УДК 004.9

ТЕХНОЛОГІЯ ПОРОДЖЕННЯ ПРОДОВЖЕННЯ ПІСЕНЬ НА ОСНОВІ СТРАТЕГІЙ ГЕНЕРАЦІЇ ТЕСТУ, TEXTMINING І МОВНОЇ МОДЕЛІ T5

Медяков О. О. – магістр кафедри «Інформаційні системи та мережі», Національний університет «Львівська політехніка», Львів, Україна.

Висоцька В. А. – канд. техн. наук, доцент, доцент кафедри «Інформаційні системи та мережі», Національний університет «Львівська політехніка», Львів, Україна.

АНОТАЦІЯ

Актуальність. Наперед навчені великі мовні моделі, на сьогодні – це локомотив розвитку не лише NLP, а й систем глибинного навчання загалом. Моделі трансформери здатні розв'язувати фактично усі задачі, що наразі існують, за умови виконання певних вимог та практик їх навчання. В свою чергу, слова, речення та тексти є базовим і найважливішим способом комунікації між інтелектуально розвиненими істотами. Звичайно, мовлення і тексти використовуються для донесення певних емоції, подій тощо. Один з основних напрямків використання мови для опису пережитих емоцій – це пісні з текстами. Проте, часто через необхідність збереження рими та римування, розмірності віршових рядків, структури пісні тощо, артистам приходиться використовувати повторення рядків у текстах. Крім того, процес написання текстів може бути тривалим.

Метою дослідження є розробка інформаційної технології генерації продовження текстів пісень на основі моделі машинного навчання T5 з (SA, specific author) та без (NSA, non-specific author) врахування стилю автора.

Метод. Для процесу генерації важливим є питання вибору стратегії декодування. Проте, заміть того, щоб надати перевагу конкретній стратегії, у системи буде підтримка множини стратегій. Зокрема такі 8 стратегій: Contrastive search, Top-p sampling, Top-k sampling, Multinomial sampling, Beam search, Diverse beam search, Greedy search, та Beam-search multinomial sampling.

Результати. Розроблено модель машинного навчання для генерації продовження текстів пісень з допомогою великих мовних моделей, зокрема моделі T5, для прискорення, доповнення та підвищення гнучкості процесу написання текстів до пісень.

Висновки. Створена модель показує відмінні результати генерації продовження текстів пісень на тестових даних. Аналіз вихідних даних показав, що модель NSA має менш деградаційні результати, а для моделі SA необхідно збалансовувати кількість тексту для кожного автора. Обраховано кілька текстових метрик як BLEU, RougeL та RougeN для кількісного порівняння результатів моделей та стратегій генерування. Значення метрики BLEU є найрізноманітнішим, і його значення значно змінюється, залежно від стратегії. При цьому Rouge метрики мають меншу варіативність, менший розмах значень. Для порівняння використано 8 різних методик декодування для генерації тексту, що підтримуються бібліотеку transformers. З усіх результатів порівняння текстів видно, що метрично найкращим методом генерації текстів пісень є beam пошук та його варіації, зокрема променевий семплінг. Contrastive search зазвичай перевершував звичайний жадібний підхід. Методи top-p та top-k не мають однозначної переваги один на одним, і в різних ситуаціях давали різні результати.

КЛЮЧОВІ СЛОВА: генерація тексту, мовна модель T5, Transformers, стиль автора, Contrastive search, Top-p sampling, Top-k sampling, Multinomial sampling, Beam search, Diverse beam search, Greedy search, та Beam-search multinomial sampling.

АБРЕВІАТУРА

ІС – інтелектуальна система;
ІТ – інформаційна технологія;
ШІ – штучний інтелект;
ПО – предметна область;
BLEU – Bilingual Evaluation Understudy;
CNN – Convolutional Neural Network;
LCS – Longest Common Subsequence;
ML – machine learning;
NLP – Natural Language Processing;
NSA – Non-Specific Author;
RNN – Recurrent Neural Network;
SA – Specific Author;
T5 – Text-To-Text Transfer Transformer.

НОМЕНКЛАТУРА

T – довжина рядка, та кількість кроків генерації;
t – поточний крок;
w_t – слово (гіпотеза), отримане на кроці t;
$w_{<t}$ – підланцюжок від початкового (першого) до t–1 слова (гіпотези);
$w_{k:t}$ – підланцюжок від i до t слова (гіпотези) включно, при цьому $t,k \in [1, T]$;
V – словник, тобто множина можливих слів;
W_0 – спеціальний початкове слово (токен, гіпотеза) ланцюжка;
EOS – символ завершення ланцюжка (з англ. end of sequence), де $W_0, EOS \in V$;
n – гіперпараметр як кількість (number of beams), розмір (beam size) або ширина (beam width) променю;
C_t – множина всіх можливих новий стрічок на основі попередньої множини гіпотез W_{t-1};
α – штраф;
s – функція, косинусоїдна подібність векторів;
h_{wi} – векторне представлення обраховане моделлю для конкатенованого рядка $w_{<t} \circ w_i$.

ВСТУП

Стратегія генерації тексту або метод декодування тексту в мовних моделях – це алгоритм вибору наступного слова послідовності із результатів мовних моделей, зважаючи на попередні слова [1–2]. Існування різних підходів до декодування зумовлене різноманітністю завдань, призначень та архітектури моделей, а також розвитком методологій в NLP [3–4].

DOI 10.15588/1607-3274-2023-4-15

Метою дослідження є розробка інформаційної технології генерації продовження текстів пісень моделлю T5 з та без врахування стилю автора. Відповідно до мети, основними задачами роботи є наступні:

– створення навчальної та тестової вибірки даних, їх опрацювання, стандартизація та підготовка;

– fine-tuning двох моделей типу T5 для задачі генерації рядків пісень;

– проектування ІС для використання цих моделей користувачами;

– експериментальна апробація натренованих моделей ML;

– аналіз та обговорення отриманих даних як результату експериментальної апробації.

Об'єктом дослідження є процес штучної генерації текстів моделями-трансформерами типу T5. Предмет дослідження є процес донавчання мовних моделей генерації для створення продовжень текстів пісень (з та без врахування стиля певного автора).

Створення такого проєкту матиме кілька ефектів:

– Науково-технічний та соціальний, що виражатиметься у створенні нового набору даних (спеціальної форми для конкретної задачі, проте відкритий для модифікацій), публікації донавчених моделей;

– Потенційно прискорить та збільшить гнучкість процесу створення тексту для пісень, оскільки ІС буде здатна створювати кілька варіантів стрічок продовження наявних рядків;

– Потенційно прискорить експериментування з різними текстами для створення кінцевої композиції.

1 ПОСТАНОВКА ПРОБЛЕМИ

Необхідно розробити ІТ генерації продовження текстів пісень з допомогою великих мовних моделей, зокрема моделі T5, для прискорення, доповнення та підвищення гнучкості процесу написання текстів до пісень з/без врахування стиля певного автора. Використання достатньо потужних та сучасних мовних моделей, зокрема T5, забезпечить якість виконання завдання, за умови створення якісного набору тренувальних даних, дотримання правил донавчання моделей та вибору правильної стратегії генерування тексту. Підхід до завдання як чистого текст-до-тексту, що підтримує T5 модель [2, 5], дозволить використовувати одну модель для генерації текстів у стилі багатьох авторів, оскільки зашифрувати автора пісні можна в самому вхідному тексті. Потенційно, такий підхід дозволяє створити одну модель для двох підзадач (генерування з та без задання стилю автора), проте з метою порівняння можливостей відтворення текстів авторів прийнято рішення побудови двох окремих моделей, що буде навчено за однакових умов.

