

Модуль metrics.py реалізує функції оцінки якості згенерованої музики. Він обчислює низку формальних музичних метрик, які дозволяють аналізувати композиції з точки зору ритмічної наповненості, гармонічної узгодженості та загального музичного змісту. Серед таких метрик – кількість порожніх тактів (Empty Bars), кількість використаних висот (Used Pitch Classes), якість нот (Qualified Notes), ритмічна точність барабанних партій та тональна дистанція між треками. Ці метрики використовуються як для внутрішнього тестування, так і для кількісного порівняння різних результатів генерації. Вони дозволяють не тільки оцінити технічну якість композицій, а й робити висновки про загальний потенціал моделі.

Нарешті, модуль main.py виконує координуючу функцію та слугує точкою входу для запуску всієї системи. Саме тут реалізована послідовність виклику всіх інших модулів: ініціалізація конфігурації, генерація латентних векторів, обробка результатів, об'єднання треків, збереження файлу та розрахунок метрик. У такий спосіб main.py забезпечує цілісність процесу генерації та дозволяє користувачеві запустити весь пайплайн за допомогою одного виклику.

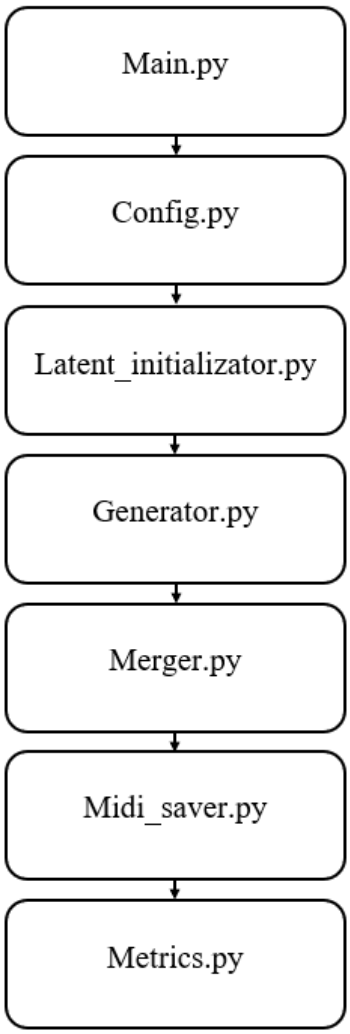


Рисунок 3.1 – Схема послідовності роботи модулів

Отже, модуль main.py виконує координуючу функцію та слугує точкою входу для запуску всієї системи. Саме тут реалізована повна послідовність дій, що забезпечує повний цикл генерації музичної композиції. Після завантаження конфігурації з config.py модуль викликає latent_initializer.py для створення латентних векторів, які потім передаються до generator.py для генерації музичних треків за допомогою навченої нейронної мережі. Згенеровані треки обробляються та синхронізуються у merger.py, після чого результат записується у форматі MIDI через midi_saver.py. Завершальним етапом є оцінка згенерованої композиції, яка виконується за допомогою metrics.py, що, після аналізу MIDI-файлу видає показники пустих тактів, кількість унікальних висот, частка якісних нот, відповідність ритмічним патернам, тощо.

3.4 Особливості реалізації програмних складових системи

У процесі реалізації програмної системи генерації багатотрекової музики ключовим завданням стало впровадження послідовної архітектури, що забезпечує логічний поділ етапів генерації, гнучке налаштування параметрів, обробку результатів та оцінювання якості. Однією з найважливіших складових системи є модуль генерації generation.py, який реалізує алгоритм створення музичного фрагменту на основі попередньо навченого латентного простору. Після ініціалізації генератор отримує на вхід набір параметрів, зокрема кількість тактів, треків та інструментів, і формує відповідний латентний вектор. Цей вектор проходить через модель, яка відтворює структуру музичного твору у вигляді багатотрекового масиву. Ключовим моментом реалізації є контроль над збереженням логічної структури композиції при варіації вхідних параметрів – наприклад, збереження ритмічного малюнку або гармонічної відповідності між треками.

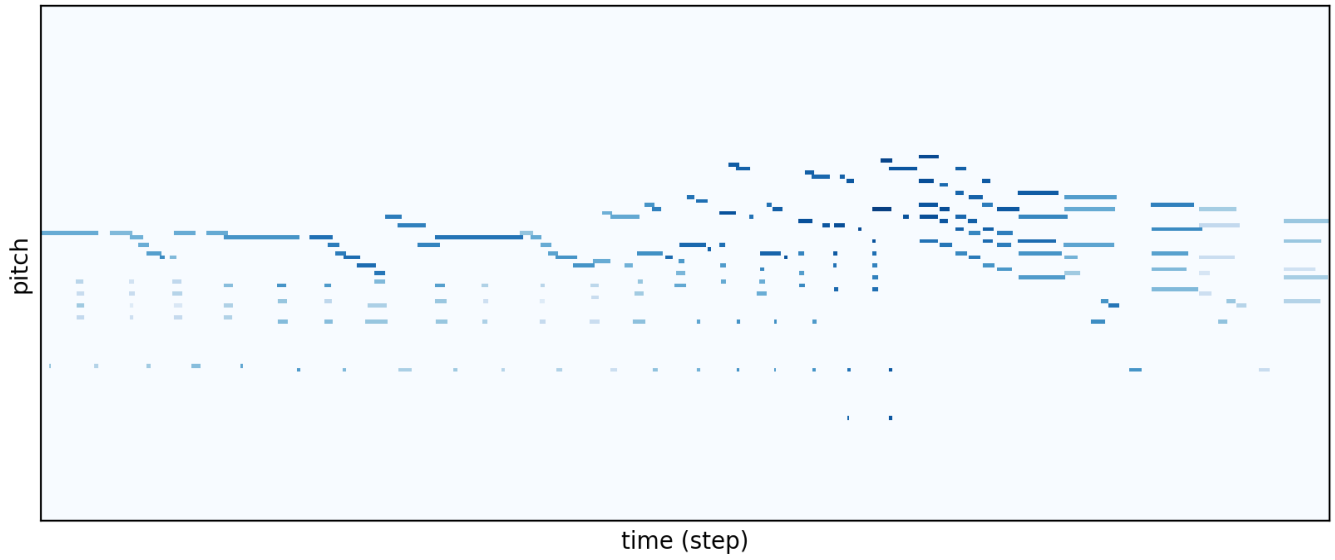


Рисунок 3.2 – Приклад piano-roll

Іншим важливим модулем є visualization.py, який відповідає за графічне представлення згенерованих музичних треків. Його реалізація базується на

бібліотеках matplotlib і pypianoroll, що дає змогу побудувати візуалізацію типу "piano roll", де вісь X відповідає за часовий проміжок, в якому нота звучить, а вісь Y відповідає за висоту ноти. Основною задачею стало створення інформативного зображення, яке дає змогу швидко оцінити структуру музики: наявність ритму, повторів, гармонії між треками. У процесі реалізації важливим було обрати кольорову палітру для кожного треку, щоб чітко розрізняти інструменти, а також забезпечити відображення тактів, щоб аналізувати періодичність і логічну побудову композиції. На рисунку 3.3 зображено саме комбінацію piano-roll з декількох треків. На зображенні видно висоту ноти(її тональність), а також скільки в секундах вона звучить.

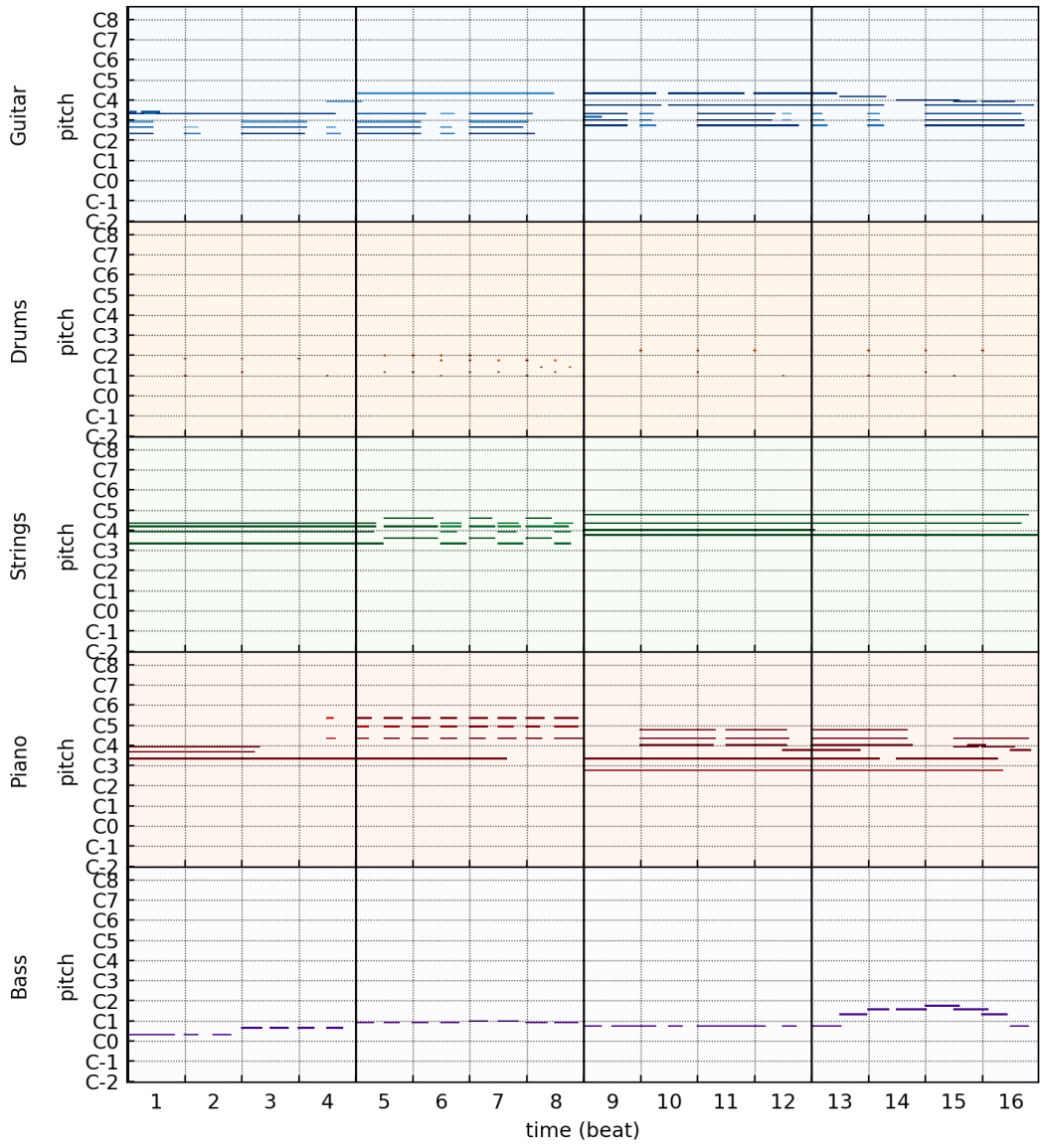


Рисунок 3.3 – Piano-Roll з кількома треками

Особливе місце займає модуль config.py, який відповідає за централізоване управління параметрами генерації музики. У цьому модулі зібрані всі основні налаштування, які можуть впливати на поведінку генератора: кількість тактів, кількість треків, типи інструментів, допустимий діапазон висот нот, тривалість нот, формат збереження тощо. Завдяки цьому підходу, усі параметри зосереджені в одному місці, що спрощує модифікацію й експерименти з різними конфігураціями. Реалізація передбачає зручне редагування налаштувань без необхідності змінювати код інших модулів. Крім того, файл config.py може бути використаний для створення серій автоматичних запусків із різними параметрами, наприклад, при тестуванні впливу кількості треків на метрики якості. Це дозволяє зручно управляти сценаріями дослідження та покращує відтворюваність результатів.

Реалізація модуля metrics.py потребувала вбудованих евристик для аналізу музичних властивостей композицій. Оскільки модель працює без нагляду, для оцінювання результату використовуються формальні метрики. Алгоритм перевіряє кожен MIDI-файл, формує структури треків та нот, після чого обчислює кількість порожніх тактів, середню кількість унікальних нот, середню тривалість нот, ритмічну узгодженість та тональну дистанцію між треками. Окремі метрики реалізовані як окремі функції, які викликаються послідовно, з результатами, що зберігаються в таблицях і графіках.

