

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)


Кафедра Медіасистеми та технології
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)
(рівень вищої освіти)

Дослідження використання штучного інтелекту
для автоматичного створення обкладинок на базі текстового змісту
(тема)


Виконав:
студент 2 курсу, групи КТСВПВм-22-1


Литвиненко С.В.

Спеціальності 186 Видавництво та поліграфія

Тип програми Освітньо-професійна

Освітня програма
Комп'ютерні технології та системи
видавничо-поліграфічних виробництв

Керівник  проф. Манаков В.П.

Допускається до захисту
Зав. кафедри МСТ

(підпис)

Дейнеко Ж.В.
(прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет _____ Комп'ютерних наук
Кафедра _____ Медіасистеми та технології
Рівень вищої освіти _____ другий (магістерський)
Спеціальність _____ 186 Видавництво та поліграфія
Тип програми _____ Освітньо-професійна
Освітня програма _____ Комп'ютерні технології
та системи видавничо-поліграфічних виробництв
(шифр і назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри МСТ _____
(підпис)

« 30 » жовтня 2023 р.

**ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ**

студентові _____ *Литвиненко Сергію Володимирович*
(прізвище, ім'я, по батькові)

1. Тема роботи _____ *Дослідження використання штучного інтелекту для
автоматичного створення обкладинок на базі текстового змісту*

затверджена наказом по університету від _____ 27 жовтня 2023 р. № 1249 Ст

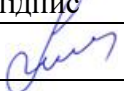
2. Термін подання студентом роботи до екзаменаційної комісії _____ 25 січня 2024 р.

3. Вихідні дані до роботи
*Національні та міжнародні стандарти створення обкладинок; Методи та принципи
генерації зображень; Методи та принципи реферування тексту.*

4. Перелік питань, що потрібно опрацювати в роботі
*Аналіз літератури відповідно до завдань дослідження; Методи і алгоритми для створення
зображень; Використання нейромереж у дизайні; Генеративні моделі; Дифузійні моделі;
Екстрактивні методи реферування; Абстрактні методи реферування; Аналіз інструментів
для програмної реалізації; Аналіз рішень для генерації зображень; Попередня обробка
тексту; Створення запиту з описом зображення; Практична частина; Економічна частина;
Висновки.*

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних
ілюстрацій
*Рисунок 2.2 – Будівельний блок генеративної змагальної мережі; Рисунок 2.4 – Генератор
DCGAN; Рисунок 2.5 – Дискримінатор DCGAN; Рисунок 2.7 – Усунення шумів;
Рисунок 2.8 – Ланцюг Маркова; Рисунок 2.10 – Алгоритм ранжування; Рисунок 4.6 –
Згенерованні обкладинки; Рисунок 3.8 – Результат в Midjourney; Рисунок 3.10 –
Результат в Stablediffusion; Економічна частина; Висновки.*

6. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Основна частина	проф Манаков В.П.		20.01.24
Економічна частина	ас. Помогалова Н.В.		23.01.24

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз літератури за темою дослідження	04.01.24	Виконано
2	Аналіз проблем та постановка задачі дослідження	06.01.24	Виконано
3	Аналітичний огляд літературних джерел	08.01.24	Виконано
4	Планування експерименту	09.01.24	Виконано
5	Проведення експериментального дослідження	12.01.24	Виконано
6	Економічна частина	13.01.24	Виконано
7	Оформлення пояснювальної записки	14.01.24	Виконано
8	Оформлення графічної частини	15.01.24	Виконано

Дата видачі завдання 30 жовтня 2023 р.

Студент


_____ (підпис)

Литвиненко С.В.

Керівник роботи


_____ (підпис)

проф. Манаков В.П.
_____ (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 85 с., 10 табл., 41 рис., 20 джерел.

DATA SCIENCE, MACHINE LEARNING, SUMMARIZATION, IMAGE GENERATION, GENERATIVE ADVERSARIAL NETWORK, DIFFUSION MODEL.

Метою даного дослідження є аналіз і реалізація алгоритмів та моделей штучного інтелекту, спроможних автоматично створювати візуально привабливі обкладинки для різних видів текстового контенту.

Методи дослідження: ретельний аналіз існуючих рішень та методів автоматичного створення обкладинок з використанням штучного інтелекту; розробка алгоритмів, здатних адаптуватися до різного за обсягом текстового контенту.

У результаті виконання кваліфікаційної роботи була розроблена програма на мові програмування Python, що дозволяє автоматично створювати обкладинки на базі текстового змісту.

ABSTRACT

Explanatory note to the diploma thesis contains: 85 p., 10 tabl., 41 pic., 20 sources.

DATA SCIENCE, MACHINE LEARNING, SUMMARIZATION, IMAGE GENERATION, GENERATIVE ADVERSARIAL NETWORK, DIFFUSION MODEL.

The purpose of this study is to analyse and implement artificial intelligence algorithms and models that can automatically create visually appealing covers for different types of textual content.

Research methods: a thorough analysis of existing solutions and methods for automatically creating covers using artificial intelligence; development of algorithms that can adapt to text content of different sizes.

As a result of the thesis, a Python program was developed that allows for the automatic creation of covers based on textual content.

ЗМІСТ

	С.
ВСТУП.....	8
1 АНАЛІЗ ЛІТЕРАТУРИ ВІДПОВІДНО ЗАВДАНЬ ДОСЛІДЖЕННЯ	10
1.1 Роль і види зображальних матеріалів у журналі та їхній дизайн.....	10
1.2 Використання нейромереж у дизайні.....	19
1.3 Постановка задачі дослідження.....	21
2 МЕТОДИ І АЛГОРИТМИ ДЛЯ СТВОРЕННЯ ЗОБРАЖЕНЬ	23
2.1 Генеративна модель.....	23
2.2 Дифузійна модель.....	33
2.3 Реферування.....	36
2.4 Екстрактивні методи реферування	37
2.4.1 Методи на основі графів	38
2.4.2 Алгоритм ранжирування сторінок.....	39
2.4.3 Завдання вилучення ключових слів.....	41
2.4.4 Завдання на виділення речень.....	42
2.5 Абстрактні методи реферування	43
2.5.1 Застосування послідовно-послідовних ШНМ	43
2.5.2 Мережі кодерів і декодерів	44
2.5.3 Механізм уваги	48
2.5.4 Видобування ключових слів	50
3 АНАЛІЗ ІНСТРУМЕНТІВ ДЛЯ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ	55
3.1 Вибір мови програмування	55
3.2 Вибір IDE	58
3.3 Аналіз рішень для генерації зображень	60
3.3.1 Midjourney - фотореалістична генерація зображень.....	60
3.3.2 DALL-E - створення складних та образних зображень.....	62
3.4 Порівняння інструментів генерації зображення	66
4 ПРАКТИЧНА ЧАСТИНА.....	71

4.1 Попередня обробка тексту	71
4.2 Реферування тексту	71
4.3 Створення запиту	72
4.4 Генерація зображення	73
4.5 Результати роботи програми.....	73
5 ЕКОНОМІЧНА ЧАСТИНА	75
5.1 Характеристика науково-дослідного рішення	75
5.2 Етапи виконання НДР, їх трудомісткість та заробітна плата	75
5.3 Розрахунок одноразових витрат на розробку НДР.....	78
5.4 Оцінка результатів науково-дослідної роботи.....	81
ВИСНОВКИ	83
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	84

ВСТУП

Нейромережі – одне з напрямлень штучного інтелекту, ціль якого – змоделювати аналітичні механізми, здійснювані людським мозком. Завдання, які вирішує типова нейромережа – класифікація, передбачення та розпізнавання. Також існує багато нейромереж, які на основі зібраної інформації можуть створювати власні елементи будь-то графічні зображення, фото, текстова інформація, тощо.

Саме такі нейромережі будуть розглянуті в дослідженні, які стисло стосуються дизайну, такі нейромережі можуть розроблювати логотипи, шрифти, обкладинки журналів, графічні та фото зображення.

Актуальність теми. З технологіями такі як нейромережі, які поступово стають доступними кожному, можливо об'єднати роботу в багатьох галузях, деякі з них можуть повністю, або частково виконувати більшу частину роботи. Наприклад як створювати тексти, обробляти фотографії, генерувати зображення по опису і так далі.

Через це не користуватись ними немає сенсу. Багато з них звичайно потребують доопрацювання, але вже з їх допомогою можливо об'єднати і скоротити хоча б чорнову роботу, на яку можна потратити багато часу.

Мета роботи – дослідження нейронних мереж, їх практичного використання, показати їх роботу в дизайні, виявити недоліки і переваги, порівняти роботу з нейромережами і без їх використання. Дізнатися наскільки є ефективна робота нейронних мереж в генерації зображень.

Наскільки новітні технології в виді нейронних мереж являються корисними і як вони можуть впливати на роботу дизайнера. За основу взятий дизайн обкладинки журналу, так як в ньому використовується багато констант, як колірна палітра, логотип, шрифти.

Все це дає змогу протестувати роботу нейромереж, виявити їх недоліки, показати наскільки вони ефективні у роботі дизайнера і наскільки вони мають значення.

Нейромережі розглядаються як допоміжний інструмент для дизайнера. Дослідження роботи полягає в тому, щоб виявити наскільки корисними можуть бути нейромережі для створення обкладинок.

З їх активним розвитком, на даний момент нейромережі спроможні створювати окремі елементи графічного дизайну, а також запобіжні інструменти для прискорення роботи дизайнера над механічними задачами.

1 АНАЛІЗ ЛІТЕРАТУРИ ВІДПОВІДНО ЗАВДАНЬ ДОСЛІДЖЕННЯ

1.1 Роль і види зображальних матеріалів у журналі та їхній дизайн

На журнальному ринку завжди точилася боротьба за «місце під сонцем», а в наші дні - в період горезвісної економічної кризи - ця боротьба загострилася. Крім того, за останні роки друковані ЗМІ потіснила лавина електронної інформації: телебачення та Інтернет. Журнали намагаються вижити, і виживають найсильніші, тобто ті, у кого усталений імідж, велика читацька аудиторія, напрацьовані зв'язки з рекламодавцями.

Величезну роль у цій боротьбі відіграє імідж редакційного колективу. Він залежить від позицій і поглядів, тематики та змісту, а також від особливостей подачі матеріалів - дизайну і верстки. І, звичайно, в різних виданнях формування номера розраховане на свою цільову аудиторію, яка шукає «свій» журнал, що відповідає її смакам та інтересам.

Читачам та засновникам журналів догодити дуже складно. Мерехтливі картинки на екранах телевізорів і комп'ютерів роблять свою справу - читач насамперед звертає увагу на фотоілюстрації. У зв'язку з цим, працівникам журналів доводиться постійно стежити за розвитком електронних ЗМІ, мати електронну версію свого видання в Інтернеті, а найголовніше - поліпшувати і оновлювати форму і зміст свого друкованого видання, в якому дедалі більшу роль починає відігравати використання фотографії.

Сьогодні фотоілюстрації стали невід'ємною частиною графічного вигляду журнальної сторінки і самі по собі мають художню цінність. Вони несуть додаткову інформацію, передають настрій або атмосферу подій, описаних у статті, привертають увагу до неї, розбивають текст, зупиняють мить і дозволяють вивчити її так, як не дозволив би цього зробити рухомий образ. Фотографія може виконувати в журналі різні функції, вона

супроводжує, ілюструє текстовий матеріал - кореспонденцію, статтю, хроніку і нерозривно пов'язана з текстом.

Ілюстрації – неодмінний елемент будь-якого видання. «Опис за допомогою слова не порівняти з адекватністю передавання наочного, тобто сприйнятого зорового уявлення про предмети через їх зображення». Проте термін «ілюстрація» не зовсім точний. Прийшов він у теорію журнального дизайну з книжкового оформлення, де означає зображення, яке пояснює, доповнює, ілюструє текст: «у точному, або вузькому, значенні - це виявлення засобами образотворчого мистецтва образного змісту, яке за допомогою слова відображається в літературі» [5].

Таке тлумачення цілком відповідає характерові зображального матеріалу в книзі або в журналі, але в журналі ілюстрація має нове призначення – повідомлення, що несе основний зміст. Значна частина журнальних ілюстрацій (до 60%) – самостійні матеріали. Важливість зображального матеріалу в періодичних виданнях підкреслює Б. Черняков: перевага слова – здатність абстрагуватися від конкретного, перевага зображення – його подібність до об'єкта відображення [3].

Якщо проаналізувати публікації з ілюстраціями в журналах «Elle», «Cosmopolitan», «Forbes», «Time», «Vogue», за 2023 рік, виявляється, що більша частина фотографій і малюнків ілюструє текст, але близько 13% – незалежні матеріали (фотографії, малюнки, карти, діаграми) (рис. 1.1, рис. 1.2). Текст у таких матеріалах пояснює і коментує зображення, яке передає основний зміст. Лише 20% зображальних елементів на сторінці журналу є однорідними, подібними за тематикою та призначенням, хоча за змістом вони не обов'язково пов'язані. Такими бувають частіше портрети або зображальний ряд нарису. Нариси також більшою частиною передають основний зміст, який коментується текстовим матеріалом.



Рисунок 1.1 – Публікації популярних журналів,
зображення ілюструють текст



Рисунок 1.2 – Публікації популярних журналів
з незалежними матеріалами на обкладинці

Отже, термін «ілюстрація» відображає характер фотографій і малюнків лише в теорії книжкового оформлення, але стосовно контенту журналів доцільно використовувати термін «зображальний матеріал» («зображальні елементи»). Таке тлумачення дає змогу розглядати зображальні елементи у двох аспектах: як елемент оформлення і як підсистему, тобто окрему публікацію, самостійне інформаційне повідомлення. «У журналістських формах єдності тексту та зображення фотографія стає рівноправним засобом передачі змісту» [3]. Тобто, зображальні елементи є не тільки елементом зовнішньої форми журналу, а й її змістовим наповненням.

А це потребує одночасного розгляду, оскільки оформлювальні функції й зміст взаємопов'язані і мають загальне призначення: характер відображення зображальних елементів впливає на їх сприйняття читачем, привертає увагу до публікації, а змістове наповнення – задовольняє потребу в інформації.

Зображальна мова аналогічна мові слова, оскільки зображення відображає об'єкти зовнішнього світу, тому й належить до семіотичної знакової системи - на відміну від семантичної системи, в якій елементи не співвідносяться з позамовними даними, а їх значущість виникає в процесі мовної діяльності. У фотодобірках, композиціях зображальні елементи створюють послідовний ланцюг і, отже, внутрішньо зв'язані не лише з текстом та іншими архітектонічними елементами, а й між собою. Такого роду зображальний ряд створює другу, подібну до текстової, інформаційну систему [3]. У цьому сенсі зображальний матеріал відрізняється від інших архітектонічних елементів - він або має другорядне значення щодо тексту, або є єдиним виразником змісту.

Каркас архітектонічної будови журналу складається з рубрикаційних членувань (структурний ряд) та зображальних елементів (наочно-значеннєвий ряд).

Велика кількість зображальних елементів різних видів значно ускладнює структуру журналу саме через їх подвійне змістове навантаження. Композиційно-просторові та конструктивні завдання також розширюються при складному оформленні врізів, заголовків. Деякі сторінки сучасних журналів взагалі побудовані переважно на зображальних елементах.

До теми ілюстративного оформлення неодноразово звертався дослідник оформлення друкованих видань Н. Гончарова, яка називає ілюстрації «виразниками логічно-сислового членування і супідрядності частин літературного твору» [7, с.62].

Наприклад у жіночих журналах типу «Elle», «Cosmopolitan», «Vogue» таке співвідношення: 80% фотографій (і малюнків) – самостійні матеріали. Якщо рахувати не ілюстрації, а публікації, у складі яких є зображальні елементи (скажімо, фоторепортаж – це одиниця виміру, хоча в ньому шість

фотографії), то і тоді виявиться, що половина знімків і малюнків ілюструє текст, решта – самостійні твори (фотозамітка, фоторепортаж, фотонарис), а текст пояснює і коментує ці зображальні матеріали, що несуть основний зміст.

Важко уявити собі пресу без фотографії. Як у паперовій пресі, так і в інтернет-виданнях фотографії супроводжують текст, доповнюють і візуалізують тему, про яку йдеться. Часто саме гарно зроблена і підібрана фотографія визначає рівень журналістського матеріалу. Це не означає, що ті, хто пише, не мають дбати про зміст і стилістику статті. Варто тільки підкреслити рівнозначну роль слова і зображення в медіях.

Як і будь-який предмет масового виробництва, журнал підпорядкований певним технічним нормам і стандартам.

У межах цих норм він набуває власного обличчя, що досягається виразними засобами, елементами оформлення, раз прийнятою структурою – через модель. Формати, текстові та зображальні блоки так співвідносяться, щоб з окремих, не пов'язаних між собою матеріалів утворилося художнє змістове ціле – журнал [8].

Оформлення журналу як художньої форми, що поєднує зміст, призначення, структуру, дизайн, функції видання, лежить в основі архітекtonіки журналу.

Головні параметри архітекtonіки спільні для будь-якого друкованого видання: книжкового, журнального, газетного. Щоправда є особливості, зумовлені відмінністю типів видань за характером їх функціонального призначення, розміром, форматом, обсягом, тематикою, наявністю зображального матеріалу, технологією макетування та верстки. Кожний друкований твір має свої відмінності на фоні загальних архітекtonічних властивостей, утворених за законами корисності, функціональності, ергономічності, краси.

Основними характеристиками архітекtonіки видання є матеріально-просторова структура і зовнішнє оформлення. Тобто архітекtonіка видання базується на двох головних параметрах - внутрішньому змістові і формі

вираження. Безперечно, матеріально-просторова структура визначає найважливіші риси журналу та її форми – загальний вигляд, формат, основні пропорції і розміри. Тому у формуванні видання функціонально і композиційно-естетично структура є головною, а зовнішнє оформлення – підпорядкованим, але активним і важливим.

Внутрішній зміст друкованого видання є основою журналу як самостійної системи, що являє собою матеріально-просторову структуру і включає елементи, які відображають найважливіші ознаки цієї сутності: призначення видання, об'ємно-просторову побудову, взаєморозташування і взаємодію елементів (формат, кількість і композиція текстових та зображальних матеріалів, постійні та тимчасові рубрики, добірки), форму подання інформації, конструктивну організацію, вплив окремих частин видання одна на одну.

Зовнішні композиційні зв'язки поєднують елементи сторінки або розвороту журналу через контраст великого і малого, гармонію шрифтів і спричинення ритму, відображеного у кількості і ширині шпальт, вертикальних і горизонтальних лінійках, композиції заголовків і текстів. У журналах матеріали, як правило, двовимірні, мають ширину і висоту, але зустрічаються умовно багатомірні – такі, що існують у трьох площинах, продовжені із сторінки на сторінку [4].

Форма вираження - друга сутність видання, яка сприймається читачем візуально; «зовнішній», матеріально-просторовий прояв змісту. Тобто зовнішнє оформлення є основою для матеріалізації внутрішнього змісту журналу, її типологічної специфіки й індивідуального образу. Воно відіграє роль символу, через який журнал ідентифікується у сприйнятті читача. На цій властивості форми засновуються об'єктивні можливості та значущість художніх засобів і процесу комплексного художньо-технічного конструювання й архітектонічного моделювання видання.

Форма має два органічно взаємозалежні компоненти: внутрішню організацію – структуру і зовнішню форму – елементи оформлення, які

утворюють пластичну оболонку, що сприймається візуально. Форма видання – це передусім форма просторової організації журнальних матеріалів. Іншими словами, це історично сформована розмірна модель, що забезпечує журналі виконання її соціальних функцій.

Внутрішня організація журналу визначається розмірно-просторовими і графічними параметрами, які закладені в архітектонічні елементи та засоби оформлення. Зовнішня форма є сукупністю елементів оформлення: шпальти, формат, членування декоративними елементами та засобами виділення – виворотки, плашки, зображення, виступи, врізи, білий простір, лінійки, фігурні позначки, ініціали, світлотіньові ефекти.

Внутрішня структура формується як складник концепції журналу, сприймається в процесі самого читання. Її елементи пов'язані зі змістом журналу, її специфікою й індивідуальністю. Тому внутрішня структура визначає застосування архітектонічних елементів і риси зовнішньої структури (обсяг, графічну організацію, періодичність), а елементи зовнішньої структури формують межі, в яких внутрішня структура розміщується й оформлюється (формат сторінки, поля) [5].