Для процесу генерації важливим є питання вибору стратегії декодування. Проте, заміть того, щоб надати перевагу конкретній стратегії, у ІС буде підтримка множини стратегій, зокрема:

1. Жадібний пошук (Greedy search);
2. Променевий пошук (Beam search);
3. Урізноманітнений променевий пошук (Diverse beam search) [6];
4. Поліноміальна вибірка (Multinomial sampling);
5. Променево-пошукова вибірка (Beam-search multinomial sampling);
6. Топ-k вибірка (Top-k sampling);
7. Топ-p вибірка (Top-p sampling);
8. Контрастивний пошук (Contrastive search).

Методики 1,2,3 і 8 є детермінованими, а 4–7 – стохастичними. Кожна з цих методик є авторегресійною [2], що можна проінтерпретувати як припущення, що розподіл генерованого ланцюжка спрощується до добутку умовних ймовірностей наступних слів. Тобто:

$$P(w_{1:T}|W_0)=\prod^{T}_{t=1}P(w_t|w_{<t},W_0).$$

У всіх випадках визначення T відбувається в процесі генерації, і рівний такому t, при якому $w_t = EOS$. Загалом, для порівняння, у роботі використано 8 різних методик декодування для генерації тексту, що підтримуються бібліотеку transformers. Для безпосередньої розробки, а також навчання моделі, підготовки даних та інших кроків процесу імплементації ІС обрано мову Python та множу необхідних бібліотек. Найважливіша бібліотека – transformers, дозволяє завантажити, донавчити та зберегти T5 модель та відповідний токенизатор. Модель є об'єктом з TensorFlow [13].

2 ОГЛЯД ЛІТЕРАТУРИ

В науковому світі прийнято експериментувати з різними стратегіями генерації тексту, підбираючи та вивчаючи кращу для тієї чи іншої задачі. У даній роботі розглядається дві мовні моделі-трансформери T5 [2, 5], донавчені для задачі генерації продовження стрічок текстів пісень. Ці моделі розв'язують одну й туж задачу, проте з різним підходом. Метою даної роботи є порівняння якості відтворення генерований текстів пісень за спеціальними текстовими метриками, для двох моделей, при використанні різних стратегій генерації.

Перша модель, яка надалі називатиметься NSA, навчена для завдання продовження пісень без будь-яких умов, а друга модель – SA, розв'язує завдання генерації при вказані стилю автора. Для другої моделі при навчанні додатковим параметром додано хто є автором того чи іншого набору рядків та їх продовження. При генерації встановлено певні додаткові обмеження та параметри, зокрема обмеження довжини стрічки – від 8 до 128 токенів, температура функції sotfmax – 0,98, штраф повторювання [3] – 0,98. Всі інші налаштування є специфічними для кожної зі стратегій.

Варто зауважити, що надалі у роботі ототожнюватиметься поняття рядка, стрічки та ланцюжка як послідовності елементів, зокрема слів. При цьому поняття слова, як елементу ланцюжка,

DOI 10.15588/1607-3274-2023-4-15

може бути замінене на токен або гіпотезу або кандидата. Ця заміна можлива, оскільки в роботі токеном вважається індекс слова у словнику.

При аналізі аналогів до ІС, що розробляється у цій роботі, необхідно враховувати як близький так і далеких аналогів, що розв'язують ту саму задачу, так і аналоги мовних моделей, які можна використати для поставленої мети. Основними непрямими аналогами для ІС порівняння стратегій генерації – є інші проєкти, що дозволяють створювати продовження текстів пісень. Наприклад:

– MuseNet від OpenAI: MuseNet є музичною мовною моделлю, яка може генерувати продовження музичних композицій.

– Jukedeck: Jukedeck є платформою, яка використовує ШІ для автоматичної генерації музики. Вона здатна створювати продовження музичних композицій з урахуванням стилю, настрою та інших параметрів.

– Magenta Project від Google: Magenta є відкритим джерелом для дослідження творчості, музики та ML. Вона включає різні моделі та інструменти для генерації музики, включаючи генерацію тексту продовження пісень.

– Amper Music: Amper Music є платформою, яка використовує ШІ для автоматичної генерації музики для відео та інших медійних проектів. Вона може генерувати продовження музичних композицій з врахуванням заданих параметрів.

Проте, концептуально вищим рівень аналогів, що необхідно розглянути є вид нейронних мереж та моделей, що можна використати для NLP задач, та зокрема для генерації тексту. Можна виділити таку множину близьких конкурентів-моделей, що можна використати для генерації тексту: CNN, RNN або Transformers.

CNN є найменш адаптованими до задачі генерації тексту (хоча вони мають місце для задач класифікації тексту тощо), що досить очевидно, зважаючи на їх неможливість роботи з потоковими даними та природою згортки.

RNN – це перша якісна альтернатива, оскільки такий вид нейронних мереж дозволяє розв'язувати задачу послідовного генерування текстів пісень. Проте, існує кілька аспектів, у яких рекурентні мережі програють трансформерам [1]:

– Рекурентні мережі не здатні зберігати довгострокові зв'язки у тексті, що дуже важливо для генерації зв'язних текстів пісень, при цьому через природу роботи механізму уваги, трансформери здатні це робити;

– Процес навчання рекурентних мереж обмежений, оскільки вони страждають від проблеми згасання градієнтів, а також їх роботу неможливо паралелізувати, що не є проблемою для трансформерів;

– Складний процес додавання додаткових параметрів вхідного тексту, зокрема, бажаний стиль тексту.

Проте, моделі Transformers мають і свої недоліки, що можуть вплинути на роботу ІС, наприклад:

– Потреба надвеликих об'ємів даних для якісної роботи, проте цю проблему можна розв'язати використанням трансферного навчання, тобто використати наперед навчені модель і довчити для конкретного завдання (у даному випадку – генерації текстів пісень);

– Деякі моделі мають тенденцію до генерування некреативного тексту, зокрема перша версія T5 [7];

– Якість роботи мережі залежить від якості даних та організації процесу донавчання;

– Потреба великого об'єму обчислювального ресурсу, що має вплив на екологію планети [8].

Звичайно, вплив більшості недоліків можливо мінімізувати відповідними діями та виконання практик донавчання Transformers. Використання трансферного навчання та Transformers, зокрема моделі T5, надає ІС множину переваг, а саме [12]:

– Контекстне генерування, тобто врахування контексту вхідних рядків тексту для генерації вихідних;

– Простота реалізації умовного генерування, із вказанням бажаного стилю, та безпосередня можливість вивчення властивостей авторства;

– Можливість переходу на більш якісні або новіші версії конкретних архітектур Transformers.

Враховуючи наданий аналіз, саме моделі-Transformers з використанням трансферного навчання є найбільш оптимальним варіантом досягнення поставленої мети.