Усі етапи реалізації виконано в середовищі Visual Studio Code з використанням бібліотек pretty_midi, pypianoroll, numpy, matplotlib, що забезпечило високий рівень інтеграції між модулями та ефективність при обробці великих обсягів згенерованих даних. Особливістю реалізації стала гнучкість – кожен модуль працює незалежно і може бути протестований окремо, що значно полегшило налагодження та внесення змін.

3.5 Аналіз функціональності системи

Інформаційна система для генерації музики потребує певного налаштування перед її використанням. Для запуску програмного продукту користувачу необхідно мати попередньо встановлену операційну систему з підтримкою Python (версія 3.7 або вище) та доступ до терміналу. Найзручніше використовувати середовище розробки Visual Studio Code (VS Code), однак можна обійтись і звичайним терміналом. У процесі використання програми користувач повинен уміти запускати shell-скрипти, редагувати конфігураційні файли та інтерпретувати результати генерації. Інтерфейс програми не є графічним, вона функціонує через командний рядок.

Першим кроком є встановлення всіх необхідних залежностей. Для цього рекомендується скористатися pipenv, хоча можна також встановити пакети за допомогою pip. Якщо pipenv ще не встановлено, його можна додати за допомогою команди pip install pipenv. Далі слід перейти до кореневої директорії проєкту та виконати pipenv install для встановлення усіх залежностей, а потім – pipenv shell, щоб активувати віртуальне середовище. Альтернативно, користувач може виконати команду pip install -r requirements.txt, яка встановить необхідні залежності, а саме setuptools, absl-py, astor, gast, grpcio, imageio, mardown, mido, numpy, pillow, pretty-midi, protobuf, pypianoroll, pyYAML, scipy, sharedArray, six, tensorboard, tensorflow-gpu ,termcolor ,werkzeug.

Наступним етапом є підготовка навчальних даних. У цьому проєкті використовується Lakh Pianoroll Dataset (LPD) – велика колекція багатотрекових піанаролів. Дані можна автоматично завантажити за допомогою скриптів ./scripts/download_data.sh, а потім обробити для подальшого використання командою ./scripts/process_data.sh. Дані зберігаються у форматі .npz, що містить лише індекси ненульових елементів та форму масиву – це дозволяє економити простір. Якщо користувач бажає підготувати власні дані, можна скористатися скриптом collect_data.sh, передавши шлях до папки з MIDI-файлами та бажаний шлях збереження (data/train.npy).

Після підготовки даних користувач може створити новий експеримент. Для цього запускається скрипт ./scripts/setup_exp.sh, передаючи йому шлях до нової директорії експерименту та короткий опис, наприклад: ./scripts/setup_exp.sh "./exp/my_experiment/" "Мій перший експеримент із генерацією музики".
У цій директорії буде створено файл конфігурації config.yaml та файл параметрів моделі. Ці файли користувач має відредагувати відповідно до власних потреб: можна задати кількість треків, тривалість нот, частоту збереження результатів тощо. Конфігурація здійснюється вручну через змінні у файлі.

```yaml
# Experiment
n_jobs: 20
log_loss_steps: 100 # set to 0 to disable loss logging
save_samples_steps: 100 # set to 0 to disable saving samples
save_summaries_steps: 0 # set to 0 to disable saving summaries
save_checkpoint_steps: 10000 # set to 0 to disable saving checkpoints
evaluate_steps: 100 # set to 0 to disable evaluation

# Data
data_source: 'sa' # 'sa', 'npy', 'npz'
data_root: ~
data_filename: train_x_lpd_5_phr
use_random_transpose: false # randomly transpose the piano-rolls at training
use_train_test_split: false # split the piano-rolls at training
```

Рисунок 3.6 – Приклад налаштування конфігурації у файлі config.yaml

Після цього користувач може або запустити лише навчання моделі (./scripts/run_train.sh "./exp/my_experiment/" "0"), або повноцінний експеримент, що включає навчання, генерацію та інтерполяцію (./scripts/run_exp.sh "./exp/my_experiment/" "0"). Під час навчання система буде автоматично зберігати проміжні результати, а також згенеровані приклади музики у трьох форматах: .npy – сирі масиви NumPy, .png – візуалізація піанороліів, .npz – формат багатотрекових піанороліів, сумісний із бібліотекою Pypianoroll. Зберігання цих форматів можна відключити, редагуючи відповідні параметри у config.yaml: save_array_samples, save_image_samples, save_pianoroll_samples.

У разі, якщо користувач не бажає тренувати модель самостійно, він може скористатися вже навченими моделями. Для цього потрібно завантажити попередньо підготовлені ваги за допомогою скрипту ./scripts/download_models.sh, або вручну завантажити архів pretrained_models.tar.gz і розпакувати його. Далі можна виконати генерацію (інференс) з готової моделі: ./scripts/run_inference.sh"./exp/default/" "0" або виконати інтерполяцію між музичними зразками: ./scripts/run_interpolation.sh "./exp/default/" "0". Результати генерації зберігаються у директоріях експерименту.

Загалом, користувач повинен пройти послідовно наступні етапи: встановлення залежностей, підготовка або завантаження навчальних даних, ініціалізація нового експерименту, редагування конфігурації, запуск тренування або генерації, аналіз та конвертація результатів.

3.6 Результати досліджень

Першим експериментом є дослідження ефективності генеративної моделі – динаміка формування музичного матеріалу в залежності від кількості кроків оновлення. Для оцінки якості роботи моделі було проведено експеримент, під час якого вона створювала фортепіанні партії "з нуля", тобто без надання будь-якого вхідного шаблону чи музичного фрагменту. Це дозволяє побачити, як модель формує музику, починаючи з порожнього полотна і поступово додаючи структуровані елементи. План дослідження полягав у тому, щоб зафіксувати результати генерації після певної кількості кроків оновлення. Було обрано шість контрольних точок: 0, 700, 2500, 6000 та 7900 кроків. На кожному з цих етапів модель зупинялась, і результати її роботи зберігались у вигляді фортепіанних ролів – спеціального графічного представлення музики, де вісь X відповідає часу, а вісь Y – висоті нот.

У ході експерименту була використана модель, яка на кожному з певних етапів тренування зберігалась, а потім – без додаткового навчання –

використовувалась для повної генерації нового музичного фрагмента з нуля. Завдяки цьому ми можемо простежити поступовий розвиток моделі, її здатність структурувати звук, підтримувати музичну логіку та уникати хаотичних рішень.

Дослідна програма включала наступні етапи:

– генерація фортепіанного ролу одразу після початку навчання (Step 0);
– повторна генерація після 700 кроків тренування;
– аналіз результату на 2500 кроці;
– подальше спостереження на 6000 кроці;
– завершальний аналіз на 7900 кроці.

На початковому етапі, до будь-якого ефективного навчання, модель ще не сформувала жодного уявлення про структуру музики. Відповідно, згенерований рол має хаотичний вигляд, набір звуків випадковий, не має ні ритміки, ні гармонії. Це відповідає очікуванням – початкові ваги ще не адаптовані до музичних закономірностей.

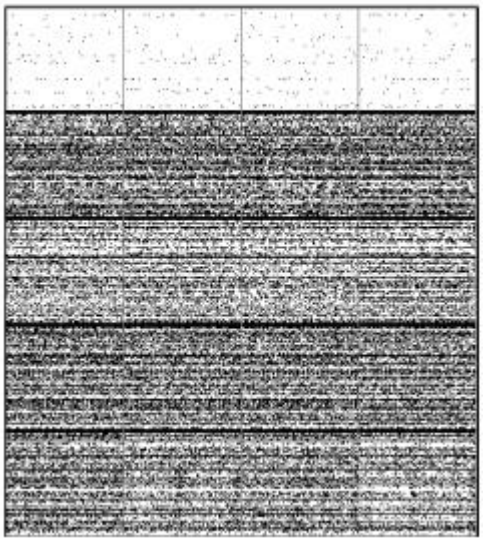


Рисунок 3.7 – Piano-roll крок 0

Після 700 ітерацій тренування модель починає демонструвати перші ознаки структурованості. Видно спроби створити ритм, окремі нотні шаблони повторюються. Втім, загальна якість все ще низька – багато нелогічних переходів, великі прогалини між звуками.

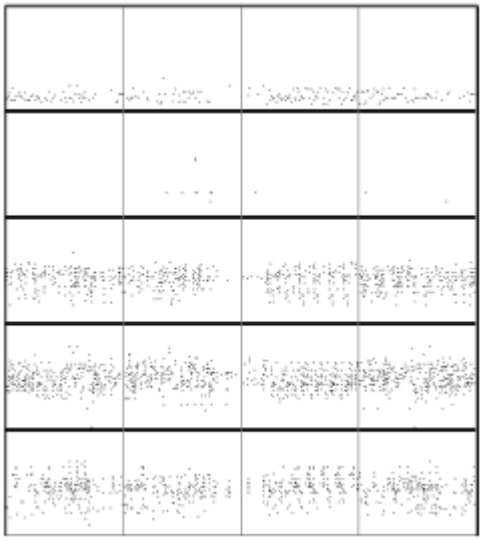


Рисунок 3.8 – Piano-roll крок 700

На цьому етапі стає очевидною перша музична логіка: інструменти починають взаємодіяти, з’являються ритмічні повторення, акордові структури, які нагадують музичні фрази. Прогрес у порівнянні з попереднім кроком є суттєвим, хоча ще присутні певні недосконалості – надмірні паузи або зайві ноти.

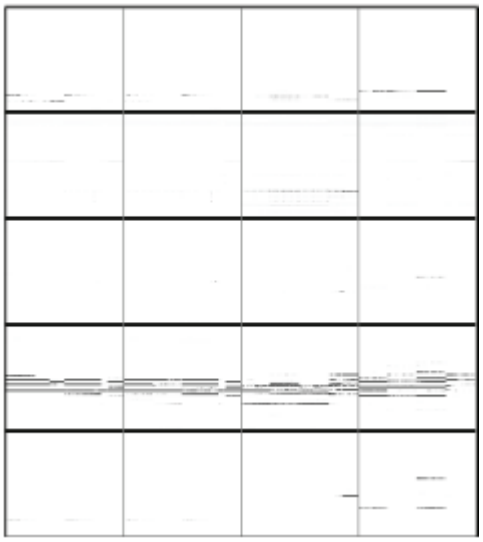


Рисунок 3.9 – Piano-roll крок 2500

У наступному кроці помічається значне покращення. Структура майже повноцінна, присутній чіткий музичний розмір, ноти гармонують одна з одною. Барабани, бас, акорди та мелодійна лінія синхронізуються між собою. Це демонструє, що модель уже навчилася відтворювати основні закономірності з

навчальних прикладів і цього достатньо для створення повноцінної композиції, але все ще не відповідає можливостям написання композиції людиною.

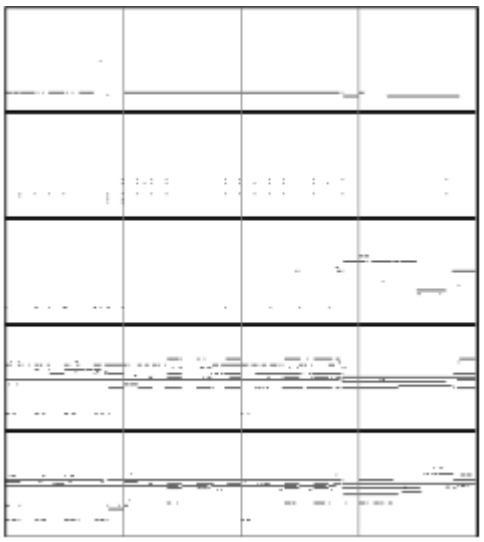


Рисунок 3.10 – Piano-roll крок 6000

На фінальному етапі навчання результат максимально наближений до людського виконання. Модель зберігає як ритмічну, так і гармонійну послідовність, кожен інструмент "грає свою роль", і загальне звучання є цілісним. Це свідчить про ефективність запропонованого архітектурного рішення і правильно обраного набору навчальних даних.