Для визначення розмірно-просторової та візуально-естетичної організації журналу використовується термін «об'ємно-просторова структура», в якому чітко проявляються матеріальний, функціональний і композиційний аспекти будь-якого художньо-конструкторського й архітектурного твору – обсяг і відкритий простір. Обсяг передається через формат видання, формат набору, кількість і ширину шпальт, шрифти, заголовні компоненти, кількість зображень.

До основних закономірностей архітектоніки, що поєднують архітектонічну чіткість видання з досконалістю форми вираження через взаємозв'язки між геометрією (структурою і пластикою), конструктивністю (макетом), розмірами (форматом), емоційно-естетичною виразністю й економічною ефективністю, відносять:

- складність геометрії форми та просторову організацію, конструктивність, декоративність, естетичну виразність видання;
- цілісність матеріально-просторової організації форми вираження, замкненість, взаємозалежність усіх елементів і композиційну цілісність, закінченість видання;
- упорядкованість структури та стабільність використання архітектонічних елементів, єдність, закінченість, виразність оформлення. Досконалість і економічна ефективність, конструктивність і емоційно-естетична гострота та виразність форми видання залежать не тільки від кількості використаних елементів, а й від раціональності його просторової організації;
- зміну розмірів форми при збереженні її геометрії пропорційним збільшенням або зменшенням усіх параметрів.

На архітектоніку журналу впливає також особливість сприйняття читачем її структурних одиниць: верхня частина є активнішою порівняно з нижньою, так само як перша та остання сторінки відносно внутрішніх. У цьому проявляється зв'язок форми та змісту: важливі матеріали виносяться на чільні місця, посилюються декоративними елементами.

Тобто оформлення журналу пов'язане зі змістом як безпосередньо, з урахуванням соціальної, політичної значущості окремих матеріалів, так і опосередковано – через форму просторової організації матеріалів, з урахуванням різноманітних частин цієї форми.

Основні залежності архітектоніки проявляються у функціональній й утилітарній досконалості, гармонійному взаємозв'язку форми і змісту, художньому осмисленні, інформативності форми, а також від критерію архітектонічності.

Сучасні українські та зарубіжні дослідники також активно досліджують сутність поняття «контент».

Так, на думку В. Різуна, контент у масовій комунікації – особи, групи осіб, суспільство, які використовують засоби комунікації, з метою досягнення одного або кількох спеціальних ефектів [6].

Контент масової комунікації відрізняється від контенту персональної або групової комунікації, бо повідомлення в сфері масової комунікації малоперсонізоване, спеціалізоване, більш швидко поширюється і має зручніші канали проходження. Дослідник також звертає увагу на те, що виокремлюють шість завдань або категорій контенту масової комунікації: розваги, новини, коментарі, виховання, зв'язки з громадськістю і реклама [4].

З появою так званих «нових медіа» дослідження поняття контенту набуло великого поширення – з'явилося чимало версій розуміння контенту. Так, однією з них є така, де під контентом розуміють будь-яке інформаційно значуще наповнення інформаційної системи – тексти, графіка, мультимедіа; частину повідомлення, яка не опрацьовується й не змінюється в процесі передачі; частину даних документа.

Контент є базовою категорією мови сучасних медіа – це техніко-технологічна та гуманітарна медіалінгвістична технологія для створення сучасної мови ЗМІ та високотехнологічних електронних послуг [4].

Проаналізувавши думки дослідників щодо визначення поняття «контент» та переглянувши словниковий фонд, можемо виокремити спільні моменти: по-перше, контент – внутрішня сутність предмета, явища; по-друге, контент – це результат певної діяльності (психолінгвістичної, комунікативної тощо).

Контент у широкому сенсі є комунікативною подією, яка відбувається між спостерігачем та іншою особою чи групою осіб у процесі комунікативної дії у певному тимчасовому, просторовому та іншому контекстах. Ця комунікативна дія може бути усною, письмовою, мати вербальні та невербальні чинники. Виокремлюють вербальний чинник комунікативної дії, говорячи про неї як про «текст» або «розмову». Звідси термін «контент» означає або завершений, або такий, дія якого триває, «продукт» комунікативної діяльності, його письмовий або мовний результат, який інтерпретується аудиторією [4, с. 135].

Актуальною для сучасних видавців та дизайнерів є проблема формування задуму оформлення, який ураховує всі необхідні компоненти

майбутньої публікації, які б задовольняли потреби читача та видавця. Перед тим, як почати розробляти проект оформлення, необхідно дати відповіді на питання, від яких залежить не лише форма подання інформації, але й сам зміст видання, і, зрозуміло, його успішність: мета, призначення, вид видання, читацька аудиторія та її інтереси, очікуване враження, відповідність публікації певним вимогам, стилю, основні елементи макету, постійні елементи, ілюстративний ряд, види текстових матеріалів, друкарські процеси, терміни виготовлення, способи розповсюдження, витрати на видання і можливі шляхи отримання фінансів.

1.2 Використання нейромереж у дизайні

Нейронні мережі є інструментом графічного дизайну, що швидко вдосконалюється. Проте значна кількість продукції графічного дизайну потребує спеціалізованого програмного інструментарію.

Серед численної кількості сучасних нейронних мереж для роботи з контентом графічного дизайну найбільш придатними та до певної міри універсальними є програми Midjourney, Stable Diffusion, DALL-E 2. Перевагою застосування нейронних мереж є суттєве прискорення процесу генерування візуального контенту і можливості комбінувати різні програми та доповнювати й вдосконалювати результати кожної з них.

До переваг нейронних мереж також зараховують можливість створення візуального контенту за текстовим запитом, тобто контактеру не потрібно навчатися малювати, опановувати анатомію людини та створювати 3D-моделі чи вивчати 3D-скульптуринг, що потребує найбільшої витрати енергії, часу та ресурсу.

Нейронні мережі дають змогу перестрибнути сходинки професійного зросту. З одного боку, це є перевагою, а з іншого – може бути й суттєвим недоліком. Унаслідок заощадження часу та енергії професійний розвиток стає другорядним для митця – цей фактор

призводить до зниження конкурентоспроможності й обмежує рамки творчості в нейронних мережах.

Недоліком є переважно монологістичне спілкування контактера з мережею. У більшості нейронних мереж запит для створення картин можна писати тільки англійською мовою. Ще один недолік результатів творчості нейронних мереж полягає в тому, що відсутній власне творчий процес, керований уявою митця та образами, що, можливо, існують у свідомості контактера з мережею. Тобто після вводу запиту художник (контактер) отримує кінцевий результат, що не підлягає подальшому коригуванню. Це звужує бачення митця (контактера) й позбавляє його первинних артефактів творчого процесу. Наприклад, при створенні ілюстрацій або концепт-арту дизайнер може зберігати проміжні результати роботи та коригувати її довершений варіант. У роботі з тривимірними графічними редакторами, при створенні 3D-сцени завжди є можливість зберігати численну кількість первинних варіантів файлів — артефактів у вигляді 3D-моделей і текстур до них. У разі потреби можна повертатись, змінювати властивості об'єктів (розміри, текстури, освітлення тощо) і досягати оптимального результату. У творчості нейронних мереж ці артефакти відсутні. Цей чинник знецінює роботу, тому що без наявності вихідних артефактів, ескізних варіантів та етапних коригувань неможливо довести авторство роботи.

Наразі для роботи з продукцією графічного дизайну найбільш придатними та універсальними є такі програми: Midjourney, Dalle-E 2, Stablediffusion.

Midjourney – нейронна мережа, що створює ілюстрації, концепт-арти, персонажі, ландшафти, архітектури тощо. Створення візуального контенту в цій мережі відбувається за текстовим запитом, який пише людина. Тому важливо підготувати деталізований і розгорнутий текстовий запит, що

буде впливати на якість отриманого візуального результату. Ця нейронна мережа найбільш придатна для створення ілюстрацій або колажів і постерів, зокрема в стилі видатних художників. Проте Midjourney ще має проблеми з пропорціями й анатомією людини (люди на картинках не реалістичні), з композиціями та поєднанням кількох об'єктів одночасно.

Dalle-E 2 – нейронна мережа, що працює подібно до Midjourney, але має значно менше можливостей для кастомізації зображення. Інтерфейс Dalle-E 2 за зручністю і логічністю візуального сприйняття гірший за інтерфейс Midjourney.

Ця мережа може створювати зображення з вищою роздільною здатністю і швидше, ніж більшість аналогічних програм [9].

1.3 Постановка задачі дослідження

Аналізуючи дослідження в сфері дизайну, можна припустити, що за невеликий проміжок часу нейромережі, що генерують зображення, захоплять інтернет з неминучою швидкістю. Їхні дивовижні можливості вражають уяву і змушують задуматися про майбутнє таких технологій.

Працюють нейромережі за одним принципом: вони здатні згенерувати картину за текстовим запитом (Text to Image AI), але, незважаючи на схожу логіку, результати виходять різні.

Слід очікувати, що при глибшому впровадженні новітніх комп'ютерних технологій, а саме штучного інтелекту нейромереж, у створення дизайн-концепцій UI на основі ілюстрацій, суттєво підніметься рівень використання цих засобів у розвитку веб-дизайну.

Метою даної роботи є дослідження використання штучного інтелекту для автоматичного створення обкладинок на базі текстового змісту. Для досягнення поставленої мети необхідно вирішити такі завдання:

- провести аналіз спеціальної науково-технічної літератури на тему даного дослідження;

- провести огляд і аналіз нейромереж;
- обрати та обґрунтувати нейромережу для генерації зображення на базі текстового змісту;
- розробити набір рекомендацій щодо генерації зображень;
- обрати та обґрунтувати методи реферування тексту;
- провести оцінку якості результату.

Отже, необхідно провести дослідження з визначення найкращої нейромережі для створення обкладинок за обраними критеріями.

2 МЕТОДИ І АЛГОРИТМИ ДЛЯ СТВОРЕННЯ ЗОБРАЖЕНЬ

2.1 Генеративна модель

Генеративна модель – це потужний спосіб вивчення будь-якого типу розподілу даних за допомогою неконтрольованого навчання, який досяг величезного успіху всього за кілька років. Всі типи генеративних моделей спрямовані на вивчення істинного розподілу даних у навчальній вибірці, щоб генерувати нові точки даних з деякими варіаціями. Але не завжди можливо дізнатися точний розподіл наших даних в явному чи неявному вигляді, тому ми намагаємося змоделювати розподіл, який максимально схожий на справжній розподіл даних. Для цього ми можемо використати можливості нейронних мереж, щоб вивчити функцію, яка може наблизити модельний розподіл до справжнього розподілу.

Два найпоширеніші та найефективніші підходи – це варіаційні автокодері (Variational Autoencoders, VAE) та генеративні змагальні мережі (Generative Adversarial Networks, GAN). VAE спрямований на максимізацію нижньої межі ймовірності логарифму даних, а GAN – на досягнення рівноваги між генератором і дискримінатором. У цьому блозі я поясню, як працюють VAE і GAN, а також інтуїцію, що лежить в їх основі.

Ми знаємо, що можемо використовувати автокодер для кодування вхідного зображення у значно меншу розмірність, яка може зберігати приховану інформацію про розподіл вхідних даних.

Але у звичайному автокодері закодований вектор можна зіставити з відповідними вхідними даними лише за допомогою декодера. Він, безумовно, не може бути використаний для генерування подібних зображень з деякою варіативністю.

Щоб досягти цього, модель повинна вивчити розподіл ймовірностей навчальних даних. VAE є одним з найпопулярніших підходів до вивчення

складних розподілів даних, таких як зображення, за допомогою нейронних мереж у неконтрольованому режимі. Це імовірнісна графічна модель, заснована на байєсівському висновку, тобто модель має на меті вивчити основний розподіл ймовірностей навчальних даних, щоб вона могла легко вибирати нові дані з цього вивченого розподілу. Ідея полягає в тому, щоб вивчити низьковимірне латентне представлення навчальних даних, яке називається латентними змінними (змінні, які не спостерігаються безпосередньо, а виводяться за допомогою математичної моделі), які, як ми припускаємо, згенерували наші фактичні навчальні дані. Ці латентні змінні можуть зберігати корисну інформацію про тип вихідних даних, які повинна генерувати модель. Розподіл ймовірностей латентних змінних z позначається $P(z)$. Гауссівський розподіл обрано як попередній для вивчення розподілу $P(z)$, щоб легко відбирати нові точки даних під час виведення.

Інтуїція, що стоїть за цією оцінкою максимальної правдоподібності, полягає в тому, що якщо модель може генерувати навчальні вибірки з цих латентних змінних, то вона також може генерувати подібні вибірки з деякими варіаціями. Іншими словами, якщо ми виберемо велику кількість латентних змінних з $P(z)$ і згенеруємо x з цих змінних, то згенерований x повинен відповідати розподілу даних $P_{data}(x)$. Тепер у нас є два питання, на які ми повинні відповісти. Як зафіксувати розподіл латентних змінних і як інтегрувати рівняння 1 за всіма вимірами z ?

Очевидно, що вручну вказувати відповідну інформацію, яку ми хочемо закодувати у прихованому векторі для створення вихідного зображення, є нудним завданням. Натомість ми покладаємось на нейронні мережі для обчислення z лише з припущенням, що цей латентний вектор може бути добре апроксимований нормальним розподілом, щоб легко зробити вибірку під час виводу. Якщо ми маємо нормальний розподіл z у n -вимірному просторі, то завжди можна згенерувати будь-який вид розподілу за допомогою досить складної функції, а обернену до неї функцію можна використати для вивчення самих латентних змінних.

У рівнянні 1 інтегрування проводиться за всіма вимірами z і тому є нерозв'язним. Однак його можна обчислити за допомогою методів інтегрування Монте-Карло, що не так просто реалізувати. Тому ми застосовуємо інший підхід для наближеної максимізації $P_\theta(X)$ у рівнянні 1. Ідея VAE полягає в тому, щоб зробити висновок про $P(z)$, використовуючи $P(z|X)$, яку ми не знаємо. Ми виводимо $P(z|X)$ за допомогою методу, який називається варіаційне виведення, що по суті є оптимізаційною задачею в байєсівській статистиці. Спочатку ми моделюємо $P(z|X)$, використовуючи простіший розподіл $Q(z|X)$, який легко знайти, і намагаємося мінімізувати різницю між $P(z|X)$ і $Q(z|X)$, використовуючи метрику KL-дивергенції, щоб наша гіпотеза була близькою до істинного розподілу. За цим слідує багато математичних рівнянь, які я не буду пояснювати тут, але ви можете знайти їх в оригінальній статті. Але мушу сказати, що ці рівняння не дуже складно зрозуміти, якщо ви володієте інтуїцією, яка лежить в основі VAE.

Наведене вище рівняння має дуже гарну інтерпретацію. Термін $Q(z|X)$ - це, по суті, наша мережа кодерів, z - наше закодоване представлення даних $x(x \in X)$, а $P(X|z)$ - наша мережа декодерів. Отже, у наведеному вище рівнянні нашою метою є максимізація лог-правдоподібності розподілу даних при деякій похибці, заданій $D_{KL}[Q(z|X) \parallel P(z|X)]$. Легко помітити, що VAE намагається мінімізувати нижню межу $\log(P(X))$, оскільки $P(z|X)$ не можна обчислити, але член KL-дивергенції ≥ 0 . Це те саме, що максимізувати $E[\log P(X|z)]$ і мінімізувати $D_{KL}[Q(z|X) \parallel P(z|X)]$. Ми знаємо, що максимізація $E[\log P(X|z)]$ є оцінкою максимальної правдоподібності і моделюється за допомогою мережі декодерів. Як я вже казав раніше, ми хочемо, щоб наше латентне представлення було близьким до гаусівського, і тому ми припускаємо, що $P(z)$ дорівнює $N(0, 1)$. Виходячи з цього припущення, $Q(z|X)$ також має бути близьким до цього розподілу. Якщо ми припустимо, що це гаусівський розподіл з параметрами $\mu(X)$ і $\Sigma(X)$, то похибка, зумовлена різницею між цими двома розподілами, тобто $P(z)$ і

$Q(z|X)$, отримана за допомогою KL-дивергенції, призводить до розв'язку в замкненій формі, наведеному нижче.

Отже, функція втрат міститиме два доданки. Перший - це втрати при реконструкції входу у вихід, а другий - це втрати від KL-дивергенції. Тепер ми можемо навчати мережу за допомогою алгоритму зворотного поширення. Але є проблема, яка полягає в тому, що перший член залежить не тільки від параметрів P , але й від параметрів Q , але ця залежність не з'являється у наведеному вище рівнянні. Тож як здійснити зворотне поширення через шар, де ми випадково вибираємо z з розподілу $Q(z|X)$ або $N[\mu(X), \Sigma(X)]$ так, щоб P змогла декодувати. Градієнти не можуть проходити через випадкові вузли. Ми використовуємо трюк репараметризації, щоб зробити мережу диференційованою. Ми робимо вибірку з $N(\mu(X), \Sigma(X))$, спочатку виділяючи $\epsilon \sim N(0, I)$, а потім обчислюючи $z = \mu(X) + \Sigma^{1/2}(X) * \epsilon$. Слід зазначити, що крок прямого зв'язку ідентичний для обох мереж (лівої та правої), але градієнти можуть поширюватися тільки через праву мережу (рис. 2.1).

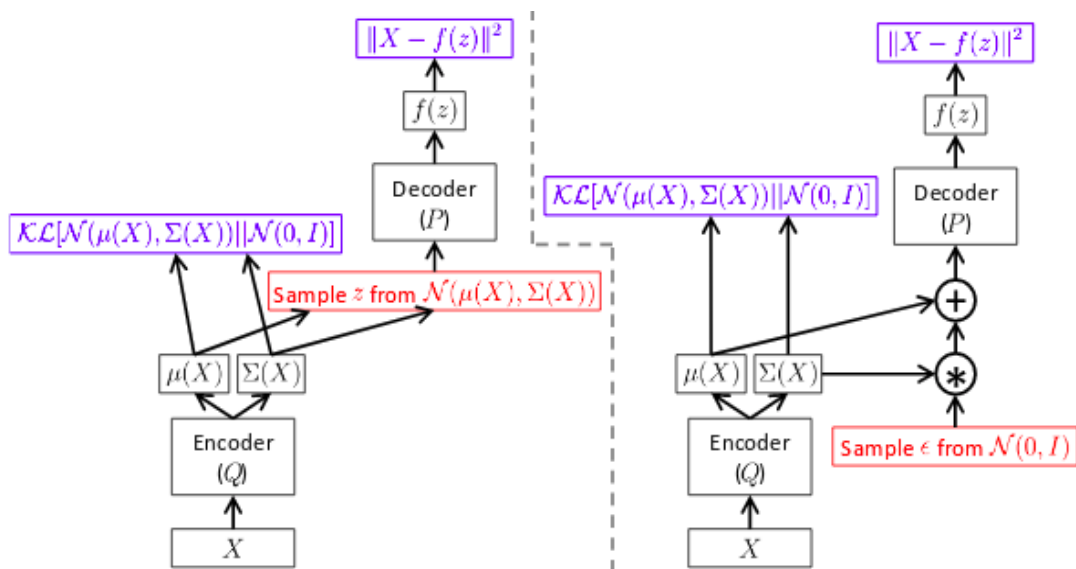


Рисунок 2.1 – Трюк репараметризації, що використовується для зворотного поширення через випадкові вузли

Під час виводу ми можемо просто взяти вибірку z з $N(0, 1)$ і подати її на декодерну мережу, щоб згенерувати нову точку даних.

Оскільки ми оптимізуємо нижню варіаційну межу, якість згенерованого зображення буде дещо нижчою, ніж у сучасних методів, таких як Генеративні Змагальні Мережі.

Найкращим у VAE є те, що він вивчає як генеративну модель, так і модель виведення. Хоча і VAE, і GAN є дуже цікавими підходами до вивчення базового розподілу даних за допомогою неконтрольованого навчання, але GAN дає кращі результати порівняно з VAE. У VAE ми оптимізуємо нижню варіаційну межу, тоді як у GAN такого припущення немає. Насправді, GAN не мають справи з явною оцінкою щільності ймовірності. Неспроможність VAE генерувати чіткі зображення означає, що модель не здатна вивчити істинний апостеріорний розподіл. VAE і GAN в основному відрізняються способом навчання. Тепер давайте зануримося в генеративні змагальні мережі.