3 МАТЕРІАЛИ ТА МЕТОДИ

При розгляді кожної зі стратегій як прикладу генерації з її використанням, на вхід мережі типу NSA подано кілька стрічок тексту пісні, що мережа раніше не бачила і відповідно декодовано. Вхідна стрічка для прикладу, авторства Т. Свіфт та А. Дезнера (фрагмент з "long story short" [15]):

"*Past me*
I wanna tell you not to get lost in these petty things
your nemeses
will defeat themselves before you get the chance to swing
and he's passing by
rare as the glimmer of a comet in the sky
and he feels like home
if the shoe fits, walk in it everywhere you go
and I fell from the pedestal
right down the rabbit hole"

1. Greedy search. Найпростіший у реалізації метод декодування, жадібний алгоритм пошуку, генерує наступним словом те, що має найбільшу ймовірність появи для кожного кроку t, тобто:

$$w_t = \text{argmax}_w P(w|w_{<t}).$$

Очевидно, що простота реалізації не гарантує якісні результати. Жадібний підхід страждає від

DOI 10.15588/1607-3274-2023-4-15

повторюваності, несумісності та деградації тексту [18], проте основним його недоліком є пропуски слів з високою ймовірністю, приховані за словом з низькою ймовірністю. Приклад декодованого рядка з використанням цього алгоритму:

```
i'm a liar, a liar, a liar
i'm a liar, a liar, a liar
i'm a liar, a liar, a liar
```

Видно, що жадібне декодування створює повторювальний текст, проте варто зауважити, що мала різноманітність тексту не супроводжується граматичними помилками.

2. Beam search. Ідея стратегії полягає у збережені кожного кроку n найбільш ймовірних гіпотез (тобто слів), та обрання кінцевим результатом той ланцюжок, що має найбільшу ймовірність загалом. Таким чином променевий пошук розв'язує головну проблему жадібного підходу. На Рис. 1 подано приклад роботи променевого пошуку при n=2.

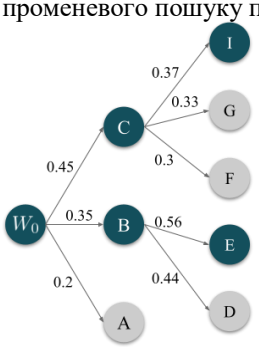

Рисунок 1 – Приклад побудови ланцюжка променевим пошуком при n=2

На кожному кроці t алгоритму виконується дві основі операції – формування множини всіх можливих новий стрічок C_t на основі попередньої множини гіпотез W_{t-1}, та обрання з неї нової множини гіпотез – n ланцюжків з найбільшим значенням функції правдоподібності.

$$C_t = \{w_{<t} \circ w_t \mid w_{<t} \in W_{t-1},\ w_t \in V\}. \quad (1)$$

$$W_t = \underset{w_{<t+1} \in C_t,\ |W_t|=n}{\operatorname{argmax}} P(w_{<t+1} \mid W_0). \quad (2)$$

Відповідно, враховуючи формули (1)–(2), вихідна стрічка описується залежністю з формули (3):

$$w_{1:T} = \underset{w'_{<T+1} \in C_T}{\operatorname{argmax}} P(w'_{<T+1} \mid W_0). \quad (3)$$

Такий пошук завжди генерує стрічку з вищою загальною ймовірністю, ніж жадібний підхід, проте не має гарантії, що отриманий ланцюжок є найбільш ймовірним результатом. Очевидні недоліки такого підходу як вища обчислювальна складність та повторюваність у результатних ланцюжках. Крім того алгоритм теж має тенденцію до генерування повторюваних стрічок та формування «слабо дивних» (англ. less surprising) текстів, у порівнянні з людським текстом [17]. Для прикладу, при n=4, та враховуючи змінену температуру й введення штрафу повторів, при декодуванні променевим пошуком створено таке продовження тексту пісні:

```
and he's a monster
oh, i'm gonna get you out of it
and you'll be able to see me through the cracks
but if the shoe fits, walk in it everywhere you go
you fell from the pedestal
right down the rabbit hole
```

Створений текст не містить повторів, через наявність штрафу, порте більшість згенерованого тексту повторює частину вхідної стрічки. Крім того існують різні методи покращення результатів генерації з променевим пошуком, такі як N-грамні штрафи [19] та інші [20]. Менш очевидні недоліки випливають з [21], де показано, що beam підхід найкраще підходить для генерації текстів, де довжина вихідного тексту легко передбачувана (як при перекладі), проте менш якісно працює для задач відкритої генерації, як та, що розглядається у роботі.

3. Diverse beam search [6]. Цей підхід є спробою вдосконалити результати класичного променевого пошуку, шляхом побудови більш різноманітних наборів променів. Відповідно до оригінальної статті [6], результати такого підходу значно кращі за звичайний beam пошук. Крім кількості променів, цей алгоритм має додатковий параметр – кількість груп променів (number of beam groups). Групи вибираються так, щоб вони були досить чіткими порівняно з іншими, а в кожній групі використовується звичайний пошук за променем [6]. Проте, як видно з результатної стрічки нижче, цей алгоритм видає значно гірші результати за променевий пошук, що найімовірніше викликано процесом донавчання (fint-tuning) моделі, а не властивостями самого алгоритму:

```
and he's a monster
he's a monster
he's a monster
he's a monster
he's a monster
he's a monster
```

Отримана стрічка зациклена на 6-грамі he's a monster, де 6 токен є спеціальним роздільником <sep>, і тому його опущено.

4. Multinomial sampling. Проблема з класичними детерміністичними алгоритмами це тренд до створення повторюваних ланцюжків. Одне з простих рішень – це додавання випадковості до процесу вибору наступного слова. Базовий спосіб досягнення цього – це проста вибірка слова з розподілу над всіма елементами словника – формула (4).

$$w_t \sim P(w \mid w_{<t}). \quad (4)$$

DOI 10.15588/1607-3274-2023-4-15

По-перше, це призводить до того, що усі слова з ненульовою ймовірністю можуть потрапити до ланцюжка, що потенційно зменшує повторюваність в результаті. По-друге, оскільки процес вибору не є детерміністичним, то модель, при однаковій вхідній стрічці, генеруватиме різні результати при кожному проході. На жаль цей метод має важливий недолік – тенденція генерувати нісенітниці або несумні тексти [17]. Один з способів нівелювати ймовірність створення несумісних N-грам – це зменшення «температури» функції softmax, тобто збільшити значення правдоподібності слів з високою ймовірністю та зменшення для слів з низькою ймовірністю. При «температурі» 1 – отримані оцінки впевненості з softmax є класичними, а чим вона нижче (від 0 до 1) тим «гостріше» виглядатиме розподіл токенів. Приклад одного з варіантів тексту, декодованого за допомогою семплінгу:

```
and it's mad as hell
don't know why you're dying
those who loved me will never love you back
i'm richer, more powerful
i'm all i want from life
i want you to be free
```

Текст маж достатньо високу різноманітність, не містить повторів з вхідної стрічки, згенеровані рядки граматичні та синтаксично коректні, проте загальна комбінація стрічок не є цілком зв'язаною. Інший приклад згенерованого тексту за цим алгоритмом:

```
not about winning
you can wreck me
dj schwer, who deserves my blood
reach me, skinny dipping, dj fit
goddamn
you're right there, man
you're so good, you know you're got a killer soon
last call out for me
as you grow increasingly understand
this is a metamorphosis
and i am a master of disguise
don't be afraid to step
```

При однакових вхідних параметрах та налаштуваннях ця стрічка є менш сумісною за попередню, адже частина словосполучень не мають змісту. Наприклад, три 2-грами reach me, skinny dipping, dj fit належать до одного рядка, і були відокремлені спеціальний сепаруючим токеном, проте вони абсолютно без сенсу, і не відносяться до вхідного тексту.