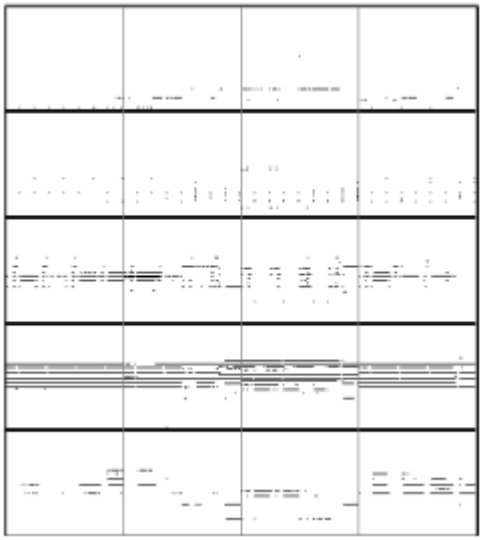


Рисунок 3.11 – Piano-roll крок 7900

Експеримент доводить, що навіть у рамках відносно короткого навчального циклу, модель GAN демонструє значне зростання якості генерації. Від повного хаосу до впізнаваної музичної структури, що імітує стилі з навчального датасету. Такий підхід дозволяє не лише згенерувати нову музику, але й вивчити, як ШІ поступово "розуміє" музичні правила.

Також проведено експеримент для порівняння, яка з моделей робить якіснішу музику. Основна мета експерименту – оцінити, наскільки моделі здатні імітувати ключові аспекти музичної логіки, гармонії, ритміки та структурованості на рівні як окремих треків, так і їхньої узгодженості між собою. Для цього було застосовано кілька метрик, що охоплюють внутрішньотрекові (intra-track) характеристики, включаючи відсоток порожніх тактів (empty bars), кількість використовуваних класів висот нот (used pitch classes), частку кваліфікованих нот (qualified notes) та ритмічні шаблони ударних (drum patterns), а також тональну відстань (tonal distance) між треками як показник гармонійної узгодженості.

Таблиця 3.1 – Результати експерименту

		Empty bars (EB; %)					Used pitch classes (UPC)				Qualified notes (QN; %)				DP (%)
		B	D	G	P	S	B	G	P	S	B	G	P	S	D
Training data		8.06	8.06	19.4	24.8	10.1	1.71	3.08	3.28	3.38	90.0	81.9	88.4	89.6	88.6
From scratch	jamming	6.59	2.33	18.3	22.6	6.10	1.53	3.69	4.13	4.09	71.5	56.6	62.2	63.1	93.2
	composer	0.01	28.9	1.34	0.02	0.01	2.51	4.20	4.89	5.19	49.5	47.4	49.9	52.5	75.3
	hybrid	2.14	29.7	11.7	17.8	6.04	2.35	4.76	5.45	5.24	44.6	43.2	45.5	52.0	71.3
	ablated	92.4	100	12.5	0.68	0.00	1.00	2.88	2.32	4.72	0.00	22.8	31.1	26.2	0.0
Track-conditional	jamming	4.60	3.47	13.3	-	3.44	2.05	3.79	-	4.23	73.9	58.8	-	62.3	91.6
	composer	0.65	20.7	1.97	-	1.49	2.51	4.57	-	5.10	53.5	48.4	-	59.0	84.5
	hybrid	2.09	4.53	10.3	-	4.05	2.86	4.43	-	4.32	43.3	55.6	-	67.1	71.8

На таблиці 3.1 зображено результати для кількох режимів генерації: з нуля (from scratch) та умовної генерації за треком (track-conditional). У межах кожного режиму представлено три архітектурних варіанти моделей: jamming, де кожен трек генерується незалежно своїм генератором; composer, де один генератор створює всі треки разом; та hybrid, що комбінує обидва підходи, використовуючи

спільні та приватні входи. Додатково подано результати для варіанта ablated, який є обмеженою версією composer-моделі без нормалізації пакетів і виконує роль нижньої межі якості.

Показники empty bars демонструють, наскільки часто треки мовчать у межах тактів. Наприклад, у варіанті jamming кількість порожніх тактів є відносно невеликою і близькою до навчальних даних, що свідчить про активність усіх інструментів. Водночас у composer та hybrid модель порожніх тактів для ударних та фортепіано сягає понад 20–30%, що є значним відхиленням і може вказувати на слабшу ритмічну реалізацію. Ablated модель узагалі не генерує ніяких осмислених нот, майже всі її треки порожні.

Кількість використовуваних pitch classes демонструє мелодичне або акордове наповнення. У jamming моделі бас відтворює мінімум класів висот, як і у навчальних даних, що узгоджується з його монофонічною роллю. Треки акордів (піаніно, гітара, струнні) охоплюють ширший діапазон класів. Composer та hybrid, попри більшу кількість класів, втрачають частину відповідності навчальним даним, що може означати надлишок випадковості або недосконалу гармонійну логіку.

Qualified notes – показник фрагментованості музики. Чим більше таких нот, тим менше дрібних, безглуздих фрагментів. Навчальні дані мають високі значення (понад 80–90%), тоді як згенеровані дані мають значно нижчі результати, особливо у composer і hybrid моделях. Це свідчить про те, що моделі часто створюють надто короткі, нефізіологічні для сприйняття ноти, ймовірно через особливості бінаризації вихідних тензорів.

DP – це показник відповідності ударних поширеним ритмічним шаблонам. Jamming модель зберігає майже повну відповідність (понад 90%), тоді як composer і hybrid – нижчі. Ablated модель не генерує нічого осмисленого в ударних, що підтверджується 0% у цій метриці.

У частині track-conditional моделі отримують дещо інші результати. Загалом відзначається краща відповідність метрикам, зокрема у jamming варіанті, де ударні майже на рівні з навчальними даними, а кількість порожніх тактів

знижена. Hybrid і composer демонструють трохи меншу якість по ряду показників, але hybrid, попри свою гнучкість, майже не програє composer і навіть випереджає його за деякими пунктами.

Окрема таблиця присвячена міжтрековій оцінці – tonal distance, що вимірює гармонійну узгодженість між треками. Чим менше значення, тим вища гармонійність. Навчальні дані мають значення 1.5–1.6 між мелодійними й акордовими треками, і менше – між акордовими треками. Ablated модель тут відсутня, оскільки не генерує гармонійних структур. Composer і hybrid моделі мають значно нижчі значення TD, що свідчить про кращу міжтрекову гармонійну взаємодію. У jamming моделі значення ближчі до навчальних, що показує її слабшу інтеграцію треків, оскільки вона не моделює міжтрекові залежності.

Загалом експеримент підтверджує, що hybrid модель досягає балансу між гнучкістю і якістю, composer демонструє хорошу гармонійну інтеграцію, але гірше справляється з ритмічним наповненням та кількістю якісних нот, а jamming краще працює на рівні окремих треків, але страждає у плані міжтрекової узгодженості. Ablated модель є неконкурентоспроможною, що свідчить про критичну важливість повноцінної архітектури з нормалізацією.

3.7 Висновки до розділу 3

Результатом реалізації методу генерації багатотрекових музичних композицій стало створення повноцінної інформаційної системи, яка поєднує в собі всі необхідні компоненти для автономного створення, обробки, оцінювання та збереження музичних творів у символічному форматі. Програмна система була побудована з урахуванням модульної архітектури, що дозволило не лише логічно структурувати етапи генерації, а й забезпечити високу гнучкість, масштабованість і зручність у налагодженні та подальшому розвитку. Її центральною ідеєю стало поетапне перетворення випадкових латентних векторів у повноцінну багатотрекову партитуру з дотриманням гармонійної і ритмічної логіки. Кожен

етап системи – від конфігурації параметрів до генерації тактів, об’єднання треків і збереження у форматі MIDI – реалізовано окремим модулем, що дозволяє легко адаптувати та змінювати систему без порушення її загальної цілісності.

Реалізація була виконана мовою Python, що дозволило скористатися багатим набором спеціалізованих бібліотек, зокрема TensorFlow для машинного навчання, miditoolkit для обробки MIDI-даних, pypianoroll і matplotlib для візуалізації та аналізу згенерованих треків, а також numpy для обробки тензорів і чисельних масивів. Обрана архітектура, побудована на основі генеративно-змагальної моделі, виявилася придатною для побудови логічно пов’язаних у часі музичних структур. Особливістю системи є її здатність працювати як у режимі "з нуля", так і в умовах наявного умовного треку, що забезпечує гнучкість у підходах до генерації.

У центрі функціонування моделі перебуває генератор, який, отримуючи на вхід комбінацію шумових векторів, формує фрагменти композицій у форматі piano-roll. Система забезпечує повну послідовність: від ініціалізації параметрів, створення латентних векторів, проходження через генеративну мережу, формування нотного наповнення кожного треку, їх структурування, узгодження за довжиною і ритмікою, до остаточного формування повноцінної партитури. Завдяки цьому досягається цілісний та узгоджений музичний результат, який зберігається у вигляді стандартного MIDI-файлу і, за потреби, додатково виводиться як візуальне представлення або обчислюється за допомогою формальних метрик якості.

У ході роботи було впроваджено також спеціальний модуль для оцінювання якості згенерованих результатів. Він дозволяє аналізувати отримані композиції за формальними характеристиками: насиченість нотами, різноманітність висот, якість нот, відповідність ритмічним шаблонам, гармонійна узгодженість між треками. Це відкриває можливість не лише створювати музику, а й системно аналізувати її якість, що особливо важливо у науковому контексті. Програмна система показала високу стабільність, простоту запуску та гнучкість

налаштувань. Всі модулі можуть бути протестовані окремо, що дозволяє швидко виявляти помилки або змінювати конфігурацію генерації.

Висновок

У результаті виконання кваліфікаційної роботи бакалавра було створено метод створення музичних композицій на основі генеративного штучного інтелекту та було реалізовано відповідну інформаційну систему на основі генеративної змагальної мережі, призначену для створення фортепіанних партій у стилі заданого музичного датасету. Метою роботи було дослідження можливості застосування сучасних методів штучного інтелекту до задач автоматичної генерації музики, а також створення прототипу програмного забезпечення, здатного генерувати музичний матеріал з нуля або на основі заданих умов. Для досягнення цієї мети було проаналізовано наукові підходи до генерації послідовностей, обрано архітектуру моделі GAN та реалізовано відповідне середовище навчання і генерації за допомогою мови програмування Python та фреймворку TensorFlow.

Отримана система виконує генерацію фортепіанних партій у двох режимах – повністю з нуля (from scratch) або з урахуванням заданої партії, наприклад, барабанної (track-conditional). Результати експериментів показали, що в обох випадках модель здатна формувати музичні фрази зі структурою, гармонією та ритмікою, які мають ознаки цілісності та музичної логіки. Особливу увагу приділено дослідженню динаміки навчання: на різних етапах розвитку моделі спостерігалося поступове покращення якості згенерованих даних – від хаотичних наборів нот до композицій, які наближуються до результатів, очікуваних від людини.

Поставлене завдання повністю виконано: був розроблений та реалізований метод, було реалізовано програмну систему, проведено експериментальне дослідження, а також зроблено аналіз ефективності моделі на різних етапах тренування та за різних умов генерації. Результати роботи підтверджують актуальність і перспективність застосування генеративних моделей у музичній сфері. Розроблений програмний засіб може стати основою для подальших досліджень або бути частиною більшої системи автоматичної композиції.

Подальше вдосконалення системи можливе шляхом розширення функціональності – додавання підтримки інших інструментів, стилів чи жанрів, підключення гібридних архітектур із залученням рекурентних мереж або трансформерів, а також реалізації інтерактивного інтерфейсу для взаємодії користувача з генеративною моделлю. Крім того, можливо підвищити якість генерації за рахунок більших та більш різноманітних датасетів, а також використання технік донавчання на конкретних музичних стилях. Розроблена система демонструє реальний потенціал впровадження у галузях, пов’язаних із музичною творчістю, навчанням, генерацією фонового контенту та інтерактивними мультимедійними проєктами.