Дивлячись на популярність генеративних змагальних мереж і якість результатів, які вони дають, я думаю, що більшість з нас погодиться з ним. Змагальне навчання повністю змінило те, як ми навчаємо нейронні мережі виконувати певні завдання. Генеративні змагальні мережі не працюють з явною оцінкою щільності, як варіаційні автокодери. Замість цього вони базуються на підході теорії ігор з метою знайти рівновагу Неша між двома мережами, генератором і дискримінатором. Ідея полягає в тому, щоб зробити вибірку з простого розподілу, наприклад, гаусівського, а потім навчитися перетворювати цей шум на розподіл даних за допомогою універсальних апроксиматорів функцій, таких як нейронні мережі.

Це досягається завдяки змагальному навчанню цих двох мереж. Модель генератора G вчиться вловлювати розподіл даних, а модель дискримінатора D оцінює ймовірність того, що зразок був отриманий з розподілу даних, а не з розподілу моделі. По суті, завданням генератора є створення природних зображень, а завданням дискримінатора – вирішити, чи

є зображення фальшивим, чи справжнім. Це можна уявити як гру міні-макс для двох гравців, де продуктивність обох мереж покращується з часом. У цій грі генератор намагається обдурити дискримінатор, генеруючи справжні зображення, наскільки це можливо, а дискримінатор намагається не бути обдуреним генератором, покращуючи свою дискримінаційну здатність. Нижче показано базову архітектуру GAN (рис. 2.2).

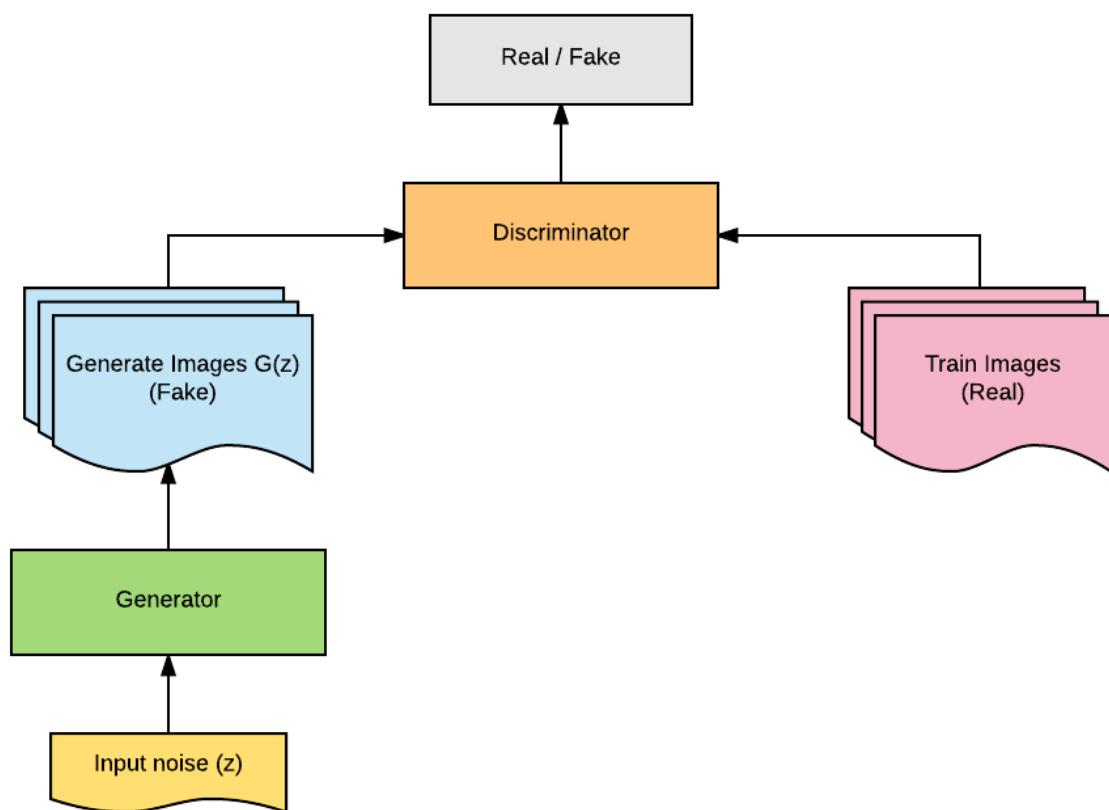


Рисунок 2.2 – Будівельний блок генеративної змагальної мережі

Ми визначаємо попередні змінні вхідного шуму $P(z)$, а потім генератор зіставляє їх з розподілом даних за допомогою складної диференційованої функції з параметрами θ_g . На додаток до цього, ми маємо ще одну мережу, яка називається Дискримінатор, яка приймає на вхід x і за допомогою іншої диференційованої функції з параметрами θ_d виводить єдине скалярне значення, яке позначає ймовірність того, що x походить з істинного розподілу даних $P_{data}(x)$.

У наведеному вище рівнянні, якщо на вхід дискримінатора надходить істинний розподіл даних, то $D(x)$ повинен видати 1, щоб максимізувати наведену вище цільову функцію w.r.t D , тоді як якщо зображення було згенеровано генератором, то $D(G(z))$ повинен видати 0, щоб мінімізувати цільову функцію w.r.t D . Це означає, що генератор повинен генерувати такі реалістичні зображення, які можуть обдурити дискримінатор. Останнє в основному означає, що G повинен генерувати такі реалістичні зображення, які можуть обдурити D . Ми максимізуємо наведену вище функцію w.r.t параметрів дискримінатора за допомогою градієнтного підйому і мінімізуємо ті ж самі параметри w.r.t генератора за допомогою градієнтного спуску. Але існує проблема в оптимізації завдання генератора. На початку гри, коли генератор ще нічого не навчився, градієнт, як правило, дуже малий, а коли він працює дуже добре, градієнти дуже високі (рис. 2.3). Але ми хочемо протилежної поведінки. Тому ми максимізуємо $E[\log(D(G(z)))]$, а не мінімізуємо $E[\log(1-D(G(z)))]$.

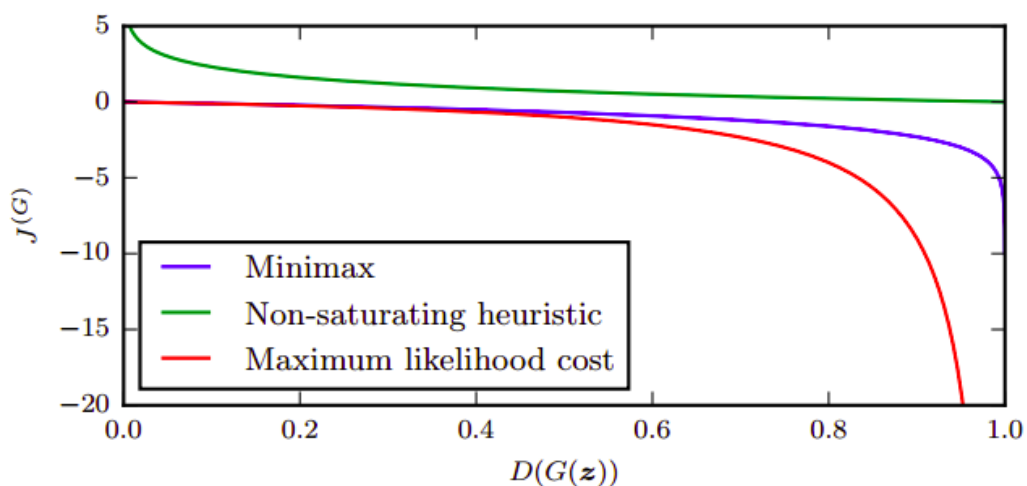


Рисунок 2.3 – Вартість генератора в залежності від реакції дискримінатора на згенероване зображення

Процес навчання складається з одночасного застосування стохастичного градієнтного спуску на дискримінаторі та генераторі. Під час навчання ми чергуємо k кроків оптимізації D та один крок оптимізації G на

міні-партії. Процес навчання зупиняється, коли дискримінатор не може відрізнити p_g від p_{data} , тобто $D(x, \theta_d) = \frac{1}{2}$, або коли $p_g = p_{data}$.

Однією з перших моделей GAN, що використовує згорткові нейронні мережі, була DCGAN, що розшифровується як Deep Convolutional Generative Adversarial Networks (Глибока згорткова генеративна змагальна мережа). Ця мережа приймає на вхід 100 випадкових чисел, взятих з рівномірного розподілу, і виводить зображення бажаної форми. Мережа складається з багатьох згорткових, деконволюційних та повністю з'єднаних шарів. Мережа використовує багато деконволюційних шарів для зіставлення вхідного шуму з бажаним вихідним зображенням. Пакедна нормалізація використовується для стабілізації навчання мережі. Активація ReLU використовується в генераторі для всіх шарів, крім вихідного шару, який використовує шар tanh, а Leaky ReLU використовується для всіх шарів у дискримінаторі. Ця мережа навчалася за допомогою міні-пакедного стохастичного градієнтного спуску, а для прискорення навчання використовувався оптимізатор Адама з налаштованими гіперпараметрами. Результати роботи виявилися досить цікавими. Автори показали, що генератори мають цікаві властивості векторної арифметики, за допомогою яких ми можемо маніпулювати зображеннями так, як нам потрібно (рис. 2.4, рис. 2.5).

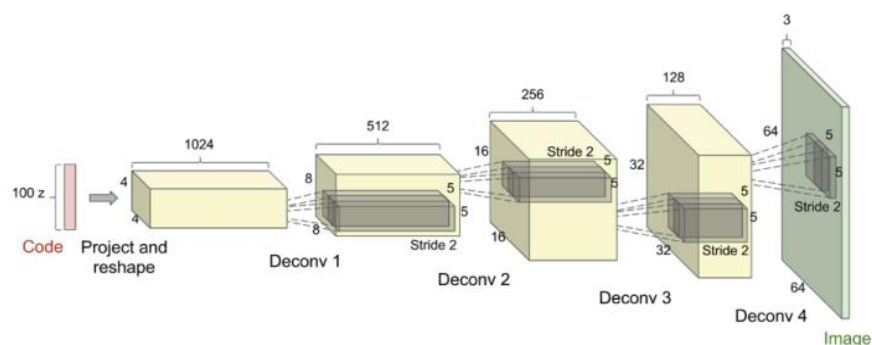


Рисунок 2.4 – Генератор DCGAN

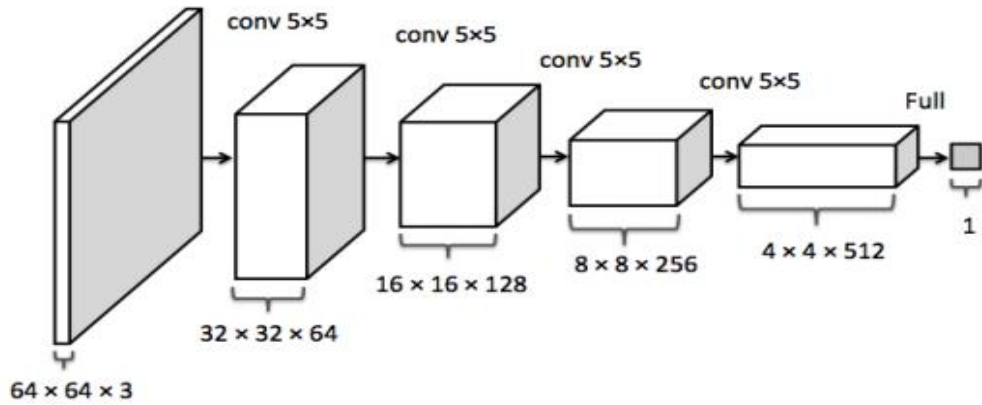


Рисунок 2.5 – Дискримінатор DCGAN

Однією з найпоширеніших варіацій GAN є умовна GAN, яка будується простим додаванням умовного вектора до вектора шуму. До cGAN ми генерували зображення випадковим чином з випадкових вибірок шуму z . Що, якщо ми хочемо згенерувати зображення з деякими бажаними характеристиками? Чи можна якось надати моделі додаткову інформацію про те, який тип зображення ми хочемо згенерувати?

Відповідь – так, і Умовний GAN – це спосіб зробити це. Обумовивши модель додатковою інформацією, яка надається як генератору, так і дискримінатору, можна керувати процесом генерації даних (рис. 2.6).

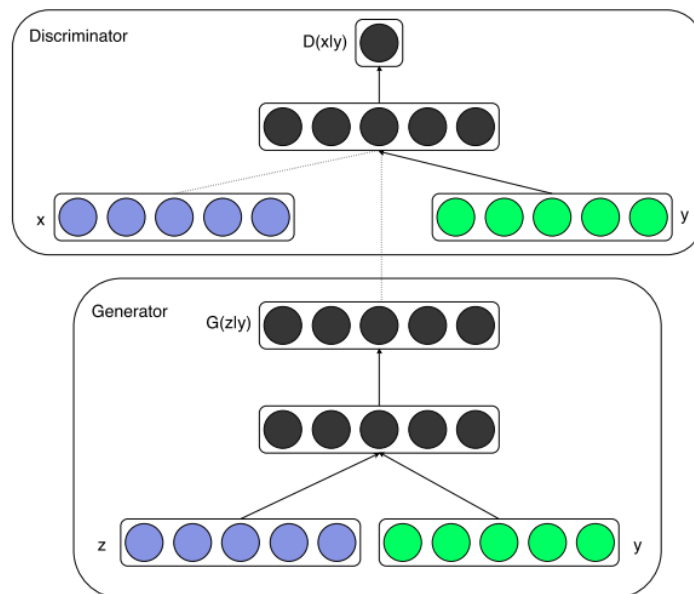


Рисунок 2.6 – Базовий приклад cGAN з вектором кондиціонування y

Однією з переваг GAN є те, що їх можна навчати навіть на невеликих навчальних даних. Дійсно, результати GAN є багатообіцяючими, але процедура навчання не є тривіальною, особливо налаштування гіперпараметрів мережі. Крім того, GAN важко оптимізувати, оскільки вони нелегко сходяться. Звичайно, існують певні поради та підказки для злому GAN-мереж, але вони не завжди можуть допомогти. Ви можете знайти деякі з них тут. Крім того, у нас немає жодних критеріїв для кількісної оцінки результатів, окрім перевірки того, чи є згенеровані зображення реалістичними на вигляд чи ні.

Моделі глибинного навчання дійсно досягають людського рівня продуктивності в керованому навчанні, але цього не можна сказати про неконтрольоване навчання. Тим не менш, вчені в галузі глибокого навчання наполегливо працюють над покращенням продуктивності неконтрольованих моделей. У цій статті ми розглянули, як насправді працюють два найвідоміші фреймворки для неконтрольованого навчання генеративних моделей. Ми дізналися про проблеми варіаційних автокодерів і про те, чому змагальні мережі краще створюють реалістичні зображення. Але є проблеми з GAN, такі як стабілізація їх навчання, що все ще є активною областю досліджень. Проте GAN дійсно потужні, і наразі їх використовують у різноманітних завданнях, таких як створення високоякісних зображень (див. це відео) та відео, переклад тексту в зображення, покращення зображень, реконструкція 3D-моделей об'єктів із зображень, генерація музики, пошук ліків від раку тощо. Крім того, багато дослідників глибинного навчання також працюють над об'єднанням цих двох моделей і над тим, щоб отримати найкраще з обох цих моделей. Бачачи зростаючі темпи розвитку глибокого навчання, я вірю, що GAN відкриють багато закритих дверей штучного інтелекту, таких як напівкероване навчання та навчання з підкріпленням. У найближчі кілька років генеративні моделі будуть дуже корисними для графічного дизайну, створення привабливих користувацьких інтерфейсів тощо. За допомогою

генеративних змагальних мереж також можна буде генерувати тексти на природній мові.

2.2 Дифузійна модель

Дифузійна модель – модель, яка використовується для навчання дифузійних моделей, слідує схемі, подібній до мереж VAE, але загалом є простішою та зрозумілішою, ніж інші мережеві архітектури. Вхідний розмір вхідного шару такий самий, як розмірність даних. Залежно від того, якої глибини потрібна мережа, може бути кілька прихованих шарів. Середній шар є лінійним шаром з відповідною функцією активації. Розмір остаточного шару знову такий самий, як вихідний вхідний шар, тому вихідні дані можна реконструювати. У Denoising Diffusion Networks останній рівень складається з двох незалежних виходів, відповідно призначених для прогнозування середнього значення та дисперсії щільності ймовірності (рис. 2.7).

Дифузійні моделі є генеративними моделями, тобто вони використовуються для створення даних, подібних до даних, на яких вони навчалися. По суті, дифузійні моделі спотворюють навчальні дані, постійно додаючи гаусівський шум, а потім вчать відновлювати дані, змінюючи цей процес додавання шуму на зворотний. Після навчання ми можемо генерувати дані, пропускаючи випадковий шум через навчений процес усунення шумів.

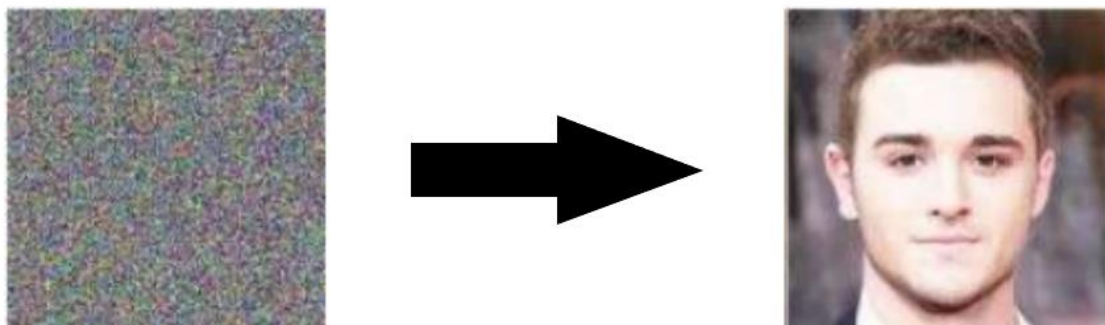


Рисунок 2.7 – Усунення шумів

Дифузійні моделі можна використовувати для створення зображень із шуму GAN і VAE є двома важливими генеративними моделями, які досягли великого успіху та визнання в багатьох застосуваннях.

GAN працюють добре, але їхнім результатам бракує різноманітності, і їх важко навчити через різноманітні проблеми, такі як згортання режиму та зникнення градієнтів тощо. VAE мають найсильнішу теоретичну основу, але моделювання хороших функцій втрат у VAE є проблемою, що робить їх вихід неоптимальним.

Ключова концепція дифузійного моделювання полягає в тому, що якщо ми можемо побудувати модель навчання, яка може навчитися систематичному розпаду інформації через шум, тоді ми зможемо повернути цей процес назад і відновити інформацію з шуму. Концепція подібна до VAE, яка намагається оптимізувати цільову функцію, спочатку проектуючи дані в латентний простір, а потім відновлюючи їх до початкового стану. Однак метою системи є не вивчення розподілу даних, а моделювання серії розподілів шуму в ланцюжку Маркова та декодування даних ієрархічним способом, щоб скасувати/усунути шум у даних.

Модель дифузії можна розглядати як модель латентної змінної. латентна змінна означає, що ми посилаємося на латентний безперервний простір ознак. Таким чином, вони можуть виглядати схожими на варіаційні автокодери (VAE) .

Існує кілька генеративних моделей на основі дифузії, які пропонують подібні ідеї, включаючи ймовірнісні моделі дифузії (Sohl-Dickstein et al., 2015), мережу балів з умовою шуму (NCSN; Yang & Ermon, 2019) і ймовірнісні моделі дифузії з усуненням шуму (DDPM; Хо та ін., 2020).

На практиці модель дифузії використовує для моделювання T-кроковий ланцюг Маркова. Тут ланцюг Маркова означає, що кожен крок залежить лише від попереднього кроку, що є м'яким припущенням. Важливо, що на відміну від моделей на основі потоку, ми не обмежені використанням певного типу нейронної мережі.

Якщо говорити точніше, модель дифузії — це модель латентної змінної, яка використовує фіксований ланцюг Маркова для відображення даних у латентному просторі. Цей ланцюжок поступово додає шум до даних, щоб отримати приблизний апостеріорний розподіл $q(x_{1:T}|x_0)$, де x_1, \dots, x_T є латентними змінними з однаковими розмірами. Нижче ми бачимо поведінку такого ланцюга Маркова для даних зображення (рис. 2.8).