5. Beam-search multinomial sampling. Комбінація стохастичного процесу вибору та детерміністичного променевого пошуку. При такій стратегії кожного кроку t з $P(w \mid w_{<t})$ незалежним чином вибирається n слів. Використовуючи цю стратегію, один з отриманих результатів має такий вигляд:

```
and he's passing by
rare as the glimmer of a comet
and he feels like home
if the shoe fits, walk in it everywhere you go
and he feels like home
if the shoe fits, walk in it everywhere you go
and he feels like home
```

Отриманий ланцюжок повністю складається з слів вхідної стрічки, що, аналогічно до попередньої стратегії, ймовірно викликано процесом навчання.

6. Top-k sampling [16]. Досить потужна модифікація стохастичного підходу вперше запропонована у [16]. Ідея полягає в тому, що з розподілу залишають лише топ k токенів з найбільшими ймовірностями. Далі обчислюється нова функція розподілу відповідно до нової множини токенів, і вже з цього нового розподілу проводять вибірку слова. Нехай множина слів, що обирається кожного кроку t позначається як $V^t_{top\text{-}k}$, тоді:

$$V^t_{top\text{-}k} = \underset{w \in V,\ |V^t_{top\text{-}k}|=k}{\operatorname{argmax}} P(w_t \mid w_{<t}). \qquad (5)$$

$$w_t \sim P_{V^t_{top\text{-}k}}(w \mid w_{<t}),\ w_t \in V^t_{top\text{-}k}. \qquad (6)$$

Така фільтрація кандидатів частково нівелює ймовірність появи дивних чи несумісних комбінацій слів, проте головний недолік – це відсутність адаптивності розміру множини кандидатів до їх розподілу. Без такої адаптивності можливі два негативних наслідки. Перший – при дуже «гострому» оригінальному розподілі, коли після перерозподу будуть утворюватися несумісні рядки. Другий – відсутність різноманітності генерованого тексту пісень, що виникатиме при рівномірності ймовірностей відібраних k токенів. Очевидно, що при k=1 алгоритм поводитиметься як жадібний, а при k=0 або k=card(V) – як звичайний семплінг. Приклади отриманих стрічок, з використанням цього алгоритму при k=15 та зменшеній температурі softmax функції:

```
the glimmer's a shard of glass
and the color's a diamond
the fire's burning bright
but he's not a monster
he's more like a monster
i'm in love, love's a sliver of glass
and his touch is sweet as honey
he's like a prince, a prince, a prince
i love you
```

Тексту властива слабка повторюваність, відсутність римування, проте текст граматично коректний. Проте, при зменшені k до 5, згенерована стрічка швидко деградує:

DOI 10.15588/1607-3274-2023-4-15

```
i'm just stumbling, stumbling, stumbling, stumbling
just stumbling, stumbling
stumbling, stumbling
stumbling, stumbling, stumbling
stumbling, stumbling
```

7. Top-*p* sampling [17]. Для корегування недоліків top-*k* підходу у [17] запропонували ідею відбору не просто найімовірніших *k* токенів, а стільки токенів, щоб їх кумулятивна ймовірність була не менше за *p*. Аналогічно до top-*k*, після фільтрації токенів їх функція розподілу перерозподіляється, після чого відбувається вибирання. Якщо позначити множину кандидатів токенів на кожному кроці *t* як $V^t_{top\text{-}p}$, то наступне слово вибиратиметься з розподілу відповідно до формули (7).

$$w_t \sim P_{V^t_{top\text{-}p}}(w \mid w_{<t}),\ w_t \in V^t_{top\text{-}p}. \qquad (7)$$

При *p*=1 алгоритм перетворюється у класичний семплінг, а при *p*=0 – у жадібний алгоритм. Приклад стрічки продовження тексту пісні згенерований при *p*=0,96:

```
but the light comes back to you
like a sailor flying through the sky
and the wind blows through the trees
you can't miss him, baby
he's just a little bit of magic
you're just a tiny bit away
and you're just a little bit farther
and you're just a little bit further
you're just a tiny bit further
i've been doing my
```

Згенерований текст і справді виглядає як продовження тексту пісні, хоча й процес генерування було перервано на незавершеній фразі. Текст граматично коректний, хоча його сумісність та зміст важко оцінити як якісні.

8. Contrastive search [18]. Щоб побороти проблему генерації виродженого тексту (англ. degenerate solutions), тобто повторювального та/чи несумісного тексту, що виникає при використанні вже розглянутих детерміністичних та стохастичних алгоритмів, автори [18] запропонували зіставне рішення – контрастивний метод декодування. Оригінальна робота авторів пропонує не лише алгоритм декодування, а і заміну класичному методу максимізації функції правдоподібності при навчання трансформерів для генерації чи в загальному – моделюванні мов. Проте, враховуючи контекст цієї роботи, та використання звичайного алгоритму навчання для двох моделей, що розглядаються, цю заміну не буде розглянуто. Важливо зауважити, що не використання цього алгоритму впливає є на можливості зіставного декодування. Ідея алгоритму генерування наступного слова складається з двох ключових аспектів [18]:

– гіпотеза повинна бути обрана з множини найімовірніших (як при top-*k*);

– згенерований результат має бути досить дискримінаційним щодо попереднього контексту.

Математично, алгоритм на кроці *t* описується як:

$$w_t = \underset{w \in V^t_{top\text{-}k}}{\operatorname{argmax}}\{(1-\alpha)\times p(w|w_{<t}) - \alpha\times(\max\{s(h_w,h_{wj})|1\le j\le t-1\})\}.$$

Формулу можна розбити на дві частини, відповідно до двох раніше описаних аспектів. Перша – обчислює впевненість моделі (тобто вихід з softmax), а друга (так званий штраф виродження) – вимірює, наскільки дискримінаційний кандидат *w* щодо попереднього контексту $w_{<t}$. Оскільки цей алгоритм є детерміністичним, і враховуючи тенденцію інших подібних алгоритмів створювати повторювані рядки, то низьке значення штрафу α автоматично породжувати однакові рядки, зміст яких залежатиме від параметру *k*.

Наприклад, при α = 0,5, модель генерує такі рядки:

k = 3
```
i'm the one who got away
i'm the one who got away
i'm the one who got away
i'm the one who got away
i'm the one who got away
i'm the one who got away
i'm the one who got away
```

k = 10
```
hey, look at me
i'm in the back of the car
and he's like a wildfire
i'm on the other side
i'm the one who got away
i'm the one who got away
i'm the one who got away
i'm the one who got away
```

k = 20
```
now i'm in the back of the car
waiting for you to come back
i'm like a second-hand hero
no, i'm not a bad guy
i'm just a bad guy
i'm just a bad guy
i'm just a bad guy
```

k = 50
```
now i'm in the back of the car
waiting for you to come undone
i'm making my way home
no one's gonna do it like me
i'm an old codger
i'm the anti-hero
i'm the anti-hero
i'm the anti-hero
```

При збільшенні кандидатів, збільшується і різноманітність перших рядків, проте процес генерації збігається до повторювання. Проте, при збільшенні штрафу, наприклад, до 0,95, виникає протилежна ситуація – при збільшенні кількості кандидатів алгоритм вибиратиме найбільш різноманітний токен, що призводить до нісенітниць. Тому необхідно знайти оптимальну комбінацію двох параметрів. Як видно з прикладу нижче, при невеликому значенні k модель генерує сумісний текст, навіть вставляючи вигуки, як частини тексту пісень. При *k*=13 у ланцюжку з'являється непов'язані комбінації слів, що не мають змістовне навантаження.