Перелік посилань

1. Ultralitics. GPT(Generative Pre-Trained Transformer). URL: https://www.ultralytics.com/glossary/gpt-generative-pre-trained-transformer.

2. IBM. What are diffusion models?. URL: https://www.ibm.com/think/topics/diffusion-models

3. Study.com. Music: Definition, Description, Characteristics URL: https://study.com/academy/lesson/what-is-music-definition-terminology-characteristics.html

4. The Ukrainians. Як музика впливає на ваш мозок та продуктивність. URL: https://theukrainians.org/yak-muzyka-vplyvaye/

5. Wikipedia. Музичний жанр. URL: https://uk.wikipedia.org/wiki/Музичний_жанр

6. Besfm. Основні поняття в музиці. URL: https://bestfm.fm/basic-concepts-in-music/

7. Енциклопедія Сучасної України. Мелодія. URL: https://esu.com.ua/article-66240

8. Study.com. What is harmony in music: definition theory. URL: https://study.com/academy/lesson/what-is-harmony-in-music-definition-theory.html

9. Britannica. Rhythm. URL: https://www.britannica.com/art/rhythm-music

10. eMastered blog. Що таке тембр. URL: https://emastered.com/uk/blog/what-is-timbre-in-music

11. MCSN. Who is composer of Music Works? URL: https://www.mcsnnigeria.org/composers/

12. Study.com. Theme & Variation in Music. URL: https://study.com/academy/lesson/theme-variation-in-music-definition-form-examples.html

13. Zero.machine.cx.ua. Що таке гармонізація мелодії? URL: https://zero.machine.cx.ua/ridnim/shho-take-garmonizaciya-melodii.html

14. Енциклопедія Сучасної України. Оркестровка, Оркестрування. URL: https://esu.com.ua/article-75764

15. ZS-Studio. Що таке аранжування і як це правильно зробити. URL: https://zs-studio.com.ua/shcho-take-aranzhuvannya-i-yak-ce-pravilno-zrobiti/

16. Wikipedia. Композиція(музика). URL: https://uk.wikipedia.org/wiki/Композиція_(музика)

17. Avid. What is a DAW? URL: https://www.avid.com/resource-center/what-is-a-daw

18. Native Instruments. What is a VST-plugin? URL: https://blog.native-instruments.com/vst-plugin/

19. Wikipedia. MIDI. URL: https://uk.wikipedia.org/wiki/MIDI

20. Revolta. Що таке плагіни VST і що вони роблять? URL: https://revolta.com.ua/didzhytal/shcho-take-plagini-vst-i-shcho-voni-roblyat.html

21. Colobridge. Що таке генеративний штучний інтелект і де він застосовується? URL: https://blog.colobridge.net/uk/2023/11/generative-artificial-intelligence-ua/

22. Top AI. 10 найкращих нейронних ШІ для створення музики в 2024. URL: https://top-ai.com.ua/produkty-ta-tehnologiyi/10-najkrashhyh-shi-dlya-stvorennya-muzyky-v-2024

23. OpenAI. MuseNet. URL: https://openai.com/index/musenet/

24. Magenta. Magenta. URL: https://magenta.tensorflow.org/

25. AIVA. Aiva. URL: https://www.aiva.ai/

26. eMastered blog. Що таке освоєння ШІ? URL: https://emastered.com/uk/blog/what-is-ai-mastering

27. Peak studio. Музичне виробництво. URL: https://www.peak-studios.de/uk/musikproduktion/

28. Civit, Miguel, et al. "A systematic review of artificial intelligence-based music generation: Scope, applications, and future trends." *Expert Systems with Applications* 209 (2022): 118190. doi: https://doi.org/10.1016/j.eswa.2022.118190

29. Roads, Curtis. "Research in music and artificial intelligence." *ACM Computing Surveys (CSUR)* 17.2 (1985): 163-190. doi: https://doi.org/10.1145/4468.4469

30. De Mantaras, Ramon Lopez, and Josep Lluis Arcos. "AI and music: From composition to expressive performance." *AI magazine* 23.3 (2002): 43-43.

31. Tensorflow. Generate music with an RNN. URL: https://www.tensorflow.org/tutorials/audio/music_generation

32. Magenta. Music Transformer. URL: https://magenta.tensorflow.org/music-transformer

33. Anas Al-Masri. How Does Backpropagation in a Neural Network Work? builtin. URL: https://builtin.com/machine-learning/backpropagation-neural-network#:~:text=Backpropagation%20is%20just%20a%20way,lower%20weights,%20and%20vice%20versa.

34. Liu, Xinqiao, Zhisheng Yang, and Jinyong Cheng. "Music recommendation algorithms based on knowledge graph and multi-task feature learning." *Scientific Reports* 14.1 (2024): 2055.

35. Namrata Dutt. Music genre classification using CNN. Medium. URL: https://medium.com/@namratadutt2/music-genre-classification-using-cnn-part-1-feature-extraction-b417547b8981.

36. GoIT Global. Що таке об’єктно-орієнтоване програмування: принципи, переваги та недоліки. URL: https://goit.global/ua/articles/shcho-take-ob-iektno-oriientovane-prohramuvannia-pryntsypy-perevahy-ta-nedoliky/

37. Techtarget. Domain-driven design (DDD). URL: https://www.techtarget.com/whatis/definition/domain-driven-design

38. SUNO. SUNO. URL: https://suno.com/create?wid=default

39. TikTok. TikTok. URL: https://www.tiktok.com/uk-UA/

40. Youtube. Youtube. URL: https://www.youtube.com/

41. GitHub. Magenta. URL: https://github.com/magenta/magenta

42. Soundraw. Soundraw. URL: https://soundraw.io/

43.Tensorflow. TensorFlow. URL: https://www.tensorflow.org/

44.PyPi. Pretty_midi. URL: https://pypi.org/project/pretty_midi/

45.NumPy. NumPy. URL: https://numpy.org/

46.MatPlotLib. MatPlotLib. URL: https://matplotlib.org/

ДОДАТКИ

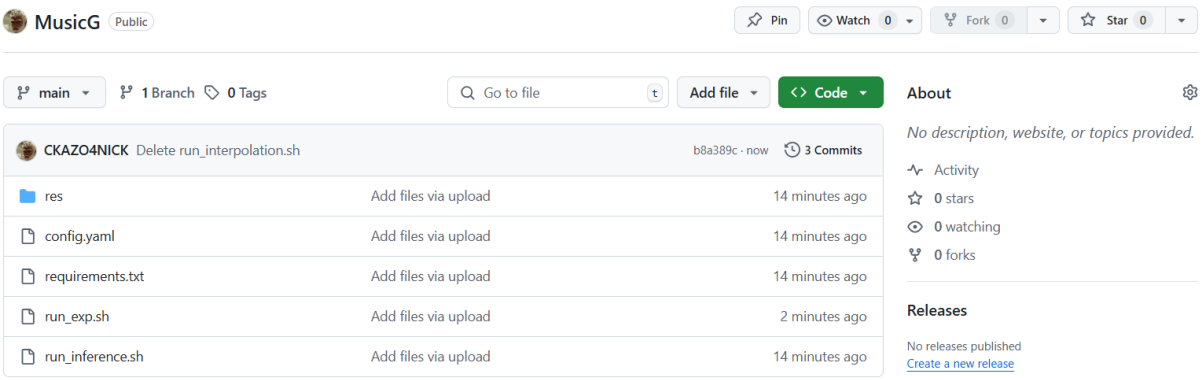
Додаток А

Програмний код

Посилання на репозиторій на GitHub:

https://github.com/CKAZO4NICK/MusicG.git

Вигляд репозиторію:



Опис вмісту репозиторію:

– config.yaml – файл налаштування конфігурації басу;

– requirements.txt – інформація про необхідні залежності для коректної роботи програми;

– run_interference.sh – запуск методу без донавчання;

– run_exp.sh – запуск методу з донавчанням;

– res – папка, у якій знаходяться результати, тобто згенерована музика.

Додаток Б

Стаття

УДК 004.89

ПРИЙМА А.В.
Хмельницький національний університет
ORCID ID: 0009-0001-7272-007X
e-mail: andriipriyma@gmail.com

ДОСЛІДЖЕННЯ МЕТОДУ ГЕНЕРАЦІЇ БАГАТОТРЕКОВИХ СИМВОЛІЧНИХ КОМПОЗИЦІЙ ЗА ДОПОМОГОЮ ГЕНЕРАТИВНОГО ШТУЧНОГО ІНТЕЛЕКТУ

В роботі наведено результат дослідження методу генерації багатотрекових символічних композицій за допомогою генеративного штучного інтелекту. Досліджено метод генерації композицій та проведено експерименти щодо його ефективності та якості моделей при різних умовах генерації.

Ключові слова: штучний інтелект, музика, музична композиція, партитура, piano-roll, модель.

PRYIMA Andrii V.
Khmelnitskyi National University

RESEARCH OF A METHOD FOR GENERATION OF MULTITRACK SYMBOLIC COMPOSITIONS USING GENERATIVE ARTIFICIAL INTELLIGENCE

This study explores a method for symbolic music generation using artificial intelligence techniques. Unlike traditional rule-based or statistical approaches, the proposed method employs deep generative models to automatically create multi-track, polyphonic musical compositions. The focus is placed on capturing the temporal structure of music and ensuring harmonic and rhythmic coherence across multiple instrument tracks. A model architecture is introduced that generates music in the form of piano-roll representations, treating musical bars as fundamental units. The system supports both

autonomous generation and human-AI collaboration, where the model can accompany a user-provided melody with additional tracks. A specially prepared dataset of symbolic music was used for training, and a series of objective metrics were developed to evaluate the quality of the generated output. Results from both quantitative analysis and a subjective user study demonstrate that the method is capable of producing musically coherent and structurally consistent compositions. This research contributes to the development of AI-assisted tools for creative music production and expands the possibilities of automated composition in the symbolic domain.

Keywords: artificial intelligence, music, musical composition, score, piano-roll, model.

Постановка проблеми

У сучасному світі штучний інтелект (ШІ) дедалі активніше проникає у сферу творчості, зокрема у створення музики. Генерація музичних композицій за допомогою ШІ відкриває нові можливості для автоматизації творчих процесів, підтримки музикантів та створення інноваційного музичного контенту. Проте, попри значний прогрес у галузі генеративних моделей, завдання створення якісної, багатотрекової та поліфонічної символічної музики залишається складним і недостатньо вирішеним. Основна проблема полягає в тому, що музика є не лише послідовністю звуків у часі, але й структурованим набором взаємопов'язаних елементів – гармонії, ритму, мелодії та інструментального балансу. Більшість моделей демонструють успіх у генерації простих мелодій або одноголосних треків, однак не враховують повною мірою складну взаємодію між різними інструментальними партіями. Це призводить до створення фрагментованих або гармонічно несумісних результатів.

Актуальність теми зумовлена потребою в розвитку інтелектуальних систем, здатних не лише імітувати, але й творчо доповнювати процес музичної композиції. Дослідження в цій сфері сприяють розширенню можливостей для музичної індустрії, освіти, інтерактивних медіа та персоналізованого аудіоконтенту. У контексті

активного розвитку генеративних нейромереж тема є перспективною як у науковому, так і в практичному аспектах.

Метою є аналіз, дослідження та покращення ефективності методу генерації багатотрекових символічних композицій за допомогою генеративного штучного інтелекту.

Аналіз останніх джерел

У роботі [1] розглядається тема автоматизованої генерації музики за допомогою штучного інтелекту як одна з найбільш привабливих та значущих у сфері ШІ, музики та мультимедіа. Автори підкреслюють, що сучасне створення музики за участю ШІ охоплює широкий спектр задач, зокрема генерацію мелодій, написання пісень, створення акомпанементу, аранжування, виконання, генерацію тембру, синтез звуку та навіть синтез співу. У роботі пропонується систематизований підхід до теми: спочатку подається вступ до основ музичної теорії та технік глибокого навчання, потім розглядаються основні складові системи ШІ-композиції (створення партитур, виконання та звукогенерація), а також розширені теми, як-от моделювання музичної структури, стилю, емоцій та тембру. Завершується робота аналізом викликів і можливих напрямів розвитку у галузі. Основна ідея – надати широкому колу дослідників та практиків уявлення про поточний стан і майбутнє штучного інтелекту в музиці.