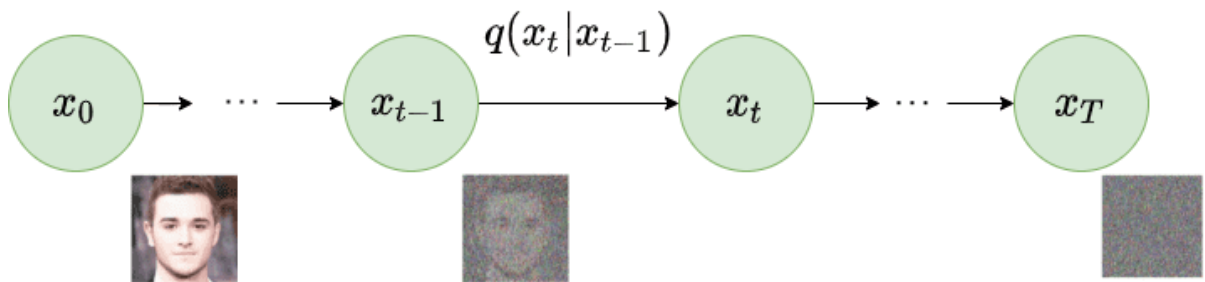


Рисунок 2.8 – Ланцюг Маркова

Ми можемо формально визначити процес прямої дифузії як ланцюг Маркова, тому, на відміну від кодера у VAE, він не вимагає навчання. Починаючи з початкової точки даних x_0 , ми додаємо гаусівський шум з дисперсією β_t до x_{t-1} на наступних T кроках і отримуємо набір вибірок шуму x_t із розподілом $q(x_t | x_{t-1})$.

Коли розмір кроку стає більшим, вибірка даних x_0 поступово втрачає свої характерні характеристики. Нарешті, коли $T \rightarrow \infty$, x_t еквівалентний ізотропному розподілу Гауса (рис. 2.9).

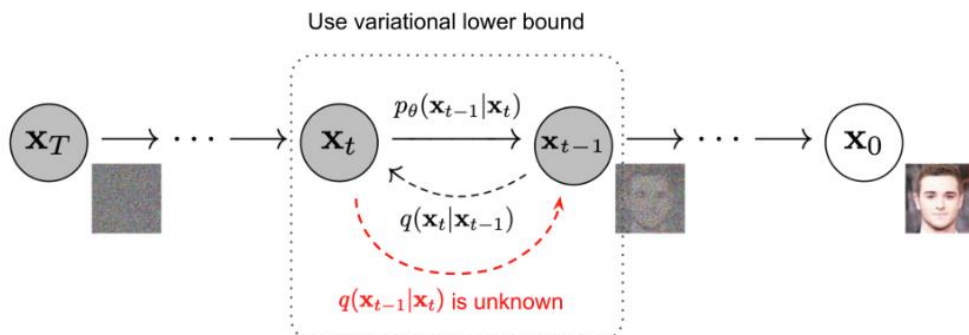


Рисунок 2.9 – Генерація зразка шляхом додавання шуму

Ланцюг Маркова процесу прямої (зворотної) дифузії генерації зразка шляхом повільного додавання (видалення) шуму.

Таким чином, ми можемо перейти від вхідних даних x_0 до закритої форми x_T у зручний спосіб. Математично це апостеріорна ймовірність.

Тут середнє значення та дисперсія функції щільності ймовірності залежать від параметра βt , який є гіперпараметром, значення якого може бути постійним протягом усього процесу або поступово змінюватися на послідовних кроках. Для призначення значення диференціального параметра можна використовувати ряд функцій для моделювання поведінки (наприклад, сигмоїда, \tanh , лінійна тощо).

Щоб генерувати зразки на кроці часу t , ми можемо використати інше поняття з термодинаміки – «динаміка Ланжевена». Відповідно до стохастичної градієнтної динаміки Ланжевена [2], ми можемо лише відбирати нові стани системи через градієнт функції щільності в оновленні ланцюга Маркова.

2.3 Реферування

Реферування тексту широко використовується багатьма веб-сайтами та додатками для створення стрічки новин та анотацій до статей.

Це стало дуже важливим для нас через наш щільний графік. Ми віддаємо перевагу коротким резюме з усіма важливими моментами, ніж читати цілий звіт і підсумовувати його самостійно. Отже, було зроблено кілька спроб автоматизувати процес створення резюме.

Реферування – це метод скорочення довгих текстів таким чином, щоб резюме містило всі важливі моменти оригінального документа.

2.4 Екстрактивні методи реферування

Узагальнення на основі екстракції: екстрактивний підхід передбачає виокремлення найважливіших фраз і рядків з документів. Потім всі важливі рядки об'єднуються для створення резюме. Таким чином, у цьому випадку кожен рядок і кожне слово в резюме фактично належить оригінальному документу, який узагальнюється.

Абстрактне узагальнення: Абстрактний підхід передбачає узагальнення на основі глибокого навчання.

Отже, він використовує нові фрази і терміни, відмінні від фактичного документа, зберігаючи при цьому суть, так само, як ми фактично підсумовуємо. Отже, це набагато складніше, ніж екстрактивний підхід.

Було помічено, що екстрактивні резюме іноді працюють краще, ніж абстрактні, ймовірно, тому, що екстрактивні резюме не потребують генерації природної мови та семантичного представлення.

Людське оцінювання: бали виставляються експертами-людьми на основі того, наскільки добре резюме охоплює пункти, відповідає на запитання та інші фактори, такі як граматична правильність і відсутність надмірностей.

Автоматична оцінка ROUGE: ROUGE розшифровується як Recall-Oriented Understudy for Gisting Evaluation. Це метод, який визначає якість резюме, порівнюючи його з іншими резюме, створеними людьми як еталон. Для оцінки моделі використовується ряд посилань, створених людьми, і резюме кандидата, згенероване машиною. Інтуїція полягає в тому, що якщо модель створює гарне резюме, то воно повинно мати спільні частини, що перетинаються з людськими посиланнями. Ця ідея була запропонована Чин-Ю Лін (Chin-Yew Lin) з Каліфорнійського університету.

Поширеними версіями ROUGE є ROUGE-n: Вимірюється на основі порівняння між машинним виходом і еталонним виходом на основі n-грам. n-грама – це безперервна послідовність з n елементів із заданого зразка тексту

або мови, тобто це просто послідовність слів. Біграми означають два слова, триграми – 3 слова і так далі. Зазвичай ми використовуємо біграми.

Де для кожної біграми "i" ми обчислюємо найменшу з кількості разів, коли вона зустрічається у створеному документі X та довідковому документі S, для всіх довідкових документів, поділену на загальну кількість разів, коли кожна біграма з'являється у всіх довідкових документах.

Це базується на оцінках BLEU.

Екстрактивне узагальнення: "методи екстрактивного узагальнення створюють резюме, вибираючи підмножину речень у вихідному тексті».

Екстрактивні референтні редактори спочатку створюють проміжне представлення, основним завданням якого є виділення або вилучення найважливішої інформації з тексту, що підлягає узагальненню на основі представлень.

Тепер, коли ми отримали проміжні репрезентації, ми переходимо до присвоєння балів кожному реченню, щоб визначити їхню важливість. Для тематичних репрезентацій оцінка речення залежить від тематичних слів, які воно містить, а для індикаторних репрезентацій оцінка залежить від особливостей речень. Нарешті, речення з найвищими балами вибираються і використовуються для створення резюме.

2.4.1 Методи на основі графів

Графові методи були вперше представлені в статті Ради Міхальчі та Пола Тарау з Університету Північного Техасу. Метод називається алгоритм Text Rank і на нього вплинув алгоритм Page Rank від Google. Цей алгоритм в першу чергу намагається знайти важливість вершини в заданому графі.

Ми дізналися, що в цьому алгоритмі кожне речення представляється у вигляді вершини. Ребро, що з'єднує дві вершини або два речення, означає, що ці два речення схожі. Якщо схожість двох речень перевищує певний поріг, то вершини, що представляють речення, з'єднуються ребром.

Коли дві вершини з'єднуються, це означає, що одна вершина віддає голос іншій. Чим більша кількість голосів за певний вузол (вершину або речення), тим важливішим є цей вузол і, очевидно, речення, яке він представляє. Тепер, голоси також мають певну вагу, кожен голос не має однакової ваги або важливості. Важливість голосу також залежить від важливості вузла або речення, за яке віддано голос, чим вища важливість вузла, за який віддано голос, тим вища важливість голосу. Таким чином, кількість голосів, відданих за речення, і важливість цих голосів визначають важливість речення. Ця ж ідея лежить в основі алгоритму ранжування сторінок Google, і того, як він вирішує і ранжує веб-сторінки, за винятком того, що вузли представляють веб-сторінки.

Якщо у нас є абзац, ми розкладемо його на набір речень. Тепер, скажімо, ми представимо кожне речення у вигляді вершини 'vi', таким чином, ми отримаємо множину вершин V. Як ми вже обговорювали, ребро з'єднує вершину з іншою вершиною тієї ж множини, тому ребро E можна представити як підмножину (V x V). У випадку орієнтованого графа, скажімо, $In(V\{i\})$ – це кількість вхідних ребер у вершину, а $Out(v\{j\})$ – кількість вихідних ребер з даної вершини, а оцінка важливості вершини задається через $S\{j\}$.

2.4.2 Алгоритм ранжирування сторінок

Згідно з алгоритмом ранжування сторінок Google, де $S(V\{i\})$ – це оцінка розглянутої вершини, а $S(V\{j\})$ представляє всі вершини, які мають вихідні ребра до $V\{i\}$. Тепер оцінка $V\{j\}$ ділиться на вихідний ступінь $V\{j\}$, який враховує ймовірність того, що користувач вибере саме цю веб-сторінку.

Давайте спочатку перевіримо алгоритм ранжування сторінок, а потім перетворимо його для ранжування тексту. Як ми бачимо вище, є 4 вершини, спочатку ми присвоюємо випадкові оцінки всім вершинам, скажімо, $[0.8, 0.9, 0.9, 0.9]$. Потім ребрам присвоюються ймовірнісні оцінки (рис. 2.10).

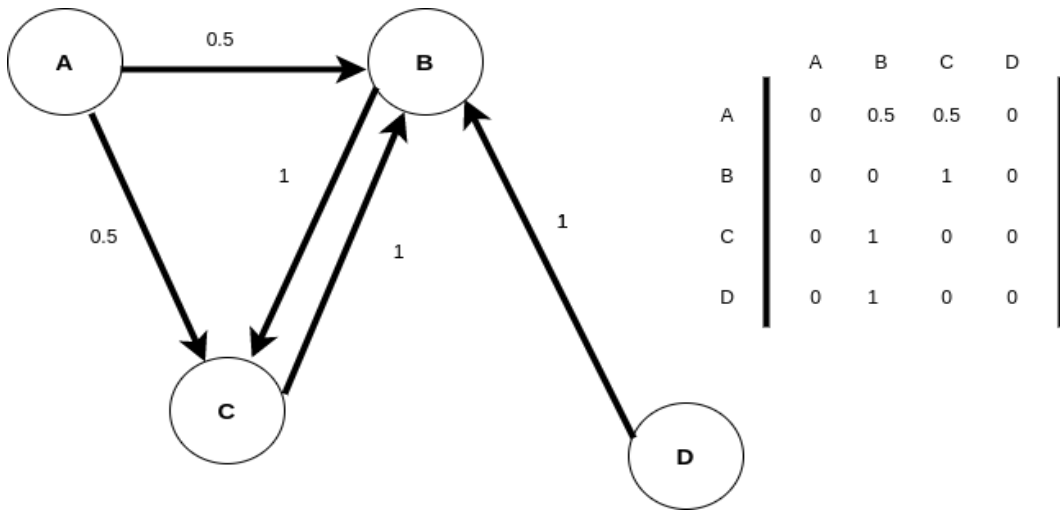


Рисунок 2.10 – Алгоритм ранжування

Матриця є матрицею суміжності графа. Можна помітити, що значення суміжних матриць є значеннями ймовірності, тобто $1/\text{ступінь}$ цієї вершини або вузла. Таким чином, фактично граф ранжування сторінок стає незваженим, оскільки рівняння містить лише доданок, який задає вагу (рис. 2.11, рис. 2.12).

$$\begin{array}{c}
 \begin{array}{c|cccc}
 & A & B & C & D \\
 \hline
 A & 0 & 1 & 1 & 0 \\
 B & 0 & 0 & 1 & 0 \\
 C & 0 & 1 & 0 & 0 \\
 D & 0 & 1 & 0 & 0 \\
 \hline
 \end{array}
 & \times & \frac{1}{(\text{Out}(V\{i\}))} = &
 \begin{array}{c|cccc}
 & A & B & C & D \\
 \hline
 A & 0 & 0.5 & 0.5 & 0 \\
 B & 0 & 0 & 1 & 0 \\
 C & 0 & 1 & 0 & 0 \\
 D & 0 & 1 & 0 & 0 \\
 \hline
 \end{array}
 \end{array}$$

Рисунок 2.11 – Матриця суміжності

$$\begin{array}{c}
 \begin{array}{c|c}
 S(a) \\
 S(b) \\
 S(c) \\
 S(d) \\
 \hline
 \end{array}
 = (1 - 0.85) + 0.85 \left(\begin{array}{c|cccc}
 & A & B & C & D \\
 \hline
 A & 0 & 0.5 & 0.5 & 0 \\
 B & 0 & 0 & 1 & 0 \\
 C & 0 & 1 & 0 & 0 \\
 D & 0 & 1 & 0 & 0 \\
 \hline
 \end{array} \cdot \begin{array}{c|c}
 \begin{array}{c} 0.8 \\ 0.9 \\ 0.9 \\ 0.9 \end{array} \\
 \hline
 \end{array} \right)
 \end{array}$$

Рисунок 2.12 – Алгоритм ранжування

Ми бачимо, що стара матриця оцінок перемножується з використанням матриці суміжності для отримання нової матриці оцінок. Ми будемо продовжувати це робити до тих пір, поки норма L2 нової матриці оцінок і старої матриці оцінок не стане меншою за задану константу, найчастіше 1×10^{-8} . Це властивість збіжності, заснована на лінійній алгебрі та теорії власних значень і векторів.

Для спрощення ми опустимо математику. Після досягнення збіжності ми отримуємо остаточні оцінки важливості з матриці оцінок.

2.4.3 Завдання вилучення ключових слів

Раніше для цього використовувався частотний фактор, який давав порівняно низькі результати. У статті про ранжування тексту представлено повністю неконтрольований алгоритм. Згідно з алгоритмом, текст природною мовою токенізується, частини мови позначаються тегами, а окремі слова додаються до графа слів у вигляді вузлів. Тепер, якщо два слова схожі, відповідні вершини з'єднуються за допомогою ребра. Подібність вимірюється за допомогою входжень слів. Якщо два слова зустрічаються у вікні з N слів, де N змінюється від 2 до 10, то ці два слова вважаються схожими. Слова з максимальною кількістю важливих ребер інцидентності обираються як найбільш важливі ключові слова (рис. 2.13).

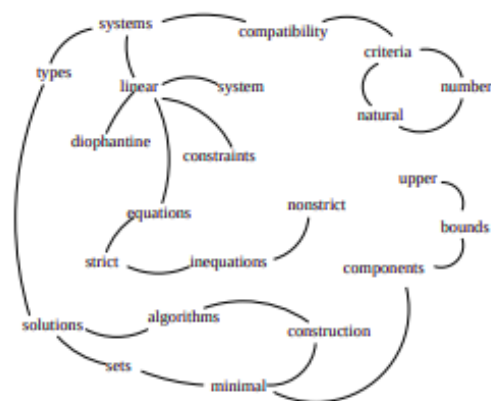


Рисунок 2.13 – Алгоритм ранжування

2.4.4 Завдання на виділення речень

Він також працює подібно до вилучення ключових слів, з тією лише різницею, що при вилученні ключових слів вершини представляли ключові слова, а тут вони представляють цілі речення. Тепер, для формування графа для ранжування речень, алгоритм створює вершину для кожного речення в тексті і додає до графа. Речення є занадто великими, тому не можна застосувати міру повторюваності. Тому в роботі використовується "схожість" між двома реченнями за допомогою перекриття змісту між двома реченнями, простіше кажучи, схожість залежить від кількості спільних лексем, присутніх у двох реченнях. Автори пропонують дуже цікавий "рекомендаційний" інсайт. Вони позначають з'єднання ребра між двома схожими реченнями або вершинами так, ніби рекомендують читачеві прочитати ще один рядок, схожий на поточний рядок, який він читає. Таким чином, схожість, яку я відчуваю, означає схожий зміст або інтерес між двома реченнями. Щоб запобігти рекомендаціям довгих речень, важливість множиться на нормалізуючий коефіцієнт (рис. 2.14).

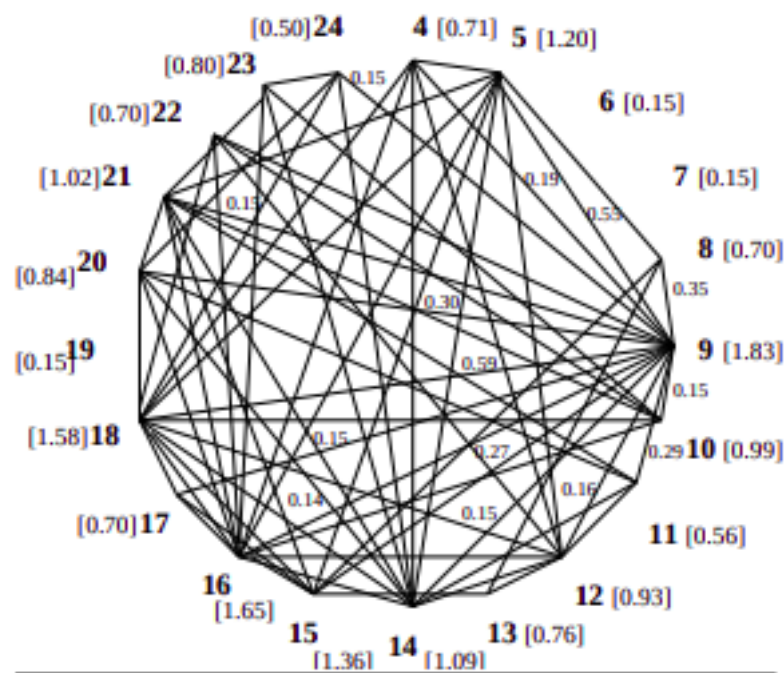


Рисунок 2.14 – Таблиця схожості

Найважливіші речення отримуються так само, як і для вилучення ключових слів.

Це загальне уявлення про те, як працює текстове ранжування, будь ласка, прочитайте оригінальну статтю, щоб дізнатися більше.

На практиці, для вилучення резюме ми використовуємо косинусну подібність, щоб визначити подібність між двома реченнями. Використовуючи цей метод, ми можемо отримати кілька пов'язаних підграфів, які позначають кількість важливих тем у всьому документі. Зв'язані компоненти підграфів дають речення, важливі для відповідних тем.

Бібліотека "Pytextrank" дозволяє застосовувати алгоритм ранжування тексту безпосередньо на python.

2.5 Абстрактні методи реферування

Абстрактні референтні програми називаються так тому, що вони не вибирають речення з вихідного тексту для створення реферату. Замість цього вони перефразовують основний зміст тексту, використовуючи словниковий запас, відмінний від вихідного документа. Це дуже схоже на те, як ми, люди, робимо резюме. Ми створюємо семантичну репрезентацію документа в нашому мозку. Потім ми підбираємо слова з нашого загального словника (слова, які ми зазвичай використовуємо), які підходять до цієї семантики, щоб створити коротке резюме, яке представляє всі пункти фактичного документа. Як ви можете помітити, розробка такого роду резюме може бути складною, оскільки для цього знадобиться генерація природної мови. Давайте розглянемо найпоширеніший підхід до цієї проблеми.

2.5.1 Застосування послідовно-послідовних ШНМ

Підхід був запропонований у статті Рамеша Наллапаті, Боуена Чжоу, Цицерона дос Сантоса, Каглара Гюльчехре, Бінга Сяна з ІВМ. Термін "моделі

перетворення послідовності в послідовність" використовується тому, що моделі призначені для створення вихідної послідовності слів з вхідної послідовності слів. Вхідною послідовністю в даному випадку є власне текстовий документ, а вихідною послідовністю - скорочене резюме.

У статті пропонується модель, натхненна моделлю кодера-декодера рекурентної нейронної мережі, яка вперше була запропонована для машинного перекладу Дмитром Богдановим з Університету Якова, Німеччина.