DOI 10.15588/1607-3274-2023-4-15

$k = 3$

i fell from a pedestal (woo, woah)
right into the rabbit hole (ooops, woah, aww)
and now a day has come that's just for me

$k = 5$

i know that the end is near (yo, nope),
but it feels like the first step

k=13:

now i need an air freshened life
get the hang of what life in our neck of delinque, lenor mcallist

Питання відкритої генерації продовження пісень важко оцінювати з точки зору змісту, рими, римування, розмірності віршів тощо. Проте, можливо порівняти та оцінити наскільки близький текст, що генерує мережа, до того, що очікувався. Для цього необхідно використовувати спеціальні текстові метрики. У даній роботі пропонується розглянути три метрики: BLEU [22], RougeL та RougeN [23].

1. BLEU-метрика, створена для оцінки машинного перекладу [22], проте вона користується достатньою популярністю і серед інших задач обробки природної мови. Алгоритм обчислення метрики ґрунтується на порівнянні N-грам в оригінальному та згенерованому тексті. Для обчислення метрики обчислюється точність для кожної довжини N-грам (відношення співпадінь до кількості N-грам цієї довжини). Далі обчислюється середнє геометричне точності, вводяться спеціальні штрафи, та обчислюється кінцева оцінка в межах [0, 1], де більше значення відповідає за кращий результат.

2. RougeL зі сімейства метрик Rouge [23], призначене для оцінки підсумовувань текстів, проте, аналогічно до BLEU адаптоване і для інших задач. Приставка L походить від LCS, тобто найдовшої спільної підстрічки. Ідея алгоритму полягає в обчисленні LCS, далі обчислення класичних метрик класифікації – точність (precision) та повнота (recall), як відношення довжини LCS до довжини згенерованого та очікуваного тексту відповідно. Після цього обчислюється F1 оцінка точності та повноти. Саме F1 значення використовуватиметься для порівняння результатів різних мереж.

3. RougeN. Аналогічно до попередньої метрики, RougeN є частиною сімейства Rouge метрик. На відміну від RougeL, використовує N-грамний підхід. При обчисленні метрики обраховується перекриття n-грамами (де n – заданий параметр) згенерованої стрічки до реальної, і далі аналогічно до RougeL, обчислюється precision, recall та F1 [23].

У роботі використано офіційну імплементацію цих метрик бібліотекою KerasNLP [24–25].

4 ЕКСПЕРИМЕНТИ

Для побудови необхідного ПЗ необхідно чітко визначити основні об’єкти ПО порівняння стратегій генерації тексту. До таких об’єктів можна віднести:

– Алгоритми декодування: дослідження стратегій декодування тексту вимагає розуміння різних алгоритмів, які використовуються для згенерування послідовності слів. Це можуть бути як чисто детерміністичні, стохастичні алгоритми чи їх комбінації. Важливо визначити переваги, недоліки та вплив цих алгоритмів на згенерований текст;

– Мовні моделі: дослідження в області, що розглядається в роботі, вимагає розуміння різних мовних моделей, зокрема тих, які використовуються для генерації тексту. У цій роботі цей аспект ПО обмежено до мовних моделей типу T5, процесу їх навчання, text-to-text підходу до NLP задач;

– Метрики оцінки: Для порівняння стратегій декодування необхідно використовувати певні метрики, що визначають якість згенерованого тексту. Це можуть бути метрики, які оцінюють подібність створених, декодованих стрічок з тими, що очікувались як результат генерації тексту, зокрема тексту продовження пісні.

Похідні об’єкти, що мають більший рівень абстракції, проте є не менш важливі для обговорення:

– Проблеми з надмірною генерацією – один з найважливіших аспектів аналізу є розгляд проблем, пов’язаних з деградованою генерацією тексту, тобто наявність повторення, неконкретності, несумісності або незв’язності. Розуміння цих проблем і розробка методів для їх вирішення є важливим кроком у вдосконаленні стратегій декодування;

– Адаптація до конкретних завдань: Враховуючи різноманітність завдань, для яких використовуються мовні моделі, важливо вивчати, які стратегії декодування найкраще працюють у конкретних випадках. Деякі завдання можуть вимагати швидкості, інші – більшої точності або креативності в генерації тексту. У даній роботі визначення конкретна задача, що розглядається – процес генерації тексту продовження пісні.

Для коректної побудови ІС та її специфікації необхідно визначити основні дефініції, терміни та об’єкти ПО. Проте ширина сфери генерації штучних текстів є достатньо потужною, тому необхідно визначити межі ПО. На рис. 2 продемонстровано діаграму прецедентів пропонованої ІС. Роль цієї діаграми показати які функціональні вимоги повинна мати системи, щоб мати змоги виявити межі ПО. З діаграми видно, що концептуально найвищим прецедентом ІС є опрацювання вхідного запиту користувача на генерацію продовження тексту пісень. Враховуючи описані прецеденти, а також особливості розробки ІС на базі великих мовних моделей, можна виділити три важливих аспекти ПО:

– Створення навчального датасету – процес опрацювання та підготовки текстових даних;

– Донавчання (fine-tuning) мовної моделі – процес навчання наперед навченої моделі для виконання конкретного завдання;

– Генерація тексту пісні – процес використання моделі та декодування її результатів.

Кожен з цих аспектів містить певні терміни та процеси та варіанти, якими необхідно оперувати для досягнення поставленої мети.

DOI 10.15588/1607-3274-2023-4-15

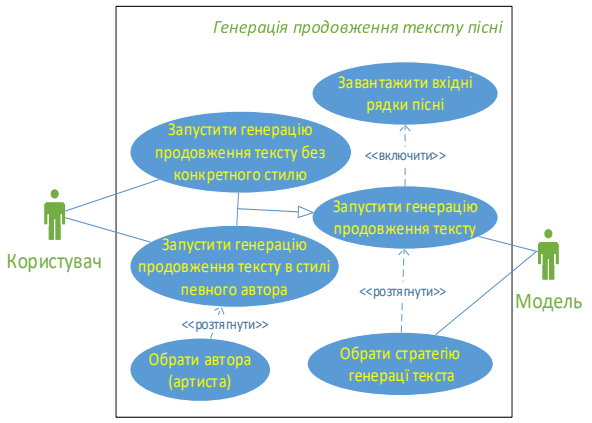

Рисунок 2 – Діаграма прецедентів системи генерації продовження текстів пісень

Створення навчального датасету є найважливішим кроком для ІС, що використовують ШІ, оскільки якість даних є необхідною умовою для побудови моделі, що відповідатиме поставленим вимогам.

З цієї частини ПО необхідно виділити, що означає генерація продовження тексту пісні. Це питання постає, оскільки означити це завдання можна кількома способами. Наприклад, класичне завдання мовних моделей – це відновлення пропущених (замаскованих) слів чи N-грам у тексті, так зване моделювання мови. За таким ж принципом можна означати процес генерації продовження пісень, тобто маскуючи закінчення рядків або випадкові n-грами з тексту пісень, і навчати модель їх відновлювати. Такий підхід вимагає від потенційних користувачів надавати на вхід не тільки текст, а місця де їм необхідно доповнити текст. Натомість у цій роботі пропонується інший підхід – пострічкове доповнення тексту пісні. Замість того аби продовжувати кожну стрічку окремо, модель буде намагатися відтворити кілька наступних стрічок. Для цього у текст буде додано спеціальний токен розділення рядків, і модель буде намагатися генерувати його в тому місці, де закінчуватиметься рядок. Другий аспект ПО – це процес fine-tuning мовної моделі. Основна мета процесу – покращити результати моделі та забезпечити більш точні та відповідні генерації тексту продовження пісень. Цей аспект включає три кроки: створення специфікованих даних (описаний раніше), вибір архітектури та переналаштування моделі.