Робота [2] починається з констатації того, що хоча межі штучного інтелекту ще не повністю визначені, комп'ютери вже здатні виконувати завдання, які колись вважались притаманними виключно людям-музикантам. Після короткого історичного вступу автор наголошує на важливості застосування ШІ у чотирьох основних напрямках музичних досліджень: композиція, виконання, музикознавство та цифрова обробка звуку. Далі подається огляд сучасних досягнень у цих галузях, що демонструють практичне використання ШІ. Завершальна частина статті розмірковує над тим, як планування та навчання в системах штучного інтелекту можуть сприяти накопиченню знань і покращенню поведінки інтелектуальних музичних систем, тобто розширенню їхніх можливостей у створенні та інтерпретації музики.

У роботі [3] розглядаються три основні типи комп'ютерних

музичних систем, які базуються на ШІ: системи для композиції, імпровізації та виконання. Після короткого опису прикладів кожного типу основна увага приділяється темі музичної виразності – однієї з найскладніших задач комп'ютерного музикування. Автори зазначають, що головним викликом у моделюванні виразності є відтворення «дотику виконавця», тобто його інтерпретаційного знання. Попередні методи, які спиралися на формальні правила, виявилися недостатніми. Натомість пропонується підхід, що наближається до людського навчання – наслідування шляхом спостереження, з використанням прикладів реальних виконавців. Описано систему SAXEX, яка здатна створювати якісні, емоційно забарвлені джазові соло, базуючись на записах людських виконавців, використовуючи методику Case-Based Reasoning.

Метою роботи є: покращення якості та різноманітності генерованих музичних композицій шляхом застосування методів генеративного штучного інтелекту.

Виклад основного матеріалу

Розроблений метод використовує поетапну генерацію музичних композицій для створення однієї цілої композиції у вигляді послідовності тактів. Схему методу наведено на рисунку 1.

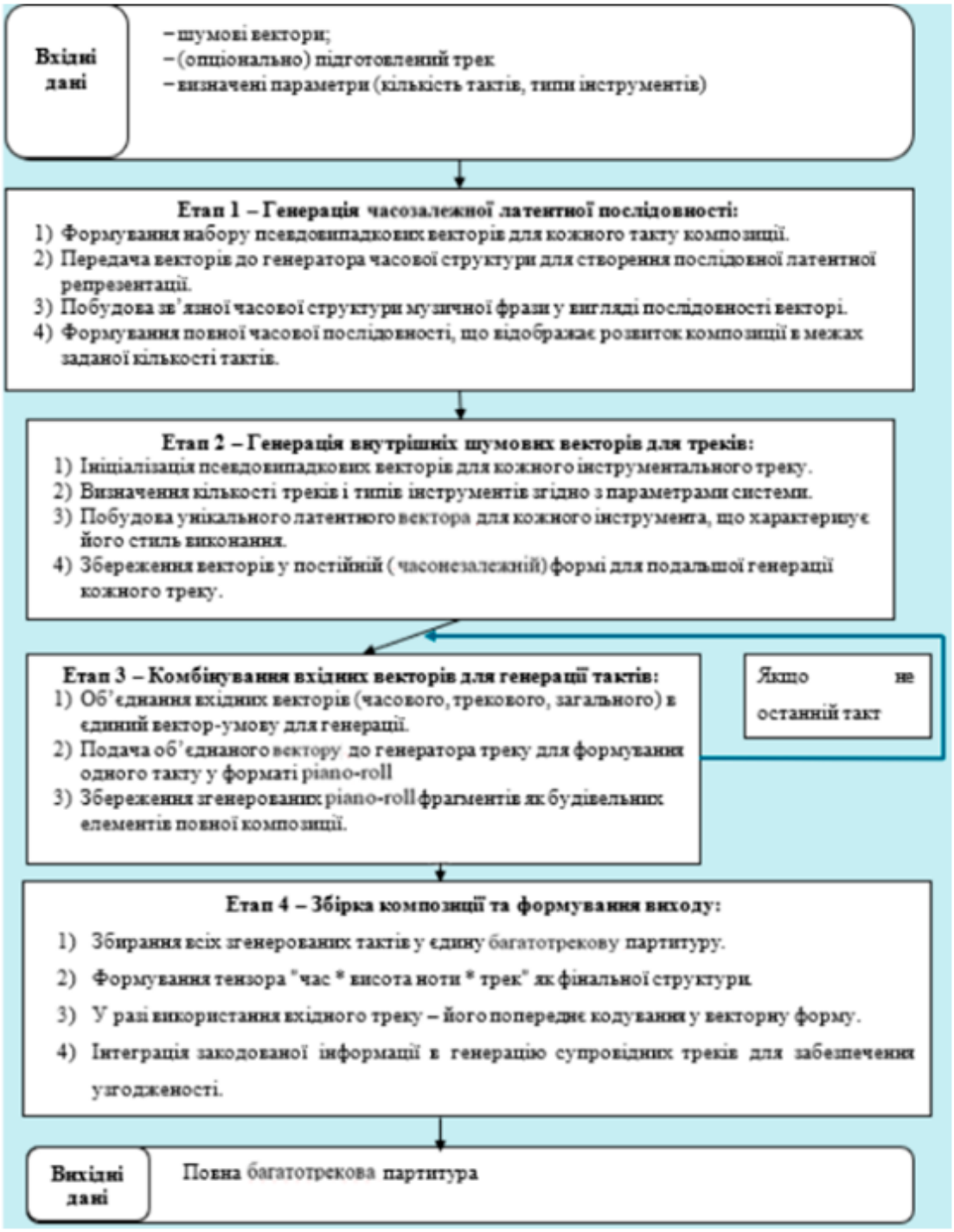


Рис. 1. Схема методу генерації багатотрекових символічних композицій

Увесь процес організовано у вигляді чотирьох логічно взаємопов'язаних етапів, кожен з яких виконує свою специфічну роль у створенні фінального музичного фрагмента. На вході система отримує шумові вектори, які виступають джерелом

генерації акомпанементу. Ще одним обов'язковим елементом вхідних даних є параметри, що визначають кількість тактів у композиції та перелік інструментів, які будуть задіяні у процесі генерації.

Перший етап – це генерація узгодженої латентної послідовності. Його суть полягає у формуванні набору векторів, які представляють собою часову структуру музичної фрази. Кожен такт композиції отримує відповідний вектор, що описує його положення у музичному часі, відображає ритмічну та структурну організацію. Ці вектори передаються до генератора часової структури, який формує повну послідовність – вона слугує "скелетом" композиції, її загальною формою. Результатом цього етапу є серія узгоджених векторів, які відповідають кількості заданих тактів і відображають розвиток музичної ідеї з часом.

Другий етап пов'язаний із генерацією внутрішніх шумових векторів для кожного інструментального треку. Система ініціалізує окремі вектори для кожного треку, беручи до уваги кількість інструментів і їхні типи, які були задані у параметрах. Для кожного інструмента створюється індивідуальний набір латентних шумових векторів, які мають передати характер гри цього інструмента, стилістичні й ритмічні особливості. Особливістю цього етапу є те, що усі вектори зберігаються у стабільній, узгодженій формі, яка буде використана на наступних етапах генерації.

На третьому етапі здійснюється комбінування різних типів векторів для створення повноцінних музичних тактів. Для цього об'єднуються часові вектори (отримані на першому етапі), вектори, пов'язані з окремими треками (створені на другому етапі), а також загальні шумові вектори у єдину умову генерації. Цей об'єднаний вектор подається до генератора відповідного інструментального треку для кожного окремого такту. Генератор формує фрагмент у форматі piano-roll, який є поширеним представленням символічної музики, де кожна нота відображається у часі та по висоті. Таким чином, утворюються окремі музичні фрагменти, які поєднують інформацію про структуру, стиль, інструмент та момент виконання. Цей етап повторюється для кожного наступного такту, поки не буде охоплено всю задану довжину композиції.

Четвертий і фінальний етап присвячений збиранню окремих тактів

у завершену багатотрекову партитуру. Усі згенеровані фрагменти інтегруються у єдину музичну структуру, де враховується розташування нот у часі, розподіл між треками, а також їхня послідовність. У випадку, якщо на вхід була подана умова у вигляді попередньо підготовленого треку, він кодується у векторну форму і також береться до уваги під час формування фінального результату. Одним з важливих аспектів цього етапу є інтеграція інформації про часову структуру, яка була сформована на початку, для забезпечення логічної та музичної узгодженості між усіма треками.

У результаті роботи всього методу на виході формується повна багатотрекова партитура, яка може містити кілька інструментальних партій, розподілених у часі, та бути використаною для подальшого відтворення, аналізу або музичної обробки. Уся система побудована з урахуванням музичної структури, ритму, гармонії та варіативності, що забезпечує гнучкість і здатність до творчої генерації музики з урахуванням заданих параметрів і умов.

Також, у методі є свої групи функцій, які зображені на рисунку 2.

Рис. 2. Функціональна система методу

На цій схемі представлено логічно структурований процес генерації музичної композиції із застосуванням штучного інтелекту. Весь

процес поділено на чотири послідовні групи функцій, які описують повний цикл – від ініціалізації генерації до виведення фінального результату у форматі, придатному для прослуховування або збереження.

Перший блок – **ініціалізація параметрів генерації** – є підготовчим етапом, у якому система отримує початкові умови для формування майбутньої композиції. Тут визначається кількість тактів, що визначає тривалість музичного фрагменту. Потім обирається кількість інструментальних треків – це дозволяє системі знати, скільки різних інструментів братиме участь у композиції. Далі здійснюється вибір типів інструментів: це може бути набір із таких типових партій, як бас, ударні, гітара, фортепіано тощо. Нарешті, ініціалізуються випадкові шумові вектори – набір числових параметрів, які вводять стохастичність у процес генерації і забезпечують різноманітність музичних результатів.

Другий блок – **генерація структури композиції** – відповідає за створення абстрактного «каркасу» майбутньої музики. На цьому етапі формується часова послідовність латентних векторів, які визначають порядок розвитку композиції, зокрема ритмічну структуру і фразування. Паралельно генеруються незмінні шумові вектори, які залишаються сталими протягом усього процесу генерації і забезпечують загальну стилістичну цілісність композиції. Потім усі вхідні вектори (структурні, трекові, загальні) комбінуються в єдину структуру, яка буде використана як умова для побудови конкретних музичних елементів.

У третьому блоці – **побудова музичних тактів** – система переходить до генерації конкретного вмісту композиції. Зкомбіновані вектори подаються до генератора тактів, який будує віртуальне представлення нот у вигляді piano-roll – матриці, де зафіксовано моменти початку, тривалість і висоту нот для кожного інструментального треку. Далі всі згенеровані такти з'єднуються у межах кожного треку, утворюючи безперервну музичну лінію. Результатом цього етапу є формування повної багатотрекової партитури – повноцінної музичної структури, готової до збереження чи прослуховування.

Четвертий блок – **збереження та виведення результату** – виконує фінальні операції, що перетворюють згенеровану структуру у форму,

зручну для користувача. Спершу всі треки об'єднуються в єдиний фінальний результат. Потім відбувається конвертація у формат MIDI– універсальний цифровий формат для збереження символічної музики, сумісний із багатьма музичними редакторами та відтворювачами. Сформований файл можна зберегти на фізичному носії, або вивести безпосередньо для подальшого використання, наприклад, у музиці, навчанні або навіть у професійній творчості.

Експериментальна частина

Щоб дослідити ефективність методу генерації багатотрекових символічних композицій, було проведено ряд експериментів. Перший експеримент присвячений перевірці чіткості вихідної музики у вигляді piano-roll на різних етапах навчання. Метою експериментального дослідження було проаналізувати динаміку навчання генеративної моделі шляхом фіксації її результатів на різних етапах тренування. Для цього було обрано п’ять контрольних точок: 0, 700, 2500, 6000 та 7900 ітерацій. На кожному з цих етапів модель зупинялась, і на основі поточного стану ваг здійснювала повну генерацію нового музичного фрагмента з нуля. Згенеровані результати зберігались у форматі piano-roll – візуальному представленні музики, де вісь X відповідає часовій шкалі, а вісь Y – висоті нот.