Однак, як ви вже зрозуміли, проблеми значно відрізняються. По-перше, для машинного перекладу нам потрібен переклад без втрат, оскільки ми хочемо отримати точне речення в перекладеному вигляді, але для створення резюме нам потрібно стиснути оригінальний документ, щоб створити резюме, тому воно повинно бути з певними втратами. По-друге, для генерації резюме довжина резюме не залежить від оригінального тексту. Ці два моменти є ключовими викликами в задачі, як зазначено в умові задачі.

Перш ніж перейти до деталей застосування, давайте розглянемо мережі кодерів і декодерів, а також причину використання шару уваги.

2.5.2 Мережі кодерів і декодерів

Якщо ми розглянемо загальний LSTM-шар (довготривала коротка пам'ять), то він виглядає приблизно так, як показано на схемі нижче. Він або видає вихід на кожен вхід, або створює вектор ознак, який потім використовується шарами щільної нейронної мережі для класифікації із застосуванням шарів softmax. Наприклад, визначення настрою, де ми пропускаємо все речення через RNN і використовуємо вектори ознак, які підходять для шарів softmax, для отримання кінцевого результату.

Екстрактивні референтні редактори спочатку створюють проміжне представлення, основним завданням якого є виділення або вилучення найважливішої інформації з тексту, що підлягає узагальненню на основі представлень (рис. 2.15).

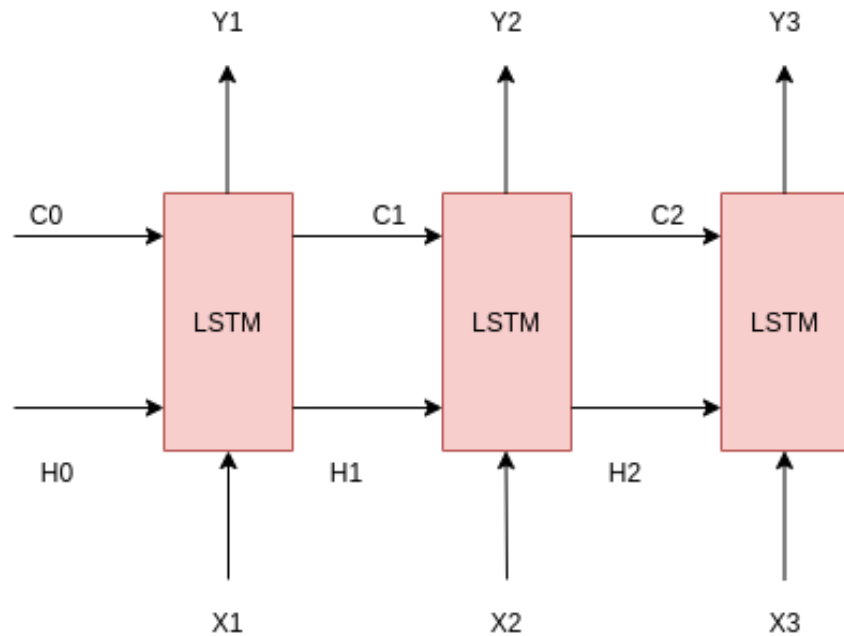


Рисунок 2.15 – LSTM шар

Але тут слід розуміти, що для поточної проблеми або проблем, подібних до цієї, включно з машинним перекладом, якщо говорити ширше, коли ми можемо сказати, що проблема полягає в перекладі з послідовності в послідовність, цей конкретний підхід до моделі не може бути застосований. Основна причина полягає в тому, що розмір вихідних даних не залежить від розміру вхідних даних, і обидва вони є послідовностями. Для вирішення цієї проблеми була введена мережева модель кодера-декодера (рис. 2.16).

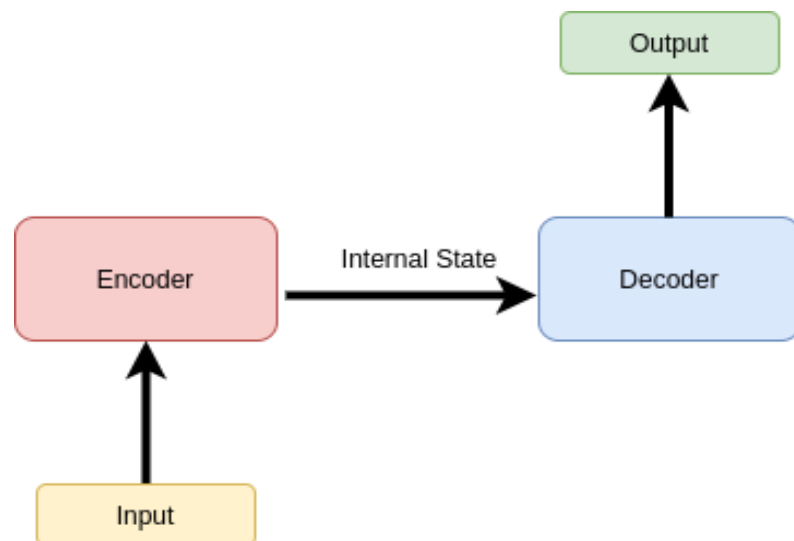


Рисунок 2.16 – Кодер-декодер

Кодер відповідає за отримання вхідного речення або оригінального документа і генерує кінцевий вектор стану (прихований стан і стан комірки). Це представлено внутрішнім станом на діаграмі. Кодер може містити шари LSTM, RNN або GRU. Здебільшого використовуються шари LSTM через відсутність проблеми вибуху та зникаючого градієнта (рис. 2.17).

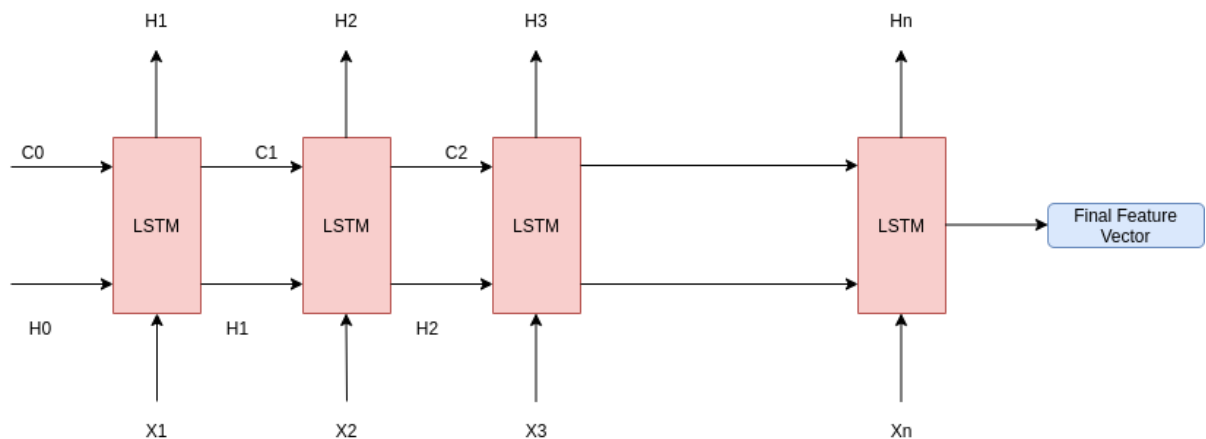


Рисунок 2.17 – Кодерна мережа

На наведеній вище схемі показано нашу кодерну мережу. У кодерній мережі одне слово подається на один крок, і, нарешті, після того, як n -те вхідне слово подається на LSTM-шар, прихований стан і стани комірок стають нашим кінцевим станом або вектором ознак. Стан комірки C_n і прихований стан H_n надсилаються до першого набору LSTM-шару декодера (рис. 2.18).

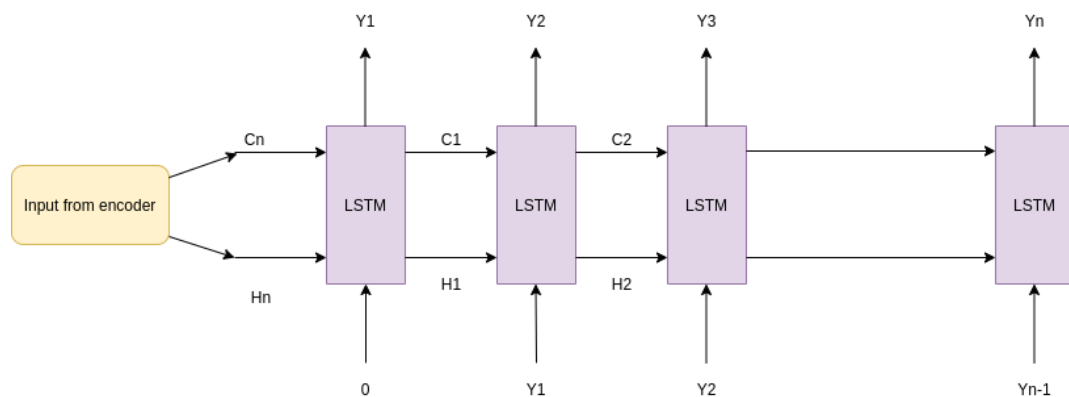


Рисунок 2.18 – Модель декодера

Так виглядає наша модель декодера. Тепер перший рівень отримує вхідні дані з кінцевих станів кодера, які є активаціями прихованих станів та станів комірок. Модель декодера приймає вхідні дані і генерує передбачувані слова вихідної послідовності, враховуючи попереднє згенероване слово. Так, для LSTM на часовому кроці 1 на вхід декодера подається вектор 0, а Y_1 – згенероване передбачене слово, на часовому кроці 2 Y_1 подається на вхід шару LSTM, а Y_2 – згенероване слово і так далі. Декодер генерує слова з кроком у часі, поки не зустрінеться тег <end>.

Тут може виникнути питання, як же утворюються слова? Що ж, ось відповідь. Модель кодера-декодера навчається на цільовому наборі або словнику слів. Тепер, на кожному кроці декодера, прихована активація LSTMs передається через шар softmax, який генерує ймовірності того, що кожне слово в словнику буде передбачене як наступне слово. Слово з максимальною ймовірністю обирається як вихідне на цьому кроці. Як же модель дізнається, яке саме слово відповідає семантиці? Для цього модель навчається на наборі даних і перетворює задачу на задачу керованої класифікації. Крім того, моделі, як правило, використовують вбудовування слів у словник з відомих векторів вбудовування, таких як word2vec від Google або Glove від Stanford NLP. Вбудовування слів допомагають отримати деякі уявлення про слово, наприклад, чи є дане слово схожим на задане слово чи ні. Іноді векторизації TFIDF також використовуються для генерації значущості контекстних слів.

Розглянемо на прикладі. Скажімо, у нас є набір даних, який містить довгі звіти та їхні людські резюме. Цю інформацію можна використовувати як мітки та цілі для навчання наших мереж кодерів-декодерів. Ми векторизуємо наші мітки та цілі, сформуємо словник. Далі ми підберемо вставки з word2vec або Glove для слів з нашого словника, а потім підставимо мітки і мішені до нашої моделі для навчання.

Але в цій конкретній проблемі узагальнення є проблема. Оригінальний документ може бути дуже великим. Скажімо, документ складається зі 100

рядків. Тепер, коли ми, як люди, підсумовуємо з 1-5 рядків, чи потрібно нам враховувати 100-й рядок? Чи не так? Коли ми підсумовуємо вручну, ми приділяємо більше уваги 1-5 рядкам, коли ми підсумовуємо ці рядки, і повільно рухаємося вперед.

Цей аспект не міг бути реалізований за допомогою звичайної моделі кодера-декодера, тому був введений механізм уваги.

2.5.3 Механізм уваги

Механізм має на меті зосередитися на деяких конкретних послідовностях з вхідних даних, а не на всій вхідній послідовності, щоб передбачити слово. Цей підхід дуже схожий на людський і, здається, вирішує проблему (рис. 2.19).

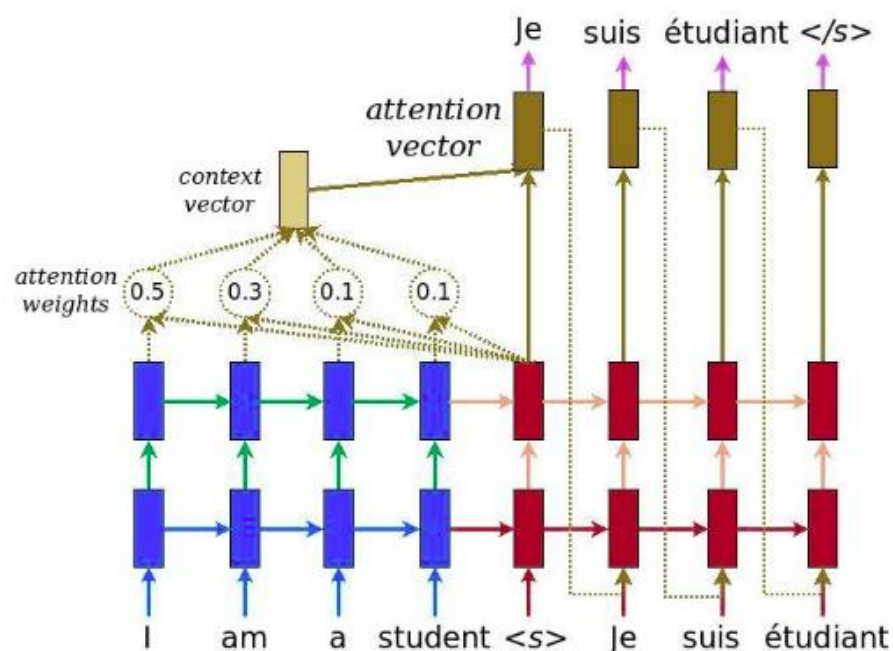


Рисунок 2.19 – Механізм уваги

На наведеній вище схемі показано реалізацію шару уваги у TensorFlow. Ми бачимо, що для передбачення слова кожному слову у вхідній послідовності присвоюється вага, яка називається вагою уваги. Сума

векторизованих ваг уваги використовується для формування векторів контексту, які використовуються для прогнозування.

Розглянемо детальніше. Пропонується кодер складати з двонаправленого ГПСШ, а декодер матиме однонаправлений ГПСШ. І кодер, і декодер матимуть однакову кількість прихованих шарів та блоків (рис. 2.20).

Давайте подивимося на механізм уваги тут.

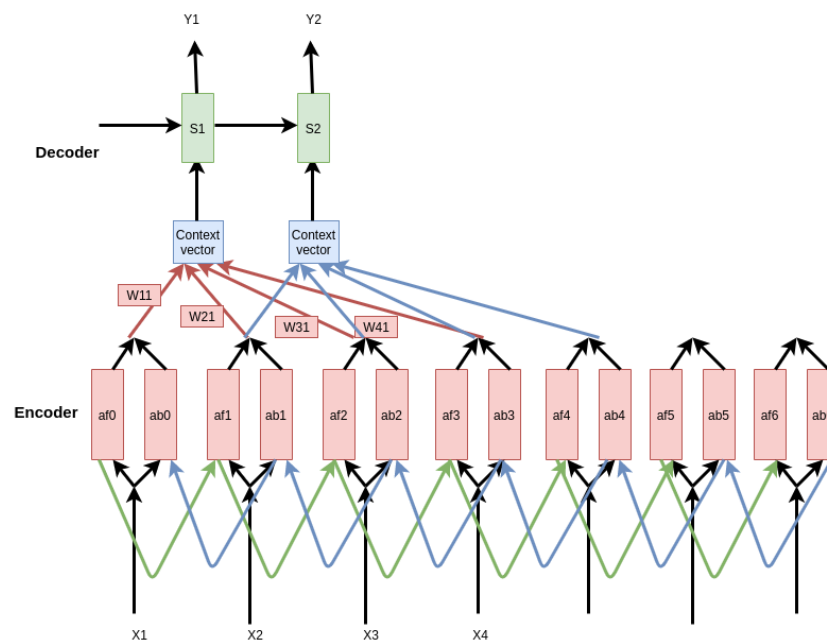


Рисунок 2.20 – Двонаправлений кодер

Верхній зелений шар показує декодер, Y_1 і Y_2 – виходи часового кроку. X_1 , X_2 – входи кодера. "af0" – це пряма активація входу 0, "ab0" – зворотна активація часового кроку 0 і так далі.

Скажімо, "an" – це комбінована активація ("abn", "afn") на часовому кроці n, отже, $a_0 = (af_0, ab_0)$, тобто комбінована як вперед, так і назад. " W_{mn} " визначає, яку вагу слід надавати входу на часовому кроці m при прогнозуванні виходу на часовому кроці n декодера.

Вектор контексту - це зважена сума ваг уваги. Він задається формулою:

$$C_{<n>} = \text{Sum}(W_{<m,n>} \cdot a_{<m>}) \text{ для всіх } m \text{ на вхідних часових кроках.}$$

Наведене вище рівняння стверджує, що вектор контексту для часового кроку n дорівнює сумі ваг, наданих входу на часовому кроці m та

комбінованій активації на часовому кроці t , для всіх t часових кроків у вхідних даних, тобто для x часових кроків, якщо у вхідних даних є x слів.

Де ми отримуємо e за допомогою нейронної мережі з одним прихованим станом. Мережа отримує вхідні дані $S(n-1)$ та $a(t)$ (комбінована активація часового кроку t) і видає $e(t,n)$ на виході. Нейронна мережа для отримання "e" також навчається під час навчання нашої моделі кодера-декодера. Ми використовуємо softmax, тому сума всіх ваг, присвоєних усім вхідним часовим інтервалам, завжди дорівнює 1.

Раніше ми обговорювали, що слова, передбачені декодером, генеруються за допомогою софтмаксу. Якщо ми використаємо всі слова з нашого словника як цільовий набір слів, то софтмакс матиме величезну кількість вихідних вершин, і передбачення буде неефективним з точки зору обчислень. Для вирішення цієї проблеми пропонується трюк великого словника, запропонований Себастьяном Жаном (Sebastien Jean) з Монреальського університету. Модель навчається в міні-партіях. Цільовий набір слів або словник-декодер кожної міні-пачки обмежується вихідним документом цієї міні-пачки. Якщо ми будемо використовувати різні підмножини словника-декодера для кожної міні-партії, існує ймовірність того, що цільовий словник стане нерівномірним за довжиною, що є дуже очевидним, оскільки різні вибірки мають різну кількість рядків і різну кількість слів. Тому ми додаємо найчастотніші слова в загальному словнику до словників підмножин, щоб зробити їх фіксованого розміру. Це зменшує витрати часу і прискорює збіжність, оскільки зі зменшенням розміру цільової множини шар softmax також скорочується.

2.5.4 Видобування ключових слів

Раніше, коли ми обговорювали модель кодера-декодера, я згадував, що ми зазвичай використовуємо вставки слів для представлення слів у документах після векторизації. Тепер давайте спробуємо подумати про те, що

нам насправді потрібно, щоб слова позначали для підбиття підсумків. Ми зрозуміємо, що вбудовувань недостатньо, оскільки для узагальнення нам потрібно зосередитися на контексті та ключових словах у фрагменті тексту. Вставки допомагають отримати загальне уявлення про слово, але не мають нічого спільного з контекстом тексту. Тому в статті пропонується враховувати такі фактори, як теги частин мови, теги іменованих сутностей та статистику TFIDF слова разом із вбудовуваннями для представлення слова. Ми перетворюємо неперервні значення TFIDF у категоріальні значення, використовуючи біни. Нарешті, ми беремо всі ознаки і вбудовування для слова і створюємо нове вбудовування для слів. Отже, теги TFIDF, POS дають нам уявлення про те, наскільки важливими є слова в контексті документа, а вбудовування слів дають загальне уявлення про слово.

Далі ми об'єднуємо їх в один довгий вектор і передаємо в мережу. Важливо зазначити, що ми використовуємо лише вставки слів для представлення слів на цільовій стороні (рис. 2.21).