Для контексту даної роботи, важливо відмітити другий крок – вибір архітектури. Цей процес є складним, оскільки існує багато різних моделей з різними розмірами, шаровою структурою та кількістю параметрів. При виборі архітектури треба враховувати особливості задачі, наявні обчислювальні можливості, наявні ресурси пам'яті та враховувати вплив процесу навчання моделей на навколишнє середовище. Останній аспект, який необхідно розглянути – це безпосередній процес генерації тексту з мовною моделлю. Для опису цього процесу необхідно ввести кілька означень.

– Токенизація – це розбиття стрічок на окремі компоненти (токени), якими можуть бути слова, символи чи фрагменти слів. При цьому кодування при токенизації – це процес відображення токенів у їх числове представлення.

– Декодування токенів – процес відображення індексів у відповідні частини мови (слова, символи тощо).

– Стратегія генерації або стратегія декодування [3] – це алгоритм вибору наступного слова, залежно від попередніх. Вибір стратегії впливає на безпосереднє використання моделі при генерації, має суттєвий вплив на кінцевий результат, сумісність та якість згенерованого тексту.

Весь процес генерації тексту продовження пісень, в межах сформованого контексту можна подати у вигляді послідовності операцій, оформлених у діаграму діяльності з рис. 3. Після введення основних термінів та підпроцесів, можливо сформувати опис головного процесу ІС – опрацювання запиту на генерацію тексту продовження пісні. На рис. 4 виведено відповідну діаграму діяльності. Об'єкти, що використовуються на рис. 4 є абстракціями, що не потребують специфікації. Опис обраний методів та засобів розробки ІС можна розділити на дві частини, перша – засоби для розв'язання задачі генерації, друга – технології розробки програмного забезпечення.

Обраною архітектурою моделі-трансформера є T5 [2]. Особливість цієї моделі – її повнота (енкодер – декодер модель), гнучкість та використання текст-до-текст підходу. Такий підхід дозволяє уніфікувати одну модель для кількох задач, наприклад генерацію тексту пісень в стилі різних авторів, шляхом додавання так званого taskspecific префіксу.

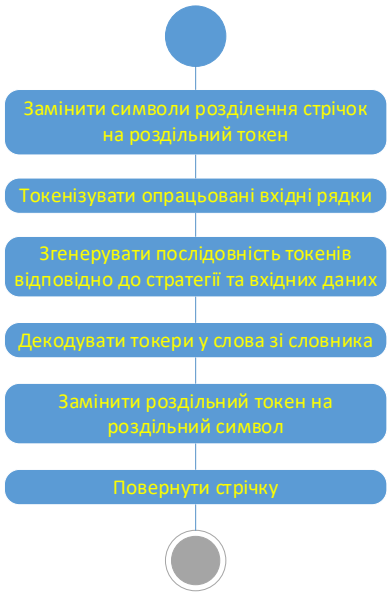

Рисунок 3 – Діаграма діяльності процесу генерації тексту продовження пісні

DOI 10.15588/1607-3274-2023-4-15

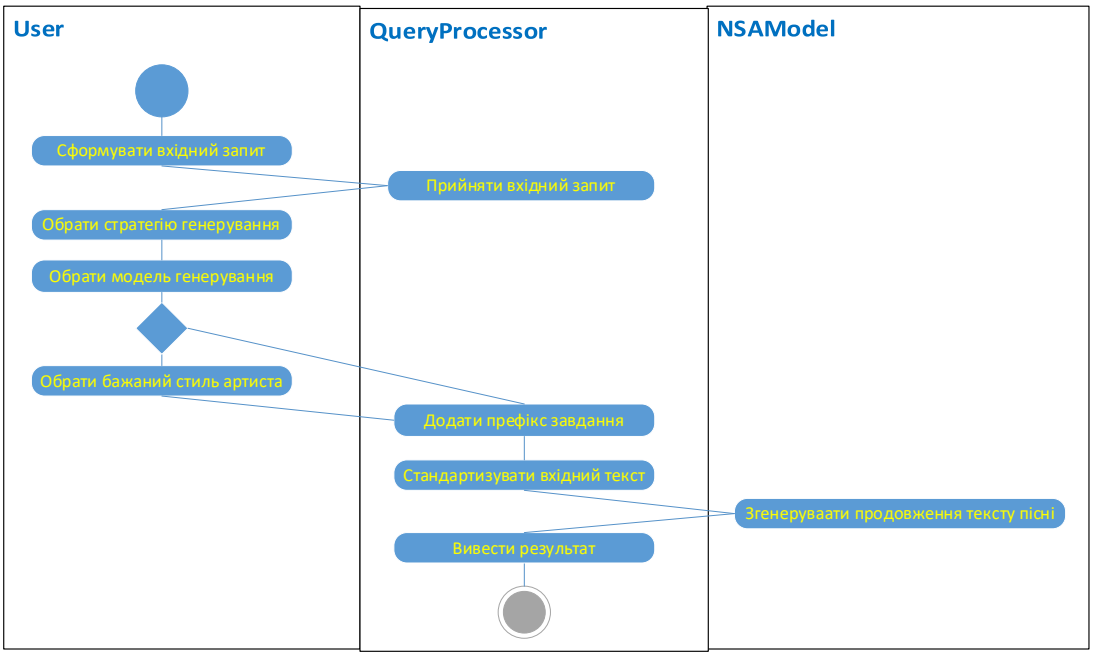

Рисунок 4 – Діаграма діяльності процесу опрацювання запиту на генерацію тексту продовження пісні

Інші переваги моделі:

– Контекстне генерування, тобто врахування контексту вхідного тексту для генерації вихідного;

– Направленість генерації, тобто використання спеціальних токенів та масок;

– Спектр різних розмірів, що дозволяє підібрати модель згідно обчислювальних обмежень та потреб.

Відповідний токенизатор та детокенизатор для цієї моделі працює на sentecepiece [9], що імплементовує модель subword units токенизації. Використання такого токенизатора має низку переваг, наприклад, він не залежить від мови, легкий та дуже швидкий [9]. Відповідно до авторів цього токенизатора, він є кращою альтернативою для задач генерації ніж subword-nmt [10] чи WordPiece [11]. На рис. 5 відображено схему взаємодії основних, додаткових та допоміжних об’єктів ПО, у вигляді діаграми класів.

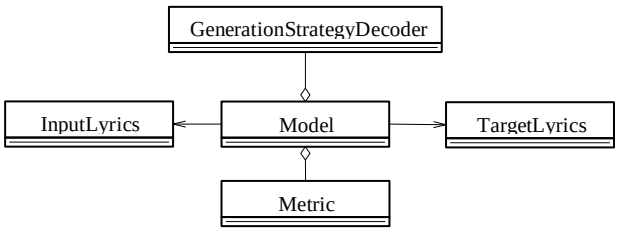

Рисунок 5 – Діаграма класів взаємодії об’єктів

5 РЕЗУЛЬТАТИ

Для використання конретного алгоритму декодування, необхідно розглянути їх імплементації в бібліотеках, що використано для побудови двох мовних моделей, що були раніше описані. Такої бібліотекою є transformers. Відповідно, для використання жадібного алгоритму, при використанні методу генерації тексту, необхідно вказати таку комбінацію параметрів:

```
model.generate(**processed_text, do_sample=False,
**gen_params)
```

Для beam search та променево-пошукова вибірки відповдіно:

```
model.generate(**processed_text, do_sample=False,
num_beams=num_beams, **gen_params)
model.generate(**processed_text, do_sample=True,
num_beams=num_beams, **gen_params)
```