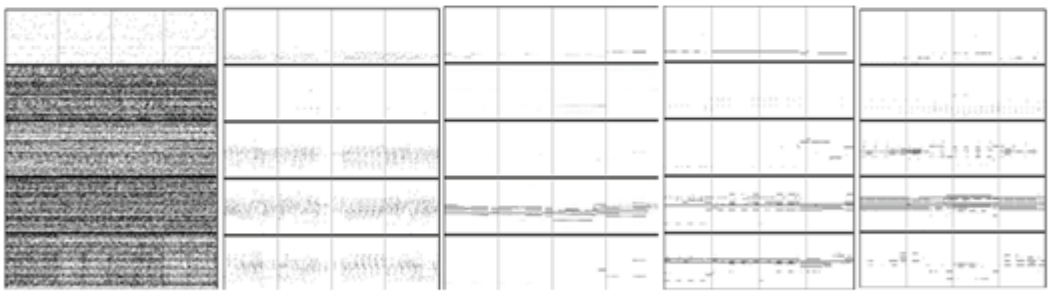


Рис. 3. Зафіксовані дані на першому етапі при а)0, б)700, в)2500 кроків, г)6000 та д)7900 кроків

А)Перший етап (0 кроків): результат генерації мав випадковий і хаотичний характер: відсутні ритмічні та гармонійні зв’язки, що є очікуваним наслідком необроблених випадкових параметрів.

Б)Другий етап (700 кроків): композиція залишається малоструктурованою, але спостерігається формування первинних ознак впорядкованості

В)Третій етап (2500 кроків): модель демонструє значний прогрес. Незважаючи на це, у результаті все ще присутні окремі недоліки.

Г)Четвертий етап (6000 кроків): генерація суттєво наближається до цілісної музичної композиції

Д)П'ятий етап (7900 кроків): найвищий рівень якості генерації.

Отже, на початку експерименту модель не може сформувати повноцінний структурований трек, але чим більше ітерацій впроваджено, тим краще стає трек.

У межах дослідження було проведено порівняльний експеримент з метою оцінити якість генерації музики різними архітектурними варіантами моделей. Основним завданням цього етапу було визначення здатності моделей відтворювати ключові характеристики музичної логіки: структурованість, ритмічну цілісність, гармонійну узгодженість та відповідність окремих треків загальній композиційній структурі. Для оцінки результатів були застосовані як внутрішньотрекові (intra-track), так і міжтрекові (inter-track) метрики.

Таблиця 1

		empty bars (**EB**; %)					used pitch classes (**UPC**)				qualified notes (**QN**; %)				**DP** (%)
		B	**D**	**G**	**P**	**S**	**B**	**G**	**P**	**S**	**B**	**G**	**P**	**S**	**D**
training data		8.06	8.06	19.4	24.8	10.1	1.71	3.08	3.28	3.38	90.0	81.9	88.4	89.6	88.6
from scratch	jamming	6.59	2.33	18.3	22.6	6.10	1.53	3.69	4.13	4.09	71.5	56.6	62.2	63.1	93.2
	composer	0.01	28.9	1.34	0.02	0.01	2.51	4.20	4.89	5.19	49.5	47.4	49.9	52.5	75.3
	hybrid	2.14	29.7	11.7	17.8	6.04	2.35	4.76	5.45	5.24	44.6	43.2	45.5	52.0	71.3
	ablated	92.4	100	12.5	0.68	0.00	1.00	2.88	2.32	4.72	0.00	22.8	31.1	26.2	0.0
track-conditional	jamming	4.60	3.47	13.3	—	3.44	2.05	3.79	—	4.23	73.9	58.8	—	62.3	91.6
	composer	0.65	20.7	1.97	—	1.49	2.51	4.57	—	5.10	53.5	48.4	—	59.0	84.5
	hybrid	2.09	4.53	10.3	—	4.05	2.86	4.43	—	4.32	43.3	55.6	—	67.1	71.8

Інтра-трекова оцінка охоплювала такі показники: частка порожніх тактів (Empty Bars), кількість використовуваних класів висот нот (Used Pitch Classes), частка кваліфікованих нот (Qualified Notes), що свідчать про фрагментованість звукових подій, та відповідність ритмічним шаблонам ударних (Drum Patterns). Для оцінки гармонійної взаємодії між треками

застосовувався показник Tonal Distance, який відображає ступінь гармонійної узгодженості між окремими інструментальними партіями.

У Таблиці 1 наведено результати для двох основних режимів генерації: повністю автономної генерації з нуля (from scratch) та умовної генерації на основі заздалегідь визначеного треку (track-conditional). Для кожного з цих режимів розглядалися три архітектури, а саме Jamming (кожен трек створюється незалежним генератором), Composer (один загальний генератор відповідає за формування всіх треків одночасно) і Hybrid (поєднує підходи, використовуючи як спільні, так і приватні латентні простори). Також було протестовано обмежену версію моделі composer без нормалізації пакетів (аблативна модель), яка виступає як умовна нижня межа якості генерації.

Результати показали, що jamming-модель генерує активні треки з невеликою кількістю порожніх тактів, що свідчить про ефективне ритмічне наповнення. Однак у composer- та hybrid-моделях частка порожніх тактів, особливо для ударних та фортепіано, є суттєво вищою (20–30%), що може вказувати на менш стійку ритмічну структуру. Ablated модель майже повністю втрачає здатність до генерації, результати є переважно порожні.

Аналіз Used Pitch Classes показав, що jamming-модель відтворює мінімальну кількість висотних класів для басу, що узгоджується з його функцією як монофонічного інструмента. Інші треки демонструють ширший діапазон, але в composer- та hybrid-моделях він вищий, ніж у навчальних даних, що може свідчити про перенасичення висотними класами та менш контрольовану гармонійну логіку.

Показник Qualified Notes виявився чутливим до структури вихідних тензорів: значення у навчальному наборі перевищували 80–90%, тоді як згенеровані дані мали значно нижчі результати, особливо в composer- та hybrid-моделях. Це вказує на наявність коротких, нефункціональних нот, що можуть виникати внаслідок особливостей бінаризації.

Для метрики Drum Patterns jamming-модель показала високу відповідність (понад 90%) типовим ритмічним шаблонам, тоді як composer- та hybrid-виконання дещо поступались. У ablated-моделі ця метрика

дорівнювала нулю, що підтверджує відсутність осмисленої генерації ударної партії.

У режимі умовної генерації (track-conditional) результати в цілому покращуються. Зокрема, jamming-модель демонструє високу ритмічну активність і зменшену кількість порожніх тактів. Hybrid- і composer-архітектури зберігають середні показники, при цьому hybrid досягає порівнянного або навіть кращого рівня якості порівняно з composer, зберігаючи більшу гнучкість.

Оцінка міжтрекової гармонії за допомогою показника Tonal Distance показала, що composer- і hybrid-моделі мають найнижчі значення, що свідчить про високу узгодженість між партіями. Jamming-модель, яка не моделює міжтрекову взаємодію, демонструє значення, ближчі до навчальних, але й менш гармонійно цілісні результати.

Загалом, результати експерименту свідчать про те, що hybrid-модель демонструє найбільш збалансовану продуктивність – вона поєднує гнучкість архітектури з досить високими показниками як на внутрішньотрековому, так і міжтрековому рівнях. Composer-модель краще справляється з гармонійною координацією треків, однак втрачає якість у ритмічному наповненні. Jamming-модель, у свою чергу, ефективна у генерації окремих партій, але менш придатна до формування узгоджених ансамблевих структур. Ablated-модель виступає як приклад неефективного архітектурного рішення, що підтверджує важливість повноцінного дизайну моделі та застосування нормалізації у процесі навчання.

Висновки

У дослідженні було реалізовано метод генерації багатотрекової символічної музики на основі штучного інтелекту. Запропонована архітектура забезпечує формування цілісних музичних фраз із збереженням ритміки, гармонії та міжтрекової взаємодії. Аналіз проміжних результатів показав поступове зростання якості генерації: від хаотичних звуків до впорядкованих структур, що свідчить про ефективність навчання. Порівняння трьох архітектур виявило переваги hybrid-моделі як найбільш збалансованої за всіма метриками. Composer-модель забезпечує кращу

гармонійну узгодженість, а jamming – вищу якість окремих треків, проте з гіршою інтеграцією між ними. Спрощена ablated-модель підтвердила критичну важливість повноцінної архітектури.

Отримані результати демонструють перспективність використання генеративних моделей для створення музики та закладають основу для подальшого розвитку у напрямку покращення виразності та музичної форми.

Література

1. Tan, Xu, and Xiaobing Li. "A tutorial on AI music composition." *Proceedings of the 29th ACM international conference on multimedia*. 2021. doi; https://doi.org/10.1145/3474085.3478875

2. Roads, Curtis. "Research in music and artificial intelligence." *ACM Computing Surveys (CSUR)* 17.2 (1985): 163-190. doi: https://doi.org/10.1145/4468.4469

3. De Mantaras, Ramon Lopez, and Josep Lluis Arcos. "AI and music: From composition to expressive performance." *AI magazine* 23.3 (2002): 43-43.

4. Zulić, Harun. "How AI can change/improve/influence music composition, performance and education: three case studies." *INSAM Journal of Contemporary Music, Art and Technology* 2 (2019): 100-114.

5. Hong, Joo-Wha, et al. "Human, I wrote a song for you: An experiment testing the influence of machines' attributes on the AI-composed music evaluation." *Computers in Human Behavior* 131 (2022): 107239.

6. Dash, Adyasha, and Kathleen Agres. "Ai-based affective music generation systems: A review of methods and challenges." *ACM Computing Surveys* 56.11 (2024): 1-34.

7. Kaliakatsos-Papakostas, Maximos, Andreas Floros, and Michael N. Vrahatis. "Artificial intelligence methods for music generation: a review and future perspectives." *Nature-inspired computation and swarm intelligence* (2020): 217-245.

8. Hassink, Nina. *AI Music vs. Human Music: Exploring Artistic and Economic Value Judgments in Music*. MS thesis. 2024.

9. Louie, R., Coenen, A., Huang, C. Z., Terry, M., & Cai, C. J. (2020,

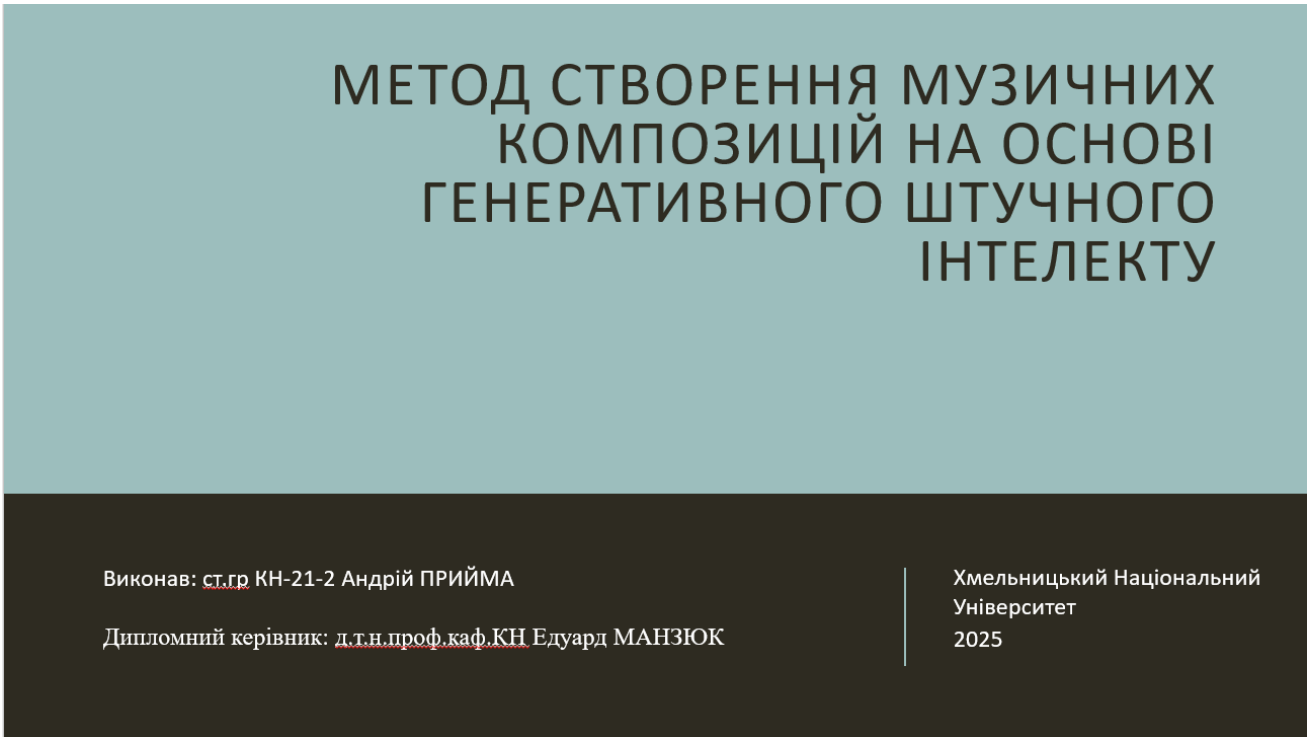
April). Novice-AI music co-creation via AI-steering tools for deep generative models. In *Proceedings of the 2020 CHI conference on human factors in computing systems* (pp. 1-13).