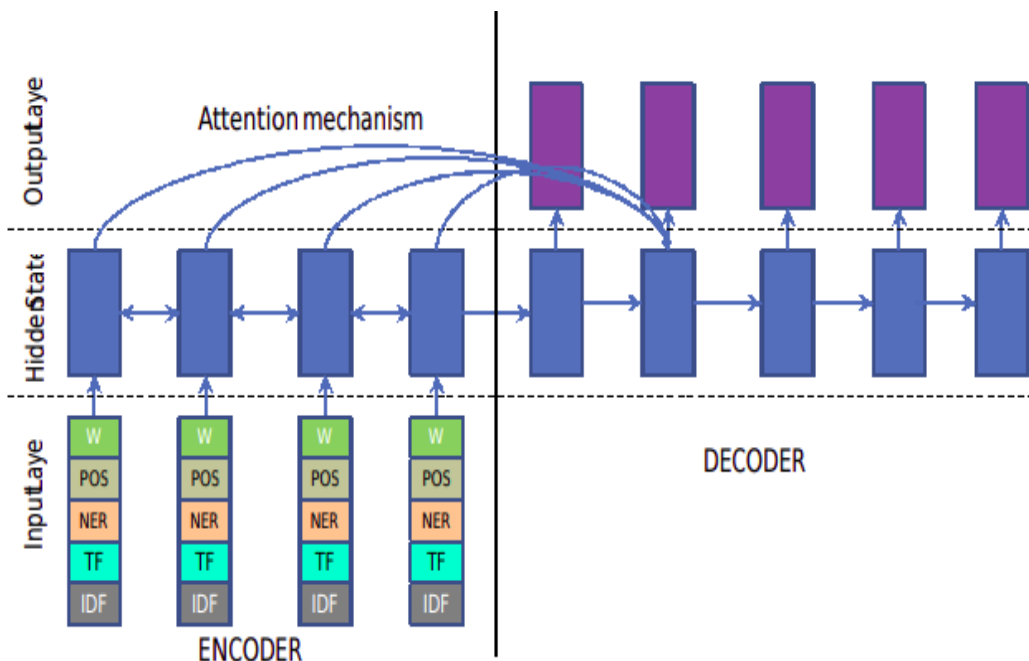


Рисунок 2.21 – Видобування ключових слів

Далі ми розглянемо ще один дуже важливий аспект, запропонований у документі. При обробці природної мови, коли ми навчаємо модель за допомогою керованої моделі, нам часто доводиться працювати з деякими словами, яких немає в нашому словнику. Такі слова називаються OOV або позасловниковими. У звичайних випадках ми обробляємо їх за допомогою тегів UNK. Але у випадку з анотаціями це буде неправильно, оскільки слова можуть мати певне значення в анотації. У статті пропонується вказівник декодера, що перемикається, для обробки таких випадків. На кожному кроці декодера зберігається вказівник на вхідний текст. Кожного разу, коли декодер стикається з терміном OOV, він вказує на термін у вхідному тексті і використовує термін з вхідного тексту безпосередньо. Таким чином, декодер може виконувати дві дії за один крок: згенерувати слово з цільового словника або вказати на нього і скопіювати. Це рішення приймається за допомогою перемикача. Якщо перемикач увімкнено, то декодер генерує слово, інакше він копіює слово з вхідного тексту (рис. 2.22).

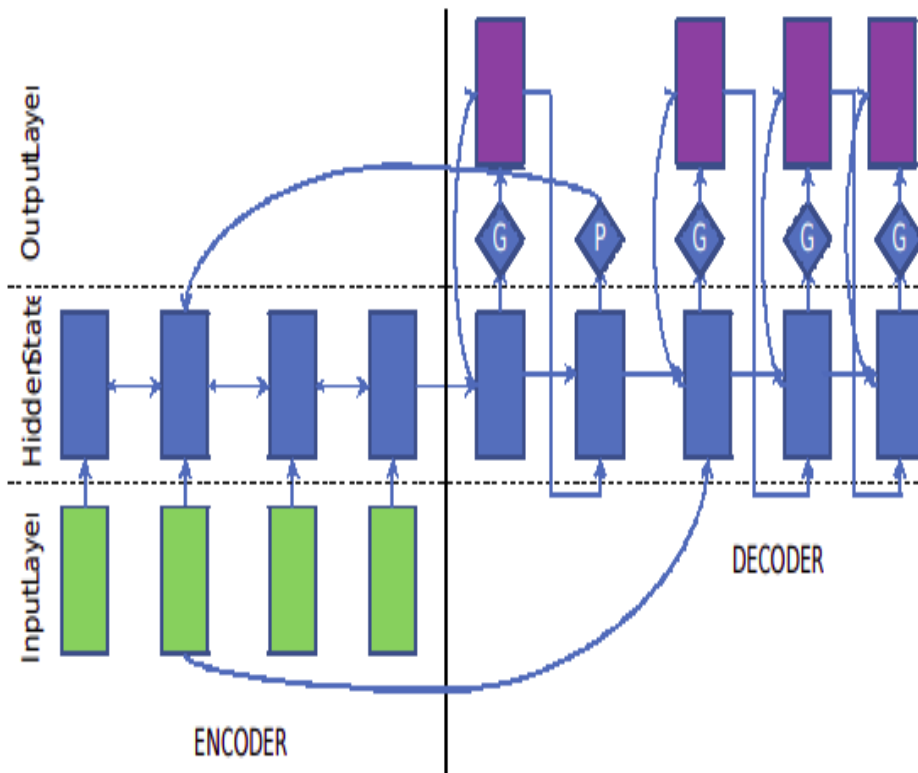


Рисунок 2.22 – Перемикаючий генератор

Виникає питання, як працює перемикач? Перемикач – це сигмоїдна функція активації на всьому векторі контексту на певному кроці декодера.

Вказівник на кожному часовому кроці повинен вказувати на слово для того, щоб скопіювати слово, це вирішується на основі розподілу ваг уваги для цих міток часу декодера. Це огляд моделі послідовного конспектування тексту. Я рекомендую вам переглянути посилання для отримання більш детальної інформації про реалізацію.

Варто зазначити, що автори запропонували ієрархічні шари уваги для довгих документів. Якщо документи дуже-дуже довгі, нам іноді може знадобитися ідентифікувати ключові речення з ключовими словами. Для цього нам потрібен ієрархічний механізм уваги. Один рівень відповідає за важливість речення, а інший – за важливість слів. Обидва рівні уваги працюють на двох рівнях одночасно (рис. 2.23, рис. 2.24).

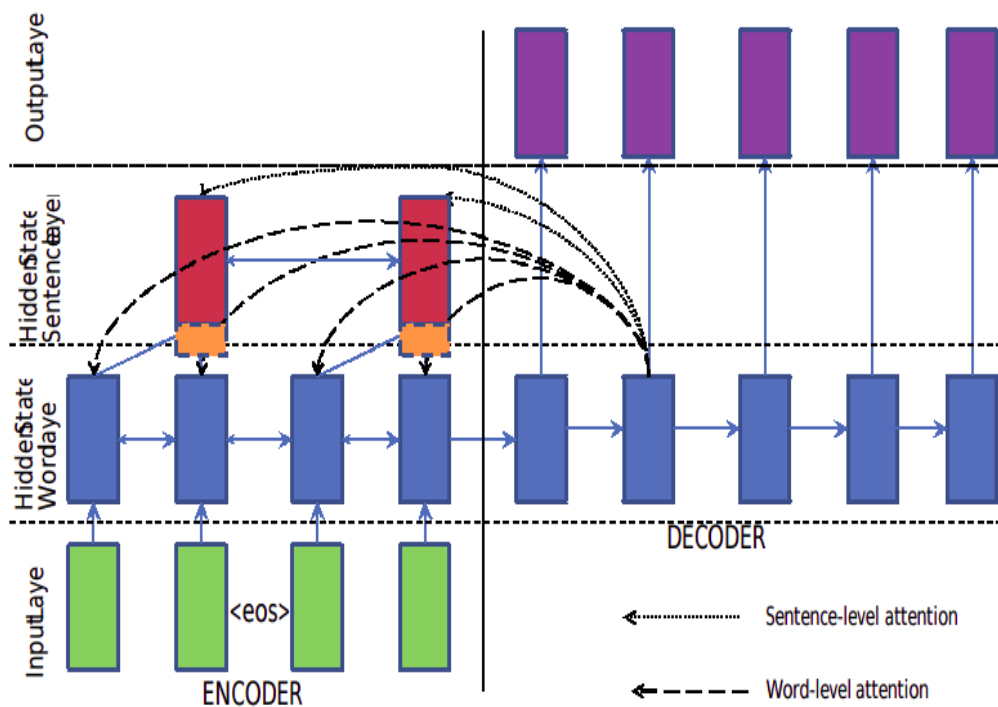


Рисунок 2.23 – Ієрархічні шари уваги

Інша подібна модель – модель Facebook.

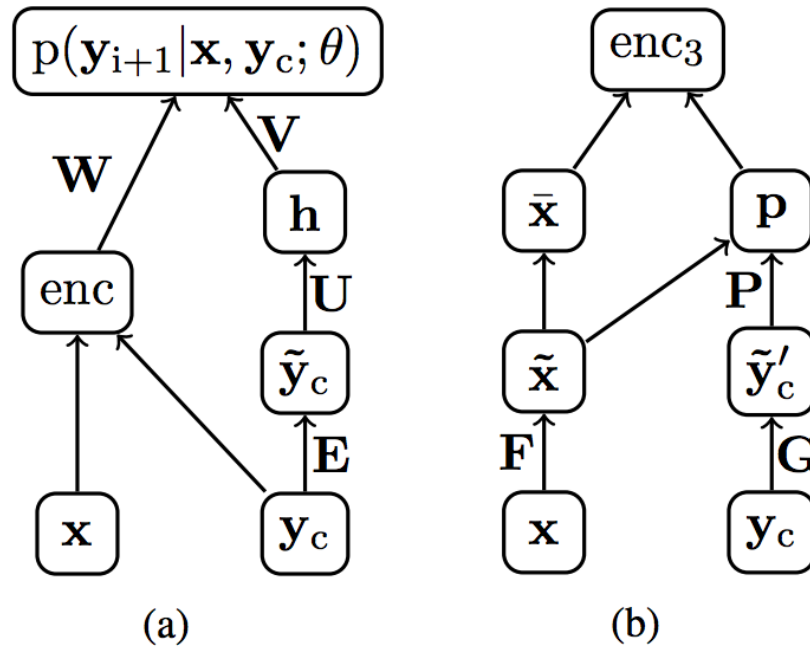


Рисунок 2.24 – Кодер з мішком слів

Кодер з мішком слів: він просто використовує мішок слів для представлення вхідного речення та ігнорує зв'язки з сусідніми словами. Декодер приймає закодований вектор або мішок слів і пророкує слово на часовому кроці.

Згортковий кодер: Згорткові шари використовуються для генерування векторів ознак зі слів-вкладень вхідних векторів, а потім декодери використовуються для створення слів на часовому кроці

Кодер на основі уваги: Цей кодер кодує слова на рівні уваги ШНМ, як ми обговорювали в попередньому підході.

Нарешті, до результатів застосовується променевий пошук, щоб отримати узагальнений текст.

3 АНАЛІЗ ІНСТРУМЕНТІВ ДЛЯ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

3.1 Вибір мови програмування

Python і Rust – дві популярні мови програмування, які можуть бути використані для аналізу даних та машинного навчання. Однак, кожен з них має свої плюси та мінуси, що може впливати на вибір.

У статті ми розглянемо порівняння Python і Rust, а також оцінимо, що простіше та швидше у використанні та яку мову краще вибрати для вашого проекту.

Python та Rust – дві різні мови програмування, які мають свої переваги та окремі недоліки. Python відомий своєю простотою та гнучкістю у використанні, що робить його популярним серед аналітиків даних та фахівців з машинного навчання.

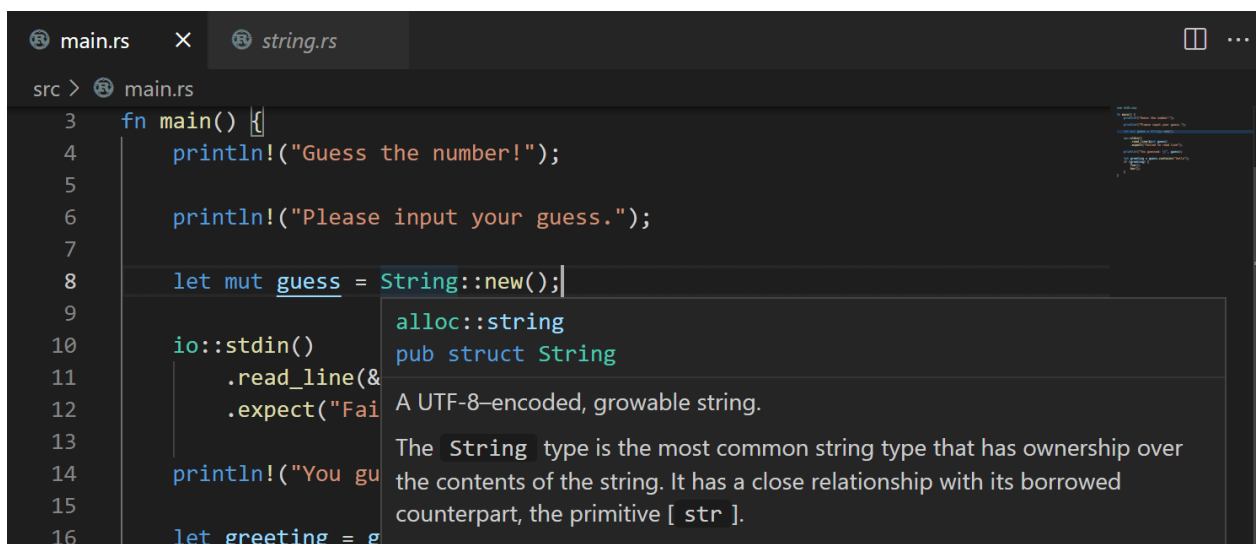
З іншого боку, Rust є мовою системного програмування з високою продуктивністю та безпекою пам'яті, що робить його популярною серед розробників програмного забезпечення та ігрових додатків. На відміну від Python, Rust надає можливості більш ефективної роботи з великим обсягом даних. Однак, Python є більш доступним для програмістів-початківців, оскільки він має велику кількість бібліотек і фреймворків для аналізу даних і машинного навчання. Тому вибір між Python та Rust залежить від цілей проекту та рівня досвіду програміста.

Python є більш простою в освоєнні мовою програмування, порівняно з Rust. Python має більш високий рівень абстракції та не вимагає від програміста знань низькорівневих деталей, таких як керування пам'яттю або типи даних. Python також має широку та різноманітну бібліотеку, яка значно спрощує роботу з даними та машинним навчанням.

З іншого боку, Rust, хоч і має суворіший синтаксис, забезпечує велику безпеку та продуктивність завдяки своїй системі типів та управлінню пам'яттю. Однак вивчення Rust може бути складним для новачків у

програмуванні. Таким чином, якщо ви новачок у програмуванні та хочете почати роботу з аналізом даних та машинним навчанням, то краще вибрати Python. Якщо вам потрібна максимальна продуктивність і безпека для ваших проектів, то варто розглянути Rust.

Python та Rust є двома з найпопулярніших мов програмування, що використовуються для аналізу даних та машинного навчання. Python має ширше застосування в цій галузі завдяки багатій екосистемі інструментів та бібліотек, таких як NumPy, Pandas, Matplotlib та Scikit-learn. Однак, Rust може надати переваги в швидкодії та безпеки коду завдяки своїй системі типів та керуванню пам'яттю. Він також може бути корисним для створення високопродуктивних компонентів для машинного навчання (рис. 3.1).



```

main.rs x string.rs
src > main.rs
3 fn main() {
4     println!("Guess the number!");
5
6     println!("Please input your guess.");
7
8     let mut guess = String::new();
9
10    io::stdin()
11        .read_line(&
12            .expect("Failed to read line.")
13        )
14        .unwrap();
15    println!("You guessed: {}", guess);
16    let greeting = g

```

alloc::string
pub struct String
A UTF-8–encoded, growable string.
The String type is the most common string type that has ownership over the contents of the string. It has a close relationship with its borrowed counterpart, the primitive [str].

Рисунок 3.1 – Мова програмування Rust

Недоліком Rust є його відносна новизна та невелика кількість бібліотек для роботи з даними. У той же час Python може мати проблеми з продуктивністю при роботі з великими обсягами даних. При виборі між Python та Rust для аналізу даних та машинного навчання слід враховувати вимоги до продуктивності та безпеки, а також доступність необхідних інструментів та бібліотек.

Коли справа доходить до швидкості, Rust зазвичай перевершує Python. Rust – це компіювана мова програмування з низькорівневою абстракцією пам'яті та багатопоточністю, що дозволяє обробляти більші обсяги даних швидше, ніж Python. Python, з іншого боку, є мовою, що інтерпретується, що означає, що вона може виконуватися повільніше, особливо при роботі з великими обсягами даних.

Також Python використовує збирач сміття, який може викликати затримки в роботі програми. Однак, Python має багаті бібліотеки для наукових обчислень та машинного навчання, які роблять його добрим вибором для швидкої розробки прототипів. При виборі між Rust та Python для проектів з аналізу даних та машинного навчання необхідно враховувати вимоги проекту до швидкості та доступності бібліотек.

Вибір між Python та Rust для аналізу даних та машинного навчання залежить від кількох факторів. Якщо у вас є великий обсяг даних, то Rust може бути кращим вибором завдяки своїй швидкості виконання. Однак, якщо вам потрібна велика екосистема інструментів та бібліотек, Python є більш розвинутою мовою для цього типу проектів (рис. 3.2).

```

/// Initialize logging for the current process.
///
/// This is not threadsafe at all, so initialization os performed through a
/// 'Once' primitive (and this function is called from that primitive).
fn init() {
    let mut directives = match os::getenv("RUST_LOG") {
        Some(spec) => directive::parse_logging_spec(spec),
        None => Vec::new(),
    };

    // Sort the provided directives by length of their name, this allows a
    // little more efficient lookup at runtime.
    directives.sort_by(|a, b| {
        let alen = a.name.as_ref().map(|a| a.len()).unwrap_or(0);
        let blen = b.name.as_ref().map(|b| b.len()).unwrap_or(0);
        alen.cmp(&blen)
    });

    let max_level = {
        let max = directives.iter().max_by(|d| d.level);
        max.map(|d| d.level).unwrap_or(DEFAULT_LOG_LEVEL)
    };

    unsafe {
        LOG_LEVEL = max_level;

        assert!(DIRECTIVES.is_null());
        DIRECTIVES = cast::transmute(~directives);

        // Schedule the cleanup for this global for when the runtime exits.
        rt::at_exit(proc() {
            assert!(!DIRECTIVES.is_null());
            let _directives: ~Vec<directive::LogDirective> =

```

Рисунок 3.2 – Логування мовою Rust

Python також має більш простий синтаксис, що може зробити його більш доступним для новачків у програмуванні. Зрештою, вибір між мовами залежить від цілей вашого проекту та ваших особистих уподобань. Якщо ви хочете швидко прототипувати модель машинного навчання, то Python може бути найкращим вибором, але якщо ви працюєте з великими обсягами даних, то Rust може стати найкращим рішенням для вас.

Використання Python та Rust для аналізу даних та машинного навчання має свої переваги та недоліки, які необхідно враховувати при виборі мови для конкретного проекту. Однак, крім технічних характеристик, варто також враховувати команду розробників та їхній досвід роботи з цими мовами. Важливо пам'ятати, що вибір мови залежить від конкретної задачі та її вимог.

3.2 Вибір IDE

IDE, або інтегроване середовище розробки – це програмне забезпечення, яке надає розробникам комплексний набір інструментів для написання, тестування, налагодження та розгортання програм. IDE є центральною частиною робочого процесу більшості програмістів, оскільки вони надають усе необхідне ефективної роботи з кодом одному місці.

Важливість IDE для розробки на Python обумовлена кількома факторами:

- продуктивність: IDE включають різні інструменти, такі як автозаповнення коду, покажчики на помилки в реальному часі, вбудовані налагоджувачі та системи контролю версій, які значно підвищують продуктивність програміста;

- особливості Python: Python має деякі унікальні особливості, які роблять його трохи відмінним від інших мов програмування. Деякі IDE мають спеціальні функції, які покращують роботу з Python, включаючи підтримку динамічної типізації, автоматичне форматування відповідно до PEP8 та інтеграцію з популярними бібліотеками Python;

– спільна робота та підтримка : Інтегроване середовище розробки допомагає спростити процес спільної роботи над проектами, оскільки воно забезпечує стандартизоване середовище для розробки. Багато IDE мають вбудовану підтримку систем контролю версій, таких як Git, що дозволяє розробникам легко відстежувати та поєднувати зміни в коді;

– навчання : Для новачків IDE може бути чудовим інструментом для вивчення Python, оскільки вони надають відразу все необхідне для початку роботи. Крім того, IDE може давати корисні поради та показувати типові помилки, що полегшує навчання.