Для урізноманітненого променевого пошуку:

```
model.generate(**processed_text, do_sample=False,
num_beams=num_beams,
num_beam_groups=num_beam_groups, **gen_params)
```

Для звичайного семплінгу відповідний метод має вигляд:

```
model.generate(**processed_text, do_sample=True,
num_beams=1, top_k=0, **gen_params)
```

Стохастичні методи top-k та top-p мають такі комбінації параметрів:

```
model.generate(**processed_text, do_sample=True,
num_beams=1, top_k=top_k, **gen_params)
model.generate(**processed_text, do_sample=True,
num_beams=1, top_k=top_k, top_p = top_p,
**gen_params)
```

А контрастивний пошук, в свою чергу, таку:

```
model.generate(**processed_text,
penalty_alpha=penalty_alpha, top_k=top_k,
**gen_params)
```


DOI 10.15588/1607-3274-2023-4-15

Для проведення експериментів та порівняння результатів необхідно мати відповідний інструментарій (стратегії декодування та метрики) та вхідні дані. Такими даними у цій роботі є три підмножини тестових даних, які використовувалися для валідації результатів навчання SA та NSA моделей. Кожна з підмножини є текстами пісень одного артиста.

Для підготовки даних необхідно для початку їх отримати. Для цього використано базу даних сервісу Genius [14] станом на літо 2021 рік у вигляді json файлу. Для створення даних відібрано 10 різних артистів, після чого відібрано тексти їх пісень. Загалом отримано 626 унікальних пісень. Кожен текст пісні – це безпосередньо стрічки слів та довідникова інформація (частини структури пісні або хто з співавторів виконує конкретний текст). Перший крок опрацювання тексту – очистка від довідникової інформації та розбиття тексту на стрічки.

Після підготовки текстових даних необхідно розвити стрічки на початок та продовження, тобто на вхідні та вихідні дані. Перед опрацюванням тексту треба визначити скільки пар вхід-вихід можна утворити з кожної пісні. Для цього проведено дослідницький статистичний аналіз для кількості стрічок у піснях, а також кількості слів та унікальних слів у піснях. На Рис. 6 відображено гістограми, boxplot та статистика кожної з характеристик.

Використовуючи дані з Рис. 6 розроблено процес розбиття стрічок на вхідні вихідні. Цей процес повинен враховувати кілька вимог, зокрема: пара вхід-вихід має бути послідовними слова пісні, довжина вхідний та вихідних стрічок повинна варіюватися від екземпляра до екземпляра, кількість вхідних та вхідних стрічок мають бути не менше встановленого мінімуму.

Для об'єднання стрічок вхідної та вихідної групи використано спеціальний токен <sep>. Цей токен також додано до токенизатора. В результаті розбиття кожної пісні на кілька пар вхідних-вихідних стрічок отримано 1874 навчальних екземплярів та 465 тестових. Для ілюстрації прикладу процесу перетворення тексту пісні на один екземпляр датасету на Рис. 7 виведено відповідне зображення.

Проведено донавчання двох мовних моделей NSA та SA для задачі генерації продовження тексту пісень. Для обох моделей як базову обрано t5-base. Ця версія T5 містить 223 мільйони параметрів. Як раніше описано, у роботі розроблено дві моделі. Перша вирішує задачу генерації проводження пісень, а друга – генерацію в стилі певного автора. Всього є 10 авторів, що відповідає 10 артистам, чиї пісні відібрано для навчання. Для першої моделі, яка надалі називатиметься NSA, task specific префікс виглядає так "continue lyrics:". Відповідно, перед початком донавчання моделі, усі вхідні тексти пісень доповнюються цим префіксом. Відповідно для другої моделі, що матиме назву SA, такий префікс залежить від автора тексту, проте загальна форма така "continue lyrics as [певний автор]". Налаштування гіперпараметрів для моделей є однаковими:

– Кількість епох: 3;
– Розмір партії даних: 2 записи;
– Оптимізатор: Adam з швидкістю навчання $5 \cdot 10^{-5}$.

Метрики навчання – значення функції втрат, перплексивність та точність топ-3. Остання метрика визначає точність передбачення необхідного токена, враховуючи три найймовірніші результати. Результати навчання для кожної з моделей оформлені у вигляді графіків кривих навчання, для NSA моделі – рис. 8, а для SA – рис. 9.

Аналіз кривих навчання вказує на те, що модель NSA матиме менш деградаційні результати, а для моделі SA необхідно збалансовувати кількість тексту для кожного автора. Після розробки та навчання моделей, можливо застосовувати моделі для генерації тексту. У якості контрольних прикладів розглянуто два варіанти, перший – розгляд безпосередньо згенерованого тексту для конкретного вхідного запиту, а другий – кількісна оцінка якості моделі для множини вхідних запитів. Загалом, для порівняння, у роботі використано 8 різних методик декодування для генерації тексту, що підтримуються бібліотеку transformers. Незважаючи на те, що існує можливість вибору конкретної стратегії декодування, у контрольну прикладі розглянуто всі 8. Для обох моделей використано однакову вхідну стрічку, однакові штрафи за повторюваність, зменшено так звану температуру функції softmax, та однакові налаштування параметрів стратегій генерування.

Для різнобічності оцінки можливостей мовних моделей, бажано мати вхідні дані різної розмірності. Для цього, обрано три артисти, чиї тексти займають найбільшу, найменшу та середню кількість від усієї навчальної та тестової вибірок, зокрема, тексти:

– артиста-1 – 25% навчальної та тестової вибірок;
– артиста-2 – 9% навчальної та тестової вибірок;
– артиста-3 – 3% навчальної та тестової вибірок.

Така різноманітність дозволить дослідити вплив кількості стрічок артиста, що мовна модель бачила в процесі fine-tuning, на значення метрик для обох моделей. Емпірично, знання про відсоток даних артиста від усіх має грати роль лише для SA моделі, де цей артист вказується, при цьому NSA модель нічого не знає про авторство тексту, що необхідно продовжувати. Порядок подання результатів оцінки значень метрик є таким: першим подано результати продовження текстів на SA моделі, тобто з вказанням бажано стилю (автора пісні), другим подано результати тих самих даних при використанні NSA моделі. Налаштування стратегій генерування, що зафіксовані для усіх процесів декодування:

– Променевий пошук (та похідні методи): кількість променів = 4;
– Diverse-beam: кількість груп променів = 4;
– Top-k: k = 5;
– Top-p: p = 0,96;