Надійшла / Paper received : *заповнюється редакцією*

Надрукована/Printed : *заповнюється редакцією*

Додаток В

Презентаційний матеріал



2

АКТУАЛЬНІЙСТЬ

Як ШІ з'явилось в житті людини, воно з'явилось у більшості сфер, включно у музичній сфері. Не кожна людина може для своєї певної мети можуть найняти професійного музиканта. ШІ дозволяє зробити музику, що буде звучати на фоні в грі чи дасть натхнення музикантам написати нову пісню.

3

МЕТА ТА ЗАДАЧІ

Мета кваліфікаційної роботи бакалавра – покращення якості та різноманітності генерованих музичних композицій шляхом застосування методів генеративного штучного інтелекту.

Задачі кваліфікаційної роботи бакалавра:

-проаналізувати сучасний стан створення музичних композицій за допомогою генеративного штучного інтелекту;

-дослідити існуючі методи створення музичних композицій за допомогою генеративного штучного інтелекту;

-розробити метод автоматизованого створення музичних композицій на основі технологій генеративного штучного інтелекту;

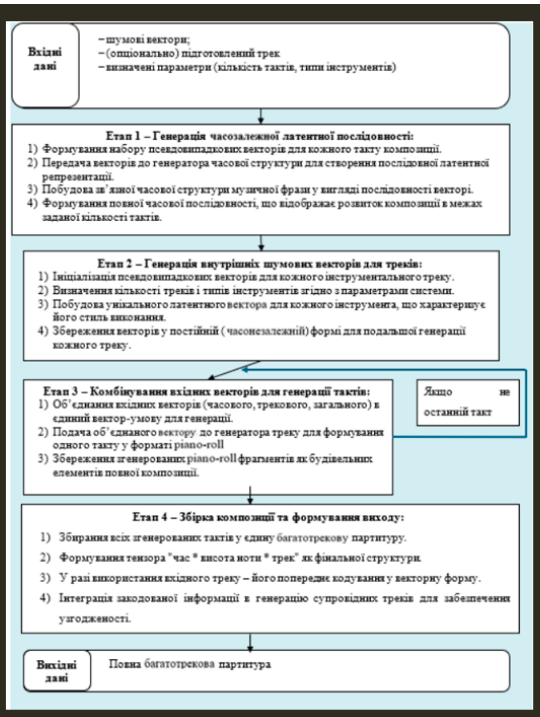
-розробити відповідну інформаційну систему, що використовує метод автоматизованого створення музичних композицій на основі технологій генеративного штучного інтелекту;

-провести тестування створеної інформаційної системи;

-дослідити ефективність інформаційної системи.

СХЕМА МЕТОДУ ГЕНЕРАЦІЇ МУЗИЧНИХ КОМПОЗИЦІЙ ЗА ДОПОМОГОЮ ГЕНЕРАТИВНОГО ШТУЧНОГО ІНТЕЛЕКТУ

4



5

ФУНКЦІОНАЛЬНА СИСТЕМА МЕТОДУ

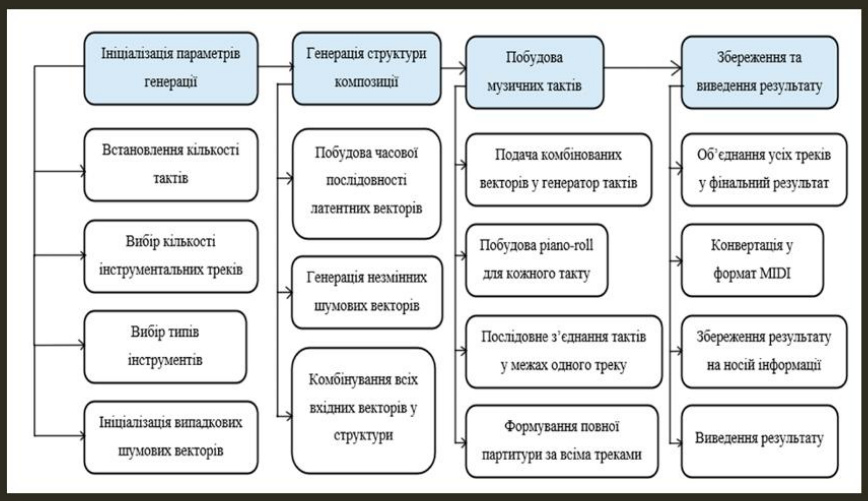


СХЕМА ВЗАЄМОДІЇ КОМПОНЕНТІВ

6

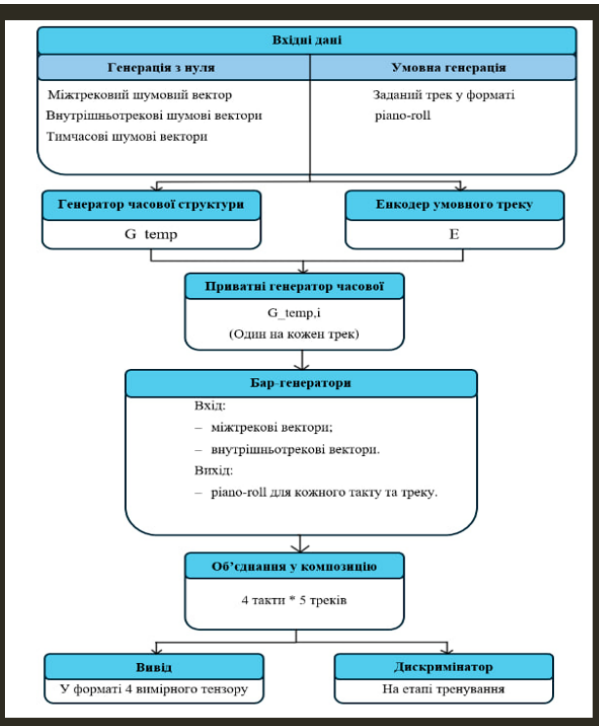
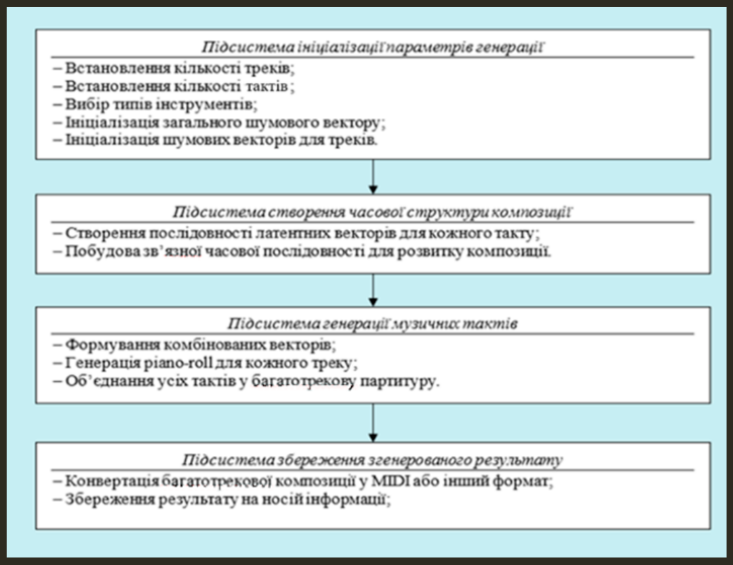


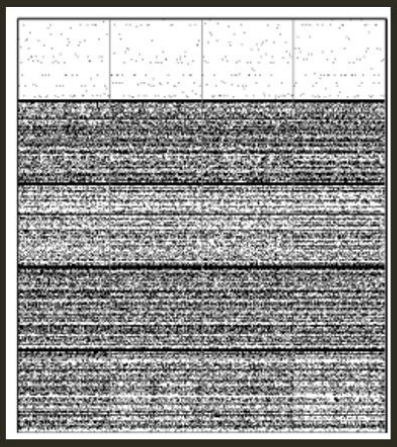
СХЕМА ІНФОРМАЦІЙНОЇ СИСТЕМИ СТВОРЕННЯ МУЗИКАЛЬНИХ КОМПОЗИЦІЙ

7

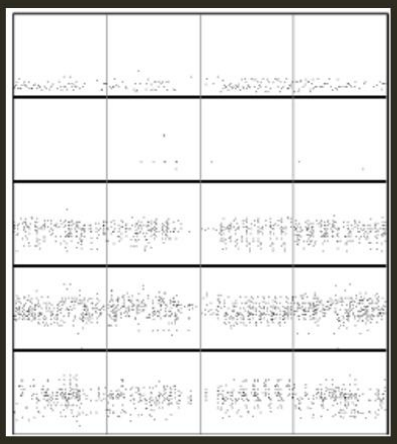


8

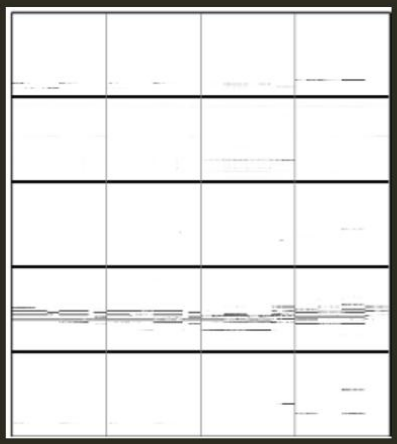
ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ГЕНЕРАЦІЇ МОДЕЛІ



КРОК 0



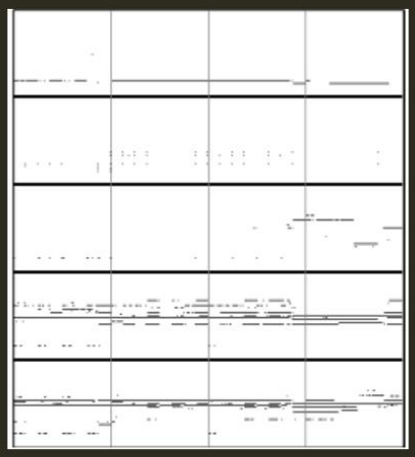
КРОК 700



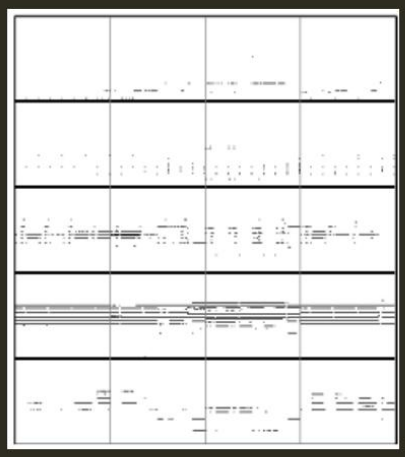
КРОК 2500

9

ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ГЕНЕРАЦІЇ МОДЕЛІ



КРОК 6000



КРОК 7900

10

ДОСЛІДЖЕННЯ ЯКОСТІ МОДЕЛЕЙ

		Empty bars(EB;%)					Used pitch classes (UPC)				Qualified notes (QN; %)				DP (%)
		B	D	G	P	S	B	G	P	S	B	G	P	S	D
Training data		8.06	8.06	19.4	24.8	10.1	1.71	3.08	3.28	3.38	90.0	81.9	88.4	89.6	88.6
From scratch	jamming	6.59	2.33	18.3	22.6	6.10	1.53	3.69	4.13	4.09	71.5	56.6	62.2	63.1	93.2
	composer	0.01	28.9	1.34	0.02	0.01	2.51	4.20	4.89	5.19	49.5	47.4	49.9	52.5	75.3
	hybrid	2.14	29.7	11.7	17.8	6.04	2.35	4.76	5.45	5.24	44.6	43.2	45.5	52.0	71.3
	ablated	92.4	100	12.5	0.68	0.00	1.00	2.88	2.32	4.72	0.00	22.8	31.1	26.2	0.0
Track-conditional	jamming	4.60	3.47	13.3	-	3.44	2.05	3.79	-	4.23	73.9	58.8	-	62.3	91.6
	composer	0.65	20.7	1.97	-	1.49	2.51	4.57	-	5.10	53.5	48.4	-	59.0	84.5
	hybrid	2.09	4.53	10.3	-	4.05	2.86	4.43	-	4.32	43.3	55.6	-	67.1	71.8

11

ВИСНОВКИ

У результаті було створено метод створення музичних композицій на основі генеративного штучного інтелекту, розроблено відповідну інформаційну систему, проведено дослідження ефективності моделі.