Visual Studio Code (часто скорочується до VS Code) — безкоштовне відкрите середовище розробки від Microsoft.

Хоча вона не є спеціалізованою IDE виключно для Python, VS Code може використовуватися для розробки на Python завдяки своїм розширенням та гнучкості. VS Code була представлена вперше в 2015 році і швидко стала популярною серед розробників завдяки своїм швидким оновленням, активній спільноті та широкій підтримці розширень.

Ось кілька особливостей VS Code, які роблять його зручним для розробки на Python:

– розширення Python: Розширення Python для VS Code, розроблене Microsoft, надає безліч функцій для розробки на Python, включаючи інтелектуальне автозаповнення, лінтинг, форматування коду, налагодження та підтримку Jupyter Notebook;

– вбудоване налагодження : VS Code має вбудовані інструменти для налагодження коду, що дозволяють встановлювати точки зупинки, переглядати змінні та контролювати виконання вашого програмного коду;

– вбудована підтримка Git : VS Code включає підтримку Git «з коробки», що дозволяє вам зручно створювати коміти, виконувати push та pull оновлень, переглядати різницю між версіями без використання командного рядка;

- підтримка розширень: Однією із сильних сторін VS Code є підтримка розширень, яка дозволяє вам налаштовувати середовище розробки під свої потреби. Існує велика кількість розширень для підтримки різних мов, бібліотек, фреймворків, інструментів форматування коду та інших функцій;

- налаштування та гнучкість : VS Code дуже налаштовується. Ви можете налаштовувати все, від теми та шрифтів до поведінки редактора коду та інтеграції з іншими інструментами.

Хоча Visual Studio Code не є виключно інструментом для розробки Python, його гнучкість і підтримка розширень роблять його дуже зручним для розробки Python. Це особливо вірно для розробників, які працюють з кількома мовами або технологіями, що потребують такої гнучкості.

3.3 Аналіз рішень для генерації зображень

3.3.1 Midjourney - фотореалістична генерація зображень

Midjourney – це новаторська платформа на основі штучного інтелекту, яка використовує передові технології для створення фотореалістичних зображень з текстових описів. На платформі використовується складний метод глибокого навчання, відомий як генеративні змагальні мережі (GAN). GAN - це тип нейронної мережі, яка складається з двох конкуруючих частин: дискримінатора та генератора. Дискримінатор навчається розрізняти реальні та згенеровані зображення, тоді як генератор навчається створювати реалістичні зображення, які відповідають опису, наданому на вході. Генератор навчається створювати все більш реалістичні зображення, які точно відповідають опису на вході, завдяки цьому процесу навчання в змагальному режимі.

Завдяки використанню GAN, Midjourney може створювати зображення високої якості, які майже не відрізняються від фотографій. Тому дизайнери та маркетологи, які хочуть створювати фотореалістичні зображення для

реклами та маркетингу, знайдуть його чудовим інструментом. Midjourney дозволяє користувачам вводити текстовий опис зображення, яке вони хочуть створити, а алгоритм штучного інтелекту створить зображення, яке точно відповідає опису. Завдяки такому високому рівню контролю над вмістом зображення користувачі можуть створювати дуже персоналізовані та індивідуальні фотореалістичні зображення.

Основною перевагою Midjourney є його здатність створювати фотореалістичні зображення, але він також має деякі недоліки. Він може створювати лише ті зображення, які відповідають дійсності, і це є одним з таких обмежень. Інакше кажучи, хоча Midjourney здатна створювати надзвичайно точні та деталізовані зображення, вона не може створювати образи, які є дуже фантазійними або складними - образи, які виходять за межі того, що можна уявити в реальному світі.

Це обмеження пов'язане з тим, що комп'ютерні розвідувальні розрахунки Midjourney готуються на основі сертифікованих зображень та інформації. Діапазон вхідних даних, які використовуються для навчання алгоритму, обмежує кількість згенерованих зображень. Midjourney здатна створювати фотореалістичні зображення з високим ступенем кастомізації за допомогою введених користувачем даних, але вона не може створювати абсолютно нові зображення або такі, що виходять за рамки того, що можливо в реальному світі.

Незважаючи на це обмеження, Midjourney залишається потужним інструментом для дизайнерів і маркетологів, яким потрібно створювати висококастомізовані та реалістичні зображення для різних цілей. Завдяки здатності Midjourney створювати високоякісні фотореалістичні зображення, користувачі можуть створювати переконливі візуальні ефекти, які точно відображають їхнє бачення та ідеї.

Високий ступінь контролю над процесом створення зображень є однією з головних переваг Midjourney. Midjourney пропонує такий рівень кастомізації та точності, якого важко досягти за допомогою інших

інструментів для створення зображень, дозволяючи користувачам вводити текстові описи зображень, які вони хочуть створити.

За допомогою Midjourney дизайнери можуть вказати характеристики та деталі, які вони хочуть бачити у створеному зображенні. Завдяки цьому вони можуть створювати високоіндивідуалізовані зображення, які точно відповідають їхнім ідеям та баченню. Більше того, комп'ютерні інтелектуальні розрахунки, які використовує Midjourney, гарантують, що створені зображення точно відповідають інформаційному зображенню, створюючи надзвичайно точні та обґрунтовані зображення.

Завдяки високому рівню контролю Midjourney є чудовим інструментом для дизайнерів, яким потрібно створювати фотореалістичні зображення для різноманітних цілей. Це включає, серед іншого, дизайн продуктів, маркетинг і рекламу. Дизайнери можуть використовувати Midjourney для створення візуально привабливого контенту, який є одночасно ефективним і дуже цікавим, точно відображаючи їхні концепції та ідеї.

3.3.2 DALL-E - створення складних та образних зображень

DALL-E – це видатна сцена на основі штучного інтелекту, створена OpenAI, яка використовує поєднання GAN і трансформаторів для створення надзвичайно складних і винахідливих зображень, що виходять за рамки фотореалістичних. На відміну від Midjourney, який обмежується створенням фотореалістичних зображень на основі текстових зображень, DALL-E може створювати глибоко винахідливі та концептуальні зображення, які не пов'язані з обмеженнями реального світу.

Завдяки використанню GAN і трансформаторів, DALL-E здатна створювати зображення, які є надзвичайно фантазійними і креативними, перевершуючи те, що можливо в реальному світі.

Завдяки GAN система DALL-E здатна навчатися на основі великої кількості даних, що дає їй змогу створювати зображення, які є надзвичайно

різноманітними і різноплановими. DALL-E здатна генерувати надзвичайно деталізовані і складні зображення завдяки трансформаторам, які дозволяють зосередитися на конкретних аспектах зображення.

DALL-E – чудовий інструмент для дизайнерів і художників, які хочуть дослідити нові творчі можливості, адже він може створювати зображення, які є дуже фантазійними та абстрактними.

За допомогою DALL-E клієнти можуть поєднувати літературні та візуальні підказки для створення глибоко перероблених і складних зображень, які відображають їхнє творче бачення та думки. DALL-E є інноваційною та захоплюючою платформою для створення зображень завдяки такому рівню креативності та контролю. Вона має потенціал розширити межі можливого у сфері створення зображень за допомогою штучного інтелекту.

Однією з головних переваг DALL-E над Midjourney та іншими платформами для створення зображень зі штучним інтелектом є здатність створювати надзвичайно фантазійні та складні зображення.

Розсуваючи межі можливого за допомогою зображень, створених штучним інтелектом, DALL-E здатна створювати зображення, які виходять за рамки того, що можливо в реальному світі.

DALL-E створений для створення креативних зображень, на відміну від Midjourney, який в першу чергу займається створенням фотореалістичних зображень з текстових описів.

Художники та дизайнери, які хочуть дослідити нові творчі можливості та розширити межі створення зображень, знайдуть цей інструмент ідеальним.

Користувачі DALL-E мають можливість вводити комбінацію текстових і візуальних підказок для створення зображень, які є дуже індивідуальними і складними. Завдяки здатності платформи генерувати зображення, які не обмежуються рамками реальності, користувачі можуть створювати дуже оригінальні та унікальні зображення, що відображають їхнє творче бачення та ідеї.

DALL-E – це потужний інструмент для художників, дизайнерів і творчих людей, які хочуть дослідити нові творчі можливості та розширити межі можливого за допомогою зображень, створених штучним інтелектом, оскільки він може генерувати дуже вигадливі та складні зображення.

Ще одна важлива особливість, яка відрізняє DALL-E від інших платформ для створення зображень зі штучним інтелектом, – це високий рівень контролю над процесом створення зображень.

За допомогою DALL-E клієнти можуть поєднувати текстові та візуальні сигнали для створення складних зображень, забезпечуючи такий рівень контролю, якого важко досягти за допомогою інших інструментів епохи зображень.

Завдяки такому рівню контролю користувачі мають свободу експериментувати з різними стилями та естетикою зображень, що дозволяє їм створювати дуже індивідуальні та персоналізовані зображення, які відображають їхнє творче бачення та ідеї. Для того, щоб точно налаштувати створені зображення і досягти бажаного вигляду, користувачі можуть регулювати різноманітні параметри і налаштування.

Крім того, здатність DALL-E створювати глибоко інноваційні та складні зображення означає, що клієнти можуть спробувати різні речі з різними стилями зображень і відчутти, що вони виходять за межі того, що можна уявити в реальності.

Художники та дизайнери зможуть розширити свої творчі горизонти і в результаті цього створювати контент, який буде одночасно захоплюючим і візуально приголомшливим.

Загалом, DALL-E є чудовим інструментом для дизайнерів і художників, які хочуть експериментувати з різними стилями та естетикою зображень і розширювати межі своєї творчості, оскільки він дає їм великий контроль над тим, як створюються зображення.

Платформа вирізняється з-поміж інших інструментів для створення зображень тим, що вона може створювати зображення, які є дуже індивідуальними та складними.

Це відкриває нові можливості для створення візуального контенту.

Midjourney і DALL-E - це революційні технології створення зображень на основі штучного інтелекту, які мають свої переваги і недоліки, що роблять їх придатними для різних застосувань.

Midjourney – чудовий інструмент для дизайнерів і маркетологів, яким потрібно створювати високореалістичні та точні зображення для рекламних і маркетингових цілей, завдяки своїй здатності генерувати висококастомізовані та фотореалістичні зображення на основі текстових описів. Однак його можливості обмежені сферою реальності, і він не здатний створювати фантастичні або складні зображення, які виходять за рамки того, що можна досягти в реальному світі.

З іншого боку, DALL-E є чудовим інструментом для художників і дизайнерів, які хочуть дослідити нові творчі можливості, оскільки він чудово справляється зі створенням складних зображень, які виходять за межі того, що можливо в реальному світі. DALL-E дозволяє користувачам вводити комбінацію текстових і візуальних підказок для створення складних зображень, забезпечуючи високий рівень контролю над процесом генерації зображень. Однак, здатність DALL-E створювати зображення, які є одночасно дуже вигадливими і складними за рахунок фотореалізму, робить його менш придатним для додатків, які вимагають зображень з високим ступенем реалістичності.

Загалом, як Midjourney, так і DALL-E є творчими досягненнями, які пропонують чудові якості та недоліки в галузі епохи зображень, керованих штучним інтелектом. Конкретні вимоги та вподобання користувача в кінцевому підсумку визначатимуть, яку з двох систем використовувати: Midjourney – найкращий варіант для високоточних і реалістичних зображень, а DALL-E – найкращий варіант для фантазійних і складних зображень.

3.4 Порівняння інструментів генерації зображення

Метод парного порівняння – найпростіший із існуючих класифікаційних тестів, оскільки він передбачає зіставлення лише двох зразків продукту. Перевага методу парних порівнянь перед ранжуванням полягає в тому, що легше винести судження, оскільки супервізор одночасно потрібно порівнювати тільки двох осіб. Друга перевага в тому, що він надає можливість поставити на один рівень людей, які мають однакові здібності.

Метод парних порівнянь дозволяє провести суворий, статистично обґрунтований аналіз узгодженості думок експертів, виявити, чи випадкові отримані оцінки. Безсумнівно, процедура методу парних порівнянь складніша методу простого ранжування, але простіше методу послідовних порівнянь. Даний метод дуже простий і дозволяє досліджувати більше об'єктів (проти, наприклад, з методом рангів) і з більшою точністю.

За умов експерименту було здійснено генерацію ілюстрацій за текстовим описом у трьох різних нейромережах.

Для аналізу та зрівня було обрано наступні нейромережі:

- Midjourney;
- Stable Diffusion;
- DALL-E 2.

Для початку роботи було написано запит чат-боту GPTChat: «Уяви себе на обкладинці Forbes». Оскільки – це лише штучний інтелект, йому складно уявити себе. Однак чат розповів про своє бачення (рис. 3.3).

Вибачте, але як мовна модель ШІ я не можу з'явитися на обкладинці. Проте я можу описати, який вигляд я міг би гіпотетично мати на обкладинці Forbes. Великим планом зображений екран комп'ютера з інтерфейсом чату, а я, ChatGPT, відповідаю на різноманітні запитання в режимі реального часу

Рисунок 3.3 – Запит до ChatGPT 1

Потім ChatGPT запропонував заголовок, який може бути зображений на обкладинці (рис. 3.4).

Розвиток штучного інтелекту: познайомтеся з ChatGPT, мовною моделлю, яка революціонізує наш спосіб спілкування

Рисунок 3.4 – Запит до ChatGPT 2

ChatGPT додав, що обкладинка може підкреслити його здібності як мовної моделі, зокрема здатність генерувати текст, подібно людині, на основі вхідних даних, які він отримує.

Колір AI обрав синій. Але не виключає наявність інших відтінків на обкладинці (рис. 3.5).

Як мовну модель штучного інтелекту мене зазвичай асоціюють із синім кольором, який може символізувати технології, інновації та інтелект. Дизайн обкладинки також може містити інші кольори, які відповідають бренду Forbes, наприклад чорний, білий і сірий

Рисунок 3.5 – Запит до ChatGPT 3

Ілюстрація може бути витонченою та сучасною, з чіткими лініями та мінімалістичним підходом, а конкретні елементи дизайну, кольори та зображення ChatGPT запропонував створити дизайнеру.

Синтаксичні запити в програмах абсолютно однакові та складені англійською мовою (рис. 3.6-3.8), для найкращого порівняльного результату взято 4 згенерованих картинки (неймережі генерують 4 варіації для одного запиту). Результати повністю створені неймережею і не проходили жодної постобробки.



Рисунок 3.6 – Результат в Midjourney



Рисунок 3.7 – Результат в DALL-E 2



Рисунок 3.8 – Результат в Stablediffusion

Основними критеріями для зрівняння було визначено:

- відповідність запиту;
- унікальність;
- якість зображення;
- оперативне генерування.

Оцінка методів за наведеними критеріями представлена у табл. 3.1.

Таблиця 3.1 – Оцінка нейромережі

Нейромережа/Критерій	1	2	3	4
Midjourney	відмінно	добре	відмінно	відмінно
DALL-E 2	середнє	середнє	відмінно	середнє
Stable Diffusion	відмінно	середнє	середнє	середнє

Наступним етапом дослідження є побудова матриць зрівнянь нейромереж за кожним критерієм окремо.

Матріці наведено у табл. 3.2-3.5.

Було побудовано матриці парних зрівнянь та визначено коефіцієнти. Натомість, за допомогою коефіцієнтів за окремими критеріями було визначено загальний коефіцієнт для кожної нейромережі (табл. 3.6). Таким чином дослідження показали, що найкращою нейромережею є Midjourney.

Результати дослідження представлені у вигляді діаграми на рис. 3.9.

Таблиця 3.2 – Матриця зрівнянь за відповідністю запиту

Критерій 1	1 Midjourney	2 DALL-E 2	3 Stable Diffusion	Коеф.
1 Midjourney	1	2	1	1,3
2 DALL-E 2	0	1	1	0,7
3 Stable Diffusion	1	2	1	1,3

Таблиця 3.3 – Матриця зрівнянь за унікальністю

Критерій 2	1 Midjourney	2 DALL-E 2	3 Stable Diffusion	Коеф.
1 Midjourney	1	1	2	1,3
2 DALL-E 2	2	1	2	1,7
3 Stable Diffusion	0	0	1	0,3

Таблиця 3.4 – Матриця зрівнянь за якістю зображень

Критерій 3	1 Midjourney	2 DALL-E 2	3 Stable Diffusion	Коеф.
1 Midjourney	1	0	0	0,3
2 DALL-E 2	2	1	2	1,7
3 Stable Diffusion	0	2	1	1,0

Таблиця 3.5 – Матриця зрівнянь за оперативним генеруванням

Критерій 4	1 Midjourney	2 DALL-E 2	3 Stable Diffusion	Коеф.
1 Midjourney	1	2	2	1,7
2 DALL-E 2	2	1	2	1,7
3 Stable Diffusion	1	1	1	1,0

Таблиця 3.6 – Загальні коефіцієнти неймереж

Неймережа /Критерій	1	2	3	4	Коеф.
1 Midjourney	1,3	1,3	0,3	1,7	1,1
2 DALL-E 2	0,7	1,7	1,7	1,7	1,3
3 Stable Diffusion	1,3	0,3	1,0	1,0	1,0



Рисунок 3.9 – Результати дослідження

4 ПРАКТИЧНА ЧАСТИНА

4.1 Попередня обробка тексту

Для зчитування тексту книги в програму була використана бібліотека для роботи з текстами у форматі pdf PyPDF2 (рис. 4.1).

```
pdfFileObj = open('Lord Of The Rings.pdf', 'rb')
pdfReader = PyPDF2.PdfReader(pdfFileObj)
number_of_pages = len(pdfReader.pages)
```

Рисунок 4.1 – Зчитування тексту

Після завантаження текст потрібно очистити, видаливши зайві переноси, малюнки, символи пунктуації, номери сторінок. Для цього було створено функцію `clean_text` (рис. 4.2).

```
def clean_text(text):
    # remove numbers
    text_nonum = re.sub(r'\d+', '', text)
    # remove punctuations and convert characters to lower case
    text_nopunct = "".join([char.lower() for char in text_nonum if char not in string.punctuation])
    # substitute multiple whitespace with single whitespace
    # Also, removes leading and trailing whitespaces
    text_no_double_space = re.sub('\s+', ' ', text_nopunct).strip()
    return text_no_double_space
```

Рисунок 4.2 – Очистка тексту

4.2 Реферування тексту

Для реферування тексту за допомогою бібліотеки NLTK було створено функцію під назвою `summarize`, яка приймає речення та слова як

вхідні дані та повертає короткий зміст тексту використовуючи алгоритм на основі графіків, який ранжує речення на основі їх важливості – TextRank (рис. 4.3).

```
def summarize(text):
    word_frequencies = {}
    for word in nltk.word_tokenize(text):
        if word not in stopwords:
            if word not in word_frequencies.keys():
                word_frequencies[word] = 1
            else:
                word_frequencies[word] += 1
    maximum_frequency = max(word_frequencies.values())
    for word in word_frequencies.keys():
        word_frequencies[word] = (word_frequencies[word]/maximum_frequency)
    sentence_scores = {}
    for sent in sentence_list:
        for word in nltk.word_tokenize(sent.lower()):
            if word in word_frequencies.keys():
                if len(sent.split(' ')) < 40:
                    if sent not in sentence_scores.keys():
                        sentence_scores[sent] = word_frequencies[word]
                    else:
                        sentence_scores[sent] += word_frequencies[word]
    import heapq
    summary_sentences = heapq.nlargest(20, sentence_scores, key=sentence_scores.get)
    summary = ' '.join(summary_sentences)
    return summary
```

Рисунок 4.3 – Функція summarize

4.3 Створення запиту

Для поліпшення результату генерування зображення реферованийний текст додатково обробляється за допомогою засобу gpt-3.5 від OpenAI. Ця дія була реалізована в функції getDallePrompt (рис. 4.4).

```
from openai import OpenAI

client = OpenAI(
    api_key="XXX",
)

def getDallePrompt(text):
    completion = client.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=[
            {"role": "system", "content": "create a short image prompt for dall-e for creating book cover based on this text: " + text},
        ]
    )
    result = completion.choices[0].message.content
    return result
```

Рисунок 4.4 – Функція getDallePrompt

4.4 Генерація зображення

Для генерації зображення було створено функцію `generateWithDalle`. Яка приймає вхідний параметр – текст, відсилає запит до DALL-E та повертає зображення у форматі json (рис. 4.5).

```

from openai import OpenAI
client = OpenAI({
    api_key="xxx",
})

def generateWithDalle(text):
    image_params = {
        "model": "dall-e-3", # Defaults to dall-e-2
        "n": 1, # Between 2 and 10 is only for DALL-E 2
        "size": "1024x1024", # 256x256, 512x512 only for DALL-E 2 - not much cheaper
        "prompt": text, # DALL-E 3: max 4000 characters, DALL-E 2: max 1000
        "user": "myName", # pass a customer ID to OpenAI for abuse monitoring
        "quality": "standard",
    }

    image_params.update({"response_format": "b64_json"}) # defaults to "url" for separate download

    try:
        images_response = client.images.generate(**image_params)
        return images_response.model_dump()
    except openai.APIConnectionError as e:
        print("Server connection error: {e.__cause__}") # from httpx.
        raise
    except openai.RateLimitError as e:
        print(f"OpenAI RATE LIMIT error {e.status_code}: {e.response}")
        raise
    except openai.APIStatusError as e:
        print(f"OpenAI STATUS error {e.status_code}: {e.response}")
        raise

```

Рисунок 4.5 – Функція `generateWithDalle`

Для відображення результату генерації було використано бібліотеку `tkinter`. Яка дозволяє зручно конвертувати `b64_json` в `png` та відображати малюнки.

4.5 Результати роботи програми

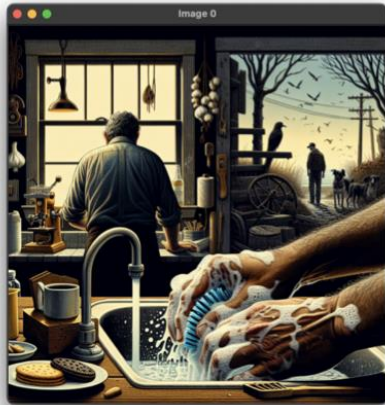
Тестування реферування та генерації обкладинки здійснювалося на базі текстового змісту таких творів: *The Fellowship of the Ring*, *Harry Potter and the Philosopher's Stone*, *The Shining*, *The Holy Bible*, *Shadows of the Forgotten Ancestors* (рис. 4.6).



The Fellowship of the Ring



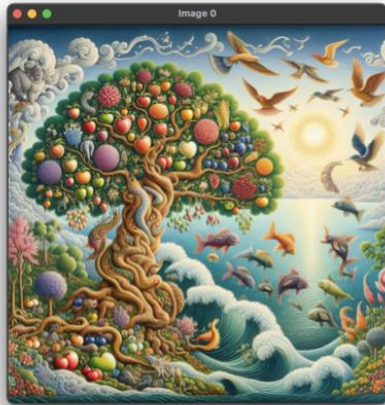
Harry Potter and the Philosopher's Stone



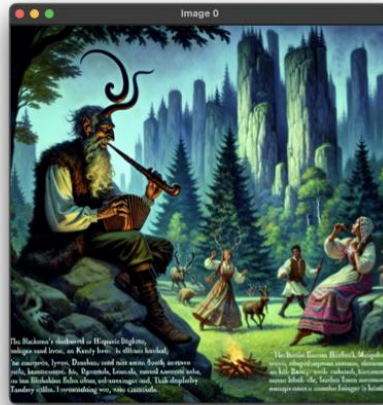
I am Legend



The Shining



The Holy Bible



Shadows of the Forgotten Ancestors

Рисунок 4.6 – Згенерованні обкладинки

5 ЕКОНОМІЧНА ЧАСТИНА

5.1 Характеристика науково-дослідного рішення

Метою даного розділу є економічне обґрунтування витрат на проведення науково-дослідної роботи (НДР), в межах якої передбачається дослідження використання штучного інтелекту для автоматичного створення обкладинок на базі текстового змісту.

Під час такого обґрунтування буде здійснено: розрахунок трудовитрат та заробітної плати працівникам, розрахунок одноразових витрат і прибутку, оцінку результатів НДР.

Реалізація НДР передбачає такі етапи:

- аналіз предметної області;
- визначення алгоритму реалізації проекту;
- дослідження використання штучного інтелекту для автоматичного створення обкладинок на базі текстового змісту;
- вибір методів для проведення експерименту;
- складання методики оптимізації зображень на веб-ресурсі;
- доведення дійсності роботи методики.

5.2 Етапи виконання НДР, їх трудомісткість та заробітна плата

Під час виконання НДР було проведено огляд існуючих теоретичних підходів у даній галузі, досліджено основні методи та алгоритми використання штучного інтелекту для автоматичного створення обкладинок на базі текстового змісту. Умовно НДР можна розділити на три етапи: підготовчий, основний і заключний.

На стадії виконання підготовчого етапу було виконано підбір і аналіз інформації для проведення відповідних до постановки завдання робіт. Проведено пошук інформації в мережі Internet та у фаховій літературі.

На етапі виконання основної частини НДР було здійснено такі роботи:

- розгляд методів на основі штучного інтелекту для автоматичного створення обкладинок на базі текстового змісту;
- аналіз розглянутих методів та відбір за певними критеріями;
- розробка методики для створення обкладинок для різного за обсягом тексту;
- доведення працездатності запропонованої методики.

У заключній частині проводяться: аналіз результатів виконання НДР, складання звіту з НДР та його захист.

Найбільш складною та відповідальною частиною при плануванні НДР є розрахунок трудомісткості робіт, тому що трудові витрати часто становлять основну частину вартості науково-дослідних робіт і безпосередньо впливають на строки розробки.

Дану роботу виконували 3 фахівця: ілюстратор, інженер програмного забезпечення, data scientist. Середня заробітна плата ілюстратора за версією сайту dou.ua становить 40 000,00 грн, інженер програмного забезпечення – 100 000,00 грн, data scientist – 110 000,00 грн.

Проведемо розрахунок трудовитрат і заробітної плати виконавця робіт.

Середньоденна заробітна плата виконавця робіт ($Z_{\text{ср.дн.}}$) розраховується за формулою:

$$Z_{\text{ср.дн.}} = \frac{Z_{\text{ср.міс.}}}{n}, \quad (5.1)$$

де $Z_{\text{ср.міс.}}$ – середньомісячна зарплата виконавця роботи;

n – число робочих днів у місяці, ($n = 22$).

Підставивши дані до формули (5.1), отримаємо середньоденну заробітну плату ілюстратора у розмірі 1818,18 грн, інженера програмного забезпечення – 4545,45 грн, data scientist – 5000,00 грн.

Етапи виконання НДР, перелік і зміст робіт, трудомісткість їх виконання, заробітна плата виконавців робіт представлені в табл. 5.1.

Таблиця 5.1 – Розрахунок трудовитрат і заробітної плати виконавців робіт

Перелік робіт	Кількість виконавців	Посада виконавця	Трудомісткість робіт, люд.-днів	Середньоденна заробітна плата, грн	Сума заробітної плати, грн
1	2	3	4	5	6
1. Підготовчий етап					
1.1. Розробка та затвердження ТЗ	1	Ілюстратор	2	1818,18	3636,36
1.2 Підготовка довідкових матеріалів та даних для виконання НДР	1	Ілюстратор	2	1818,18	3636,36
2. Основний етап					
2.1 Постановка задачі	1	Ілюстратор	1	1818,18	1818,18
2.2 Розгляд сучасних методів створення обкладинок	1	Data-scientist	3	5000,00	15000,00
2.3 Аналіз розглянутих методів та відбір за критеріями	1	Data-scientist	3	5000,00	15000,00
2.4. Розробка методики автоматизації створення обкладинок	1	Data-scientist, інженер програмного забезпечення	2	5000,00 4545,45	10000,00 9090,90
3 Тестування					
3.1 Відбір тестових зображень	1	Ілюстратор	1	1818,18	1818,18
3.2 Підтвердження результату	1	Ілюстратор	1	1818,18	1818,18
4. Заключний етап					
4.1 Аналіз результатів проведення роботи	1	Ілюстратор	2	1818,18	3636,36
4.2 Формування висновків та пропозицій за темою дослідження	1	Ілюстратор	1	1818,18	1818,18
4.3 Технічне оформлення звіту виконання НДР	1	Ілюстратор	2	1818,18	3636,36
Усього			20		70909,06

5.3 Розрахунок одноразових витрат на розробку НДР

Калькуляція собівартості розраховується відповідно до існуючих нормативних актів України.

До складу калькуляції входять такі статті витрат:

- матеріальні витрати;
- витрати на оплату праці;
- єдиний соціальний внесок;
- амортизація основних засобів (вартість машинного часу);
- витрати на спожиту електроенергію;
- інші витрати.

Матеріальні витрати визначаються витратами на матеріали, визначені їх потребою для виконання робіт, і цін, що діють на момент складання калькуляції. Для проведення НДР потрібно: механічний олівець (1 шт.) та блокнот (1 шт.). Дані матеріальні витрати потрібні для ілюстратора.

Матеріальні витрати розраховуються за такою формулою:

$$M = \sum_{j=1}^n Q_j \times C_j, \quad (5.2)$$

де M – сумарні витрати на матеріали, в тому числі малоцінні предмети, що швидко зношуються (носії, папір, канцелярське приладдя тощо), або на літературу, яка необхідна для проведення роботи, тощо;

Q_j – кількість використаних одиниць j -го виду матеріалів, $j = (1 \div n)$;

C_j – ціна одиниці j -го виду матеріалів.

Розрахунок матеріальних витрат представлено в табл. 5.2.

Таблиця 5.2 – Розрахунок матеріальних витрат

Найменування	Од. вим.	Кількість, (Q_j)	Ціна (C_j), грн	Сумарні витрати на матеріали (M), грн
Олівець механічний	шт.	1	80,00	8,00
Блокнот	шт.	1	40,00	40,00
Усього				48,00

Витрати на оплату праці розраховуються виходячи з необхідного для виконання робіт складу й кількості працівників, а також із середньомісячної заробітної плати. Відповідно до проведених розрахунків витрати на оплату праці виконавців роботи дорівнюють 70909,06 грн.

Єдиний внесок на загальнодержавне соціальне страхування (ЄСВ) – консолідований страховий внесок, збір якого здійснюється в систему загальнообов’язкового державного соціального страхування в обов’язковому порядку і на регулярній основі з метою забезпечення захисту у випадках, передбачених законодавством, прав застрахованих осіб і членів їх сімей на отримання страхових виплат (послуг) за діючими видами загальнообов’язкового державного соціального страхування.

Ставка єдиного соціального внеску (ЄСВ) дорівнює 22 % від витрат на оплату праці, тобто розмір ЄСВ дорівнює 15599,99 грн.

При виконанні НДР застосовувалися 3 комп’ютери вартістю 30000,00 грн. кожен. Вищенаведене устаткування є власністю організації виконавця, тому доцільно розрахувати суму амортизаційних відрахувань на період виконання НДР.

Амортизація основних засобів розраховується за формулою:

$$AB = \sum_{k=1}^L \frac{BO_k}{TE_k} \times T, \quad (5.3)$$

де AB – сума амортизаційних відрахувань, нарахованих під час проведення науково-дослідницької роботи;

BO_k – вартість основних засобів k -го виду;

TE_k – термін експлуатації основних засобів k -го виду, днів;

T – термін науково-дослідницької роботи, днів;

L – кількість видів обладнання.

Підставивши відомі значення до формули (5.3), отримаємо величину амортизаційних відрахувань – 1100,92 грн.

Витрати на використану обладнанням електроенергію (B_e) розраховуються за формулою:

$$B_e = M \cdot t \cdot T_{kBm}, \quad (5.4)$$

де M – потужність устаткування, тобто кількість енергії, споживаної за одиницю часу (кВт/година);

t – кількість годин використання устаткування за період проведення науково-дослідницької роботи;

T_{kBm} – тариф, тобто вартість використання 1 кВт електроенергії.

Споживна потужність комп'ютера складає 0,5 кВт за годину. Тариф споживачів за першим класом напруги, тобто 35 кВт та більше), складає 2,64 грн/кВт. Підставивши значення до формули (5.4), визначимо величину витрат на спожиту електроенергію у розмірі 211,20 грн.

До інших статей витрат відносяться адміністративні витрати (водопостачання, водовідведення, освітлення, опалення), які прийнято у розмірі 20 % від витрат на оплату праці, та вартість оплати послуг зв'язку.

Вартість оплати послуг зв'язку (безлімітний пакет Інтернет) становитиме 350,00 грн за 20 днів виконання НДР.

Генерування зображень здійснювалося за допомогою сервісу openai.com. Послуги даного сервісу становлять – 188,00 грн.

За час виконання НДР витрати на відрядження, інформаційні послуги та маркетингові заходи не мали місця.

Результати розрахунку кошторису витрат, тобто одноразових витрат, на виконання НДР «Дослідження використання штучного інтелекту для автоматичного створення обкладинок на базі текстового змісту» наведені в табл. 5.3.

Таким чином, кошторис витрат на виконання даної НДР визначає сумарні витрати за статтями п.1÷п.6 та складає 102588,98 грн.

Таблиця 5.3 – Кошторис витрат на розробку НДР

№ з/п	Стаття витрат	Сума, грн
1	Заробітна плата	70909,06
2	Єдиний соціальний внесок (22 % від п.1)	15599,99
3	Матеріальні витрати	48,00
4	Амортизація основних засобів	1100,92
5	Витрати на спожиту електроенергію	211,20
6	Інші витрати, у тому числі:	
6.1	адміністративні витрати (20 % від п.1)	14181,81
6.2	вартість послуг зв'язку	350,00
6.3	вартість послуг сервісу ChatGPT	188,00
	Усього витрати на розробку (<i>Bp</i>)	102588,98

5.4 Оцінка результатів науково-дослідної роботи

Результат – це завершальний наслідок послідовності дій, виражений якісно або кількісно. В загальному випадку оцінка результатів НДР – це визначення ефективності отриманих рішень порівняно з сучасним науково-технічним рівнем.

Відповідно до теми даної роботи можна зробити висновок про те, що результатом впровадження НДР є зменшення ціни та часу на виробництво обкладинки порівняно з неавтоматичним методом.

Результат від впровадження НДР визначається за формулою:

$$\Delta P_j = |X_{б_j} - X_{н_j}|, \quad (5.5)$$

де ΔP_j – покращення j -ої характеристики досліджуваного процесу за рахунок впровадження результатів НДР ($j = 1, m$);

m – кількість досліджуваних характеристик;

$X_{б_j}$ – базове значення j -ої характеристики;

$X_{н_j}$ – нове значення j -ої характеристики після впровадження НДР.

У якості досліджуваної характеристики обрано ціну та час розробки обкладинки. До впровадження методики створення обкладинки коштувало 500,00 грн та займало 5 днів.

Підставивши відповідні значення ціни та часу створення обкладинки до формули (5.5), визначимо результат від впровадження НДР у чисельному вигляді:

$$\Delta P_1 = |500-1| = 499,00 \text{ (грн)},$$

$$\Delta P_1 = |2880-1| = 2879 \text{ (хв)}.$$

Аналогічні розрахунки були також проведені для іншої книги та наведені у табл. 5.4.

Таблиця 5.4 – Результат від впровадження НДР

Критерій	Книга 1			Книга 2		
	до	після	різниця	до	після	різниця
Ціна, грн	500,00	1	-499,00	650,00	1	-649,00
Час виконання, хв	2880	1	-2879	3600	1	-3599

Таким чином, отриманий результат свідчить про те, що завдяки результату від впровадження НДР обкладинки, створені за допомогою розробленої методики мають значно меншу ціну та час створення. Роботу в цілому можна вважати ефективною або такою, що має високий науковий та технічний рівень.

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи було проведено дослідження використання штучного інтелекту для автоматичного створення обкладинок на базі текстового змісту.

Було проведено аналіз аналогів та спеціальної науково-технічної літератури на тему дослідження, а також систематизовано наведену в різних джерелах інформацію. Проведено аналіз нейромереж для генерування зображень.

Розглянуто методи аналізу та вибір оптимальної нейромережі. У межах дослідження було встановлено, що найефективнішим методом вирішення проблеми вибору є експертний метод.

Проаналізовані методи реферування великих за обсягом текстів, що дозволило розробити оптимальний алгоритм для створення опису зображення на базі текстового змісту.

Було здійснено вибір та обґрунтування оптимальної нейромережі, яка допоможе створити зображення для обкладинки. Експертний метод показав, що оптимальною нейромережею є Midjourney. Розроблено рекомендації щодо правильного генерування зображень, які надалі будуть використовуватися у створенні обкладинок.

В експериментальній частині роботи з урахуванням результатів теоретичних досліджень та вимог технічного завдання було розроблено програму автоматичної генерації зображення для обкладинки.

У результаті проведених економічних досліджень, можна зробити висновок про те, що кожна гривня витрат на розробку НДР забезпечує зниження витрат коштів та часу на створення обкладинки. Дана науково-дослідна робота має позитивний показник економічної ефективності.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Черняков Б. І. Зображальна журналістика в друкованих засобах масової інформації: Ілюстрована періодика від виникнення до середини ХІХ ст. / Б. Черняков.- К.: Центр вільної преси, 1998. - 115 с
2. Різун В. В. Моделювання і технологія редакторських систем / Київ. нац. ун-т імені Тараса Шевченка. / В. Різун. - К., 1995. - 200 с.
3. Різун В. В. Вплив гарнітури шрифту на сприймання тексту / Наукові записки ін-ту журналістики. / В. Різун - К., 2000. - Т.1. - С. 129–130.
4. Шевченко, В. Е. (2000). Композиція та архітектоніка друкованого видання. Вісник КНУ ім Т. Шевченка.-2000.-Вип, 8, 70-75.
5. Renee Gade. Mastering Image Creation with Midjourney AI Tools: How to Create Stunning AI Images with Midjourney. / Renee G., 2023.–pp.2–33.
6. Кулішова Н.С., Ткаченко В.П. Методичні вказівки з виконання атестаційної роботи здобувачів вищої освіти на другому (магістерському) рівні для студентів усіх форм навчання спеціальності 186 «Видавництво та поліграфія». Харків: ХНУРЕ, 2020.
7. Irbite, A., & Strode, A. (2021, May 28–29). Artificial intelligence vs designer: The impact of artificial intelligence on design practice. In Society. Integration. Education [Proceedings of the Conference] (Vol. 4, pp. 539–549). Rezekne Academy of Technologies [in English].
8. Божко Т., Ареф'єв В. Нейронні мережі як інструмент графічного дизайну. Вісник КНУКіМ. Серія «Мистецтвознавство», (48), С. 125–135.
9. Асматі Чібіашвілі, Штучний інтелект у мистецьких практиках. URL: <http://sm.mari.kiev.ua/article/view/248425> (дата звернення: 30.11.2023).
10. From Manual to Magical: How AI Design Automation is Reshaping the Industry. URL: <https://www.itex.ai/blog/from-manual-to-magical-how-ai-design-automation-is-reshaping-the-industry> (дата звернення: 30.11.23).
11. Automating Design Review with Artificial Intelligence and BIM: State

of the Art and Research Framework. URL: <https://ascelibrary.org/doi/10.1061/9780784482421.045> (дата звернення: 30.11.23).

12. How MidJourney And DALL•E 2 Help Designers To Create Unique Concepts. URL: <https://parametric-architecture.com/howmidjourney-and-dalle-2-help-designers-to-createunique-concepts/> (дата звернення: 03.12.23)

13. AI-generated Book Covers: Is Midjourney the Future of Book Cover Design. URL: <https://mibllart.com/blog/midjourney-book-cover-design/> (дата звернення 03.12.2023).

14. Midjourney as an artificial intelligence system. URL: <https://cutt.ly/g0IcRIJ> (дата звернення 03.12.2023).

15. Stable Difusion. URL: <https://cutt.ly/00IcYW5> (дата звернення 03.12.2023).

16. DALL-E. URL: <https://cutt.ly/N0IcD1C> (дата звернення 03.12.2023).

17. What is conceptual design? And how to wrap your mind around ideation. URL: <https://is.gd/JEtVEC> (дата звернення 03.12.2023).

18. Генерація зображень з опису або як пряцює нейромережа DALL-E. URL: <https://www.imena.ua/blog/how-the-dall-e-neural-network-works/> (дата звернення 03.12.2023).

19. Моніторинг та оцінка діяльності структурного підрозділу організації. URL: <https://cutt.ly/2NdBDug> (дата звернення 03.12.2023).

20. Станіслав Білецький, Штучний інтелект у дизайні. URL: <http://sm.mari.kiev.ua/article/view/248425> (дата звернення: 30.11.2023).