12

ДЯКУЮ ЗА УВАГУ

Anti-Plagiarism (UA) v-15.281 Educational

The maximum coincidence with one document 2.0%

Dictionaries check: en_US, ru_RU, ua_UA. **Errors in the documents: 9%**

	Document		Sum coincidence on the DB	
ID: 244069 Title: КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА на тему Метод створення музичних композицій на основі технологій генеративного штучного інтелекту Added in a DB: 2025-06-06 Authors: Андрій ПРИЙМА Heads: Едуард МАНЗЮК Consultants: Opponents:	Symbols	Lexemes	Symbols	Lexemes
	77925	1168	3554 (5%)	54 (5%)

Plagiarism sources

ID	Description	Plagiarism presence in the document	
		Symbols	Lexemes

Протокол аналізу звіту подібності науковим керівником

Заявляю, що я ознайомився (-лась) з Повним звітом подібності, який був згенерований Системою виявлення і запобігання плагіату щодо роботи:

Автор: Андрій ПРИЙМА

Співавтор:

Назва: КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА на тему Метод створення музичних композицій на основі технологій генеративного штучного інтелекту

Науковий керівник: Едуард МАНЗЮК, д.т.н., доцент

Підрозділ: Кафедра комп'ютерних наук

Коефіцієнт подібності 1:6.5%

Коефіцієнт подібності 2:3.3%

Мікропробіли: 0

Заміна букв: 0

Інтервали: 0

Білі знаки: 16

Дата створення звіту: 2025-06-06 22:17:17.0

Після аналізу Звіту подібності констатую наступне:

☑ **Запозичення, виявлені в роботі є законними і не є плагіатом. Рівень подібності не перевищує допустимої межі. Таким чином робота незалежна і приймається.**

☐ **Запозичення не є плагіатом, але перевищено граничне значення рівня подібностей. Таким чином робота повертається на доопрацювання.**

☐ **Виявлено запозичення і плагіат або навмисні текстові спотворення (маніпуляції), як передбачувані спроби укриття плагіату, які роблять роботу невідповідною вимогам законодавства (Ст. 32. ЗУ Про вищу освіту, пункт 3.1, Ст. 42. ЗУ Про освіту) та вимог НАЗЯВО (Критерій 5), а також кодексу етики і процедурам. Таким чином робота не приймається.**

Обгрунтування:

Дата 06.06.25 експерт Первський С.С.

РІШЕННЯ ЕКСПЕРТНОЇ КОМІСІЇ КАФЕДРИ КОМП'ЮТЕРНИХ НАУК

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Назва кваліфікаційної роботи *Метод створення музичних композицій на основі технологій генеративного штучного інтелекту*
Автор *студент групи КН-21-2 Андрій ПРИЙМА*
Освітня програма *Комп'ютерні науки*
Рівень вищої освіти *перший (бакалаврський)*
Спеціальність *122 – Комп'ютерні науки*
Науковий керівник: *д.т.н., проф. каф. комп'ютерних наук Едуард МАНЗЮК*

На основі аналізу кваліфікаційної роботи на дотримання вимог академічної доброчесності (у т.ч. відсутності ознак академічного плагіату) з урахуванням результатів перевірки роботи спеціалізованим програмними засобами комісія зробила такий висновок:

№	Висновок	Позначка про відповідність
1	Ознаки академічного плагіату	
1.1	Запозичення, виявлені в роботі, є законними і не є академічним плагіатом (далі – зазначаються підстави віднесення запозичень до правомірних, якщо потрібно). Робота приймається до захисту.	***відповідає***
1.2	Виявлені запозичення не є академічним плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи (далі – зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). Робота приймається до захисту, але має бути відкоригована.	
1.3	Виявлені запозичення не є академічним плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. Робота може бути допущена до захисту після того як буде відкоригована та доопрацьована і успішно пройде повторну перевірку на академічний плагіат.	
1.4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття текстовних запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
2	Інші види порушень академічної доброчесності	***відсутні***

Підтвердження:

Запозичення, виявлені в роботі Андрія ПРИЙМИ, не є плагіатом, оскільки: запозичення розміщені в розділі огляду існуючих підходів, не описують безпосередньо авторську роботу і не стосуються її результатів; усі запозичення фрагментарні; до запозичень входять фрагменти, які не мають авторства і містять поширені конструкції та загальновідомі терміни, скорочення. Рівень подібності не перевищує допустимої межі. Таким чином, робота є законною та приймається до захисту.

Обсяг запозичень, визначений системами виявлення збігів/ідентичності/схожості:
- за системою Anti-Plagiarism: 2%;
- за системою StrikePlagiarism КП1: 6,5%, КП2: 3,3%.

06.06.2025

Завідувач кафедри *Олександр БАРМАК*

Гарант освітньої програми *Олександр МАЗУРЕЦЬ*

Керівник кваліфікаційної роботи *Едуард МАНЗЮК*

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
МОН УКРАЇНИ

Кафедра комп'ютерних наук

РЕЦЕНЗІЯ

на кваліфікаційну роботу бакалавра

студента *гр. КН-21-2 Прийма Андрій Віталійович*
за темою: *Метод створення музичних композицій на основі генеративного штучного інтелекту*

1. Актуальність обраної теми
З урахуванням того, що штучний інтелект впроваджується все частіше та більше в життя людини, тема роботи є актуальною.

2. Повнота розкриття мети та завдань роботи
Мета роботи чітко визначена та встановлена. Розроблено метод для створення музичних композицій. Поставлені завдання, відповідно темі, а саме аналіз, розробка та дослідження, послідовно виконані.

3. Зміст кожного розділу роботи
Кожен розділ логічно пов'язаний та послідовно структурований. Теоретична частина дозволяє сформувати основу для проєктування, подальшого дослідження та експериментування.

4. Оцінка розробленого методу та його практична цінність
Розроблений метод генерації музичних композицій дозволяє створювати музику з урахуванням ритмічної та гармонійної логіки. Метод є гнучким, дозволяє генерувати музику як з нуля, так і на основі заданого треку. Метод підтримує різні стилі інструментів і зберігає міжтрекову узгодженість. Це відкриває можливості для практичного застосування у створенні фонової музики для відео, ігор, мультимедіа.

5. Якість оформлення кваліфікаційної роботи бакалавра
Робота виконана відповідно нормам та вимогам, структурно повна, містить усі необхідні розділи, відповідні дослідження.

6. Недоліки кваліфікаційної роботи бакалавра
Недоліком методу є велика потреба у ресурсах, адже метод генерації музики хоч і подібний до генерації тексту, але дещо відрізняється, від чого він і потребує більше потужностей для генерації мелодій.

7. Загальний висновок (допускається чи не допускається до захисту), та оцінка на яку заслуговує кваліфікаційна робота.
Враховуючи рівень виконання та забезпечення усіх необхідних вимог, робота може бути допущена до захисту. Рекомендована оцінка ***«відмінно»****.*

Рецензент *доц., кафедри ТТЗ, к.техн.наук Браворська Н.Г.*

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
МОН УКРАЇНИ

Кафедра комп'ютерних наук

ВІДГУК НАУКОВОГО КЕРІВНИКА

на кваліфікаційну роботу бакалавра

студента *гр. КН-21-2 Андрія ПРИЙМИ*

за темою *Метод створення музичних композицій на основі технологій генеративного штучного інтелекту*

1. Актуальність теми

Актуальним завданням, яке потребує аналізу і досліджується у даній роботі, є розробка методів автоматизованого створення музичних композицій за допомогою генеративного штучного інтелекту. В умовах стрімкого розвитку технологій машинного навчання та зростаючого попиту на інструменти творчої автоматизації, необхідно передбачити застосування програмних модулів, які дозволяли б генерувати якісний музичний контент різного стилістичного та емоційного спрямування. Розробка такого методу є актуальною задачею комп'ютерних наук.

2. Відповідність роботи предметній області Стандарту спеціальності 122 Комп'ютерні науки

За стандартом, а саме описом предметної області, об'єктами вивчення та діяльності є математичні, інформаційні, імітаційні моделі реальних явищ, об'єктів, систем і процесів та методи і технології отримання, зберігання, обробки, передачі та використання інформації. Метою роботи саме є розробка методу створення музичних композицій на основі генеративних нейронних мереж та розробка інформаційної системи реалізації вказаного методу. При вирішенні поставленої задачі використано математичні моделі, методи машинного навчання та алгоритми обробки музичних даних. Тому результати виконання кваліфікаційної роботи бакалавра відповідають стандарту бакалавра спеціальності 122 -- Комп'ютерні науки.

3. Професійні та особистісні якості бакалавра

При роботі над кваліфікаційною роботою бакалавра Прийма Андрій проявив себе кваліфікованим фахівцем та дисциплінованим студентом, вчасно виконуючи поставлені етапи дослідження. Як в процесі написання пояснювальної записки, так і при розробці прикладного програмного забезпечення проявив достатні для одержання успішного результату компетентності та результати навчання. Опанував професійні навички за напрямком «Комп'ютерні науки».

4. Ступінь самостійності під час виконання кваліфікаційної роботи

Одержані в роботі результати є наслідком особистої діяльності студента, який самостійно виконував всі поставлені задачі, включаючи проєктування архітектури нейронної мережі та реалізацію програмної системи.

5. Ступінь оволодіння методами дослідження

При реалізації кваліфікаційної роботи показав достатній рівень компетентностей та володіння необхідними інструментами та технологіями, зокрема TensorFlow, Python, методами глибокого навчання та метриками оцінки якості генерованого контенту.

6. Повнота та якість розкриття теми роботи

Тема роботи в повній мірі обґрунтована й розкрита, проведено аналіз актуальності та відомих рішень в межах обраної теми, поставлені завдання виконані, розроблено програмне забезпечення для валідації та верифікації запропонованого методу генерації музичних композицій.

7. Логічність, послідовність, аргументованість, літературна грамотність викладення матеріалу

Структура роботи та послідовність викладення логічні та відповідають поставленій меті. Викладення матеріалу послідовне, аргументоване, літературно грамотне з належним рівнем технічної деталізації.

8. Можливість практичного застосування кваліфікаційної роботи бакалавра, окремих її частин

Розроблений у роботі метод та його програмна реалізація може бути використана музикантами, композиторами та розробниками мультимедійного контенту для автоматизації творчих процесів та генерації музичного матеріалу.

9. Висновок про можливість допуску кваліфікаційної роботи бакалавра до захисту, на яку оцінку заслуговує робота

*Враховуючи високий рівень виконання та забезпечення усіх необхідних вимог, робота може бути допущена до захисту. Рекомендована оцінка «**відмінно**» .*

Керівник _____ *д.т.н., проф. каф. КН Едуард МАНЗЮК*