DOI 10.15588/1607-3274-2023-4-15

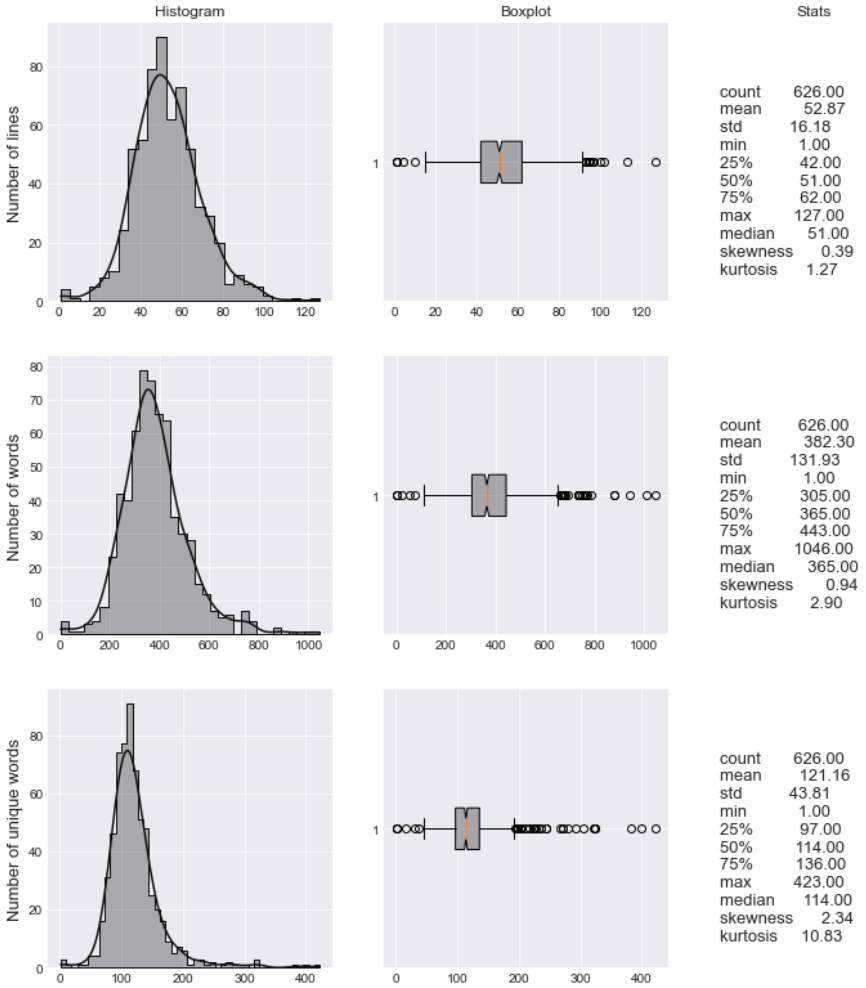

Рисунок 6 – Статистичний аналіз кількості стрічок, слів та унікальних слів у відібраних піснях

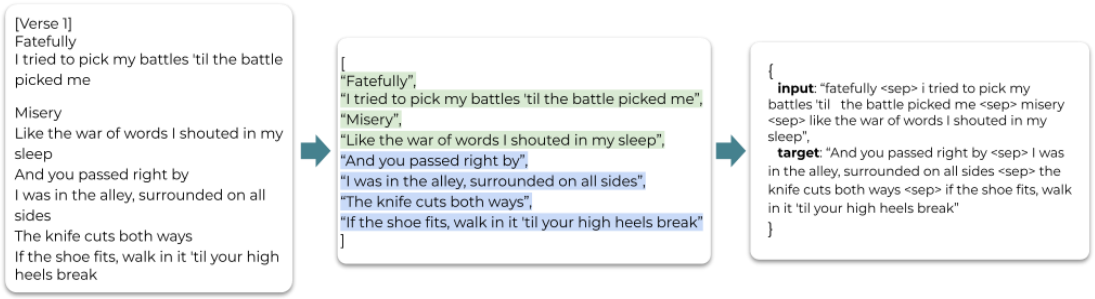

Рисунок 7 – Приклад опрацювання тексту пісні (чиста, розбиття та стрічки, формування пари вхід-вихід)

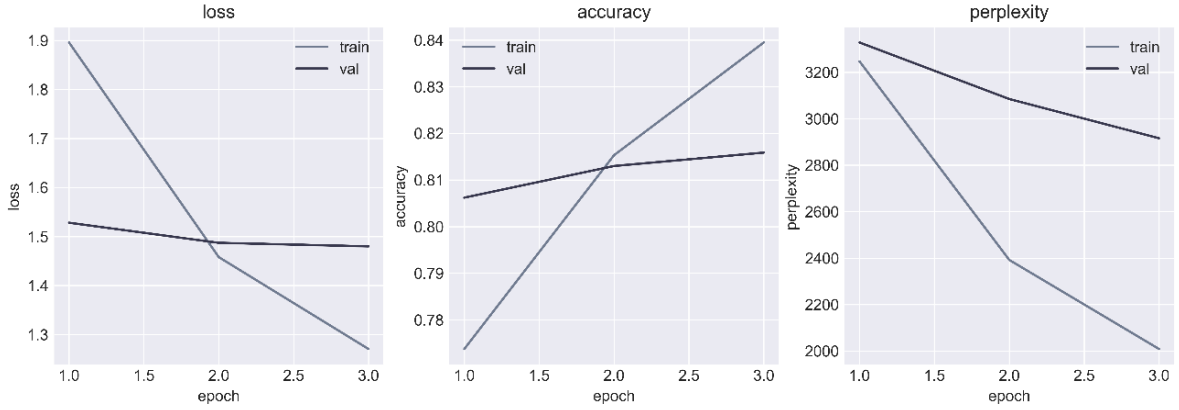

Рисунок 8 – Криві навчання моделі NSA

DOI 10.15588/1607-3274-2023-4-15

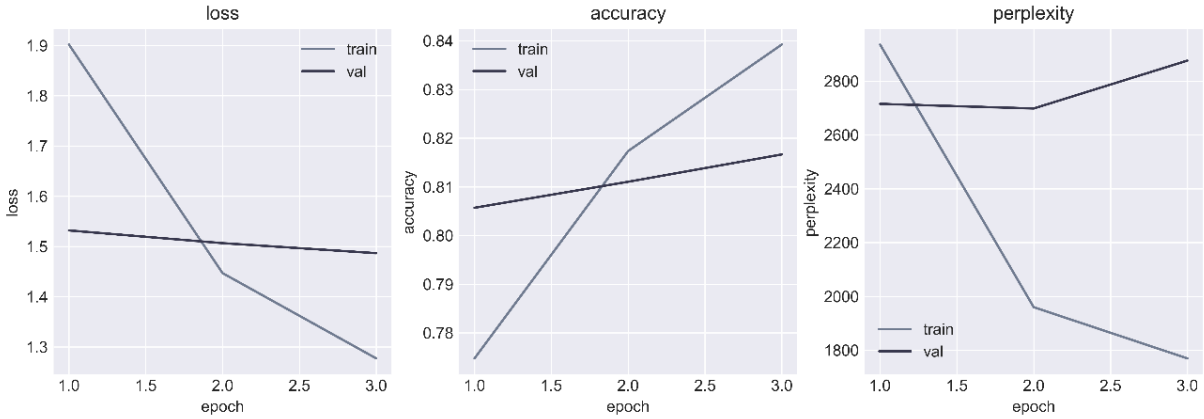

Рисунок 9 – Криві навчання моделі SA

– Констрастивний пошук: k = 3, штраф виродження = 0,9;

Для метрик RougeN порядок n-грами рівний 2, а для метрики BLEU від 1 до 4.

Результати NSA моделі виведено на рис. 10–11, а з SA – рис. 12–13.

Для другого контрольного прикладу відібрано множину вхідних-вихідних пар стрічок, та обраховано кілька текстових метрик, для кількісного порівняння результатів моделей та стратегій генерування. Такими метриками обрано BLEU, RougeL та RougeN. На Рис. 12 виведено значення метрик для надвеликої вибірки з тестових даних для моделі NSA, оскільки її результати є кращими, за відповідні з SA (рис. 13). Усі результати з рис. 8 та 9 не містять повторюваності тексту, проте частина з текстів є граматично некоректною, частина має не логічні поєднання слів. Променевий пошук, top-p та урізноманітнений променевий пошук містять аномально довгі рядки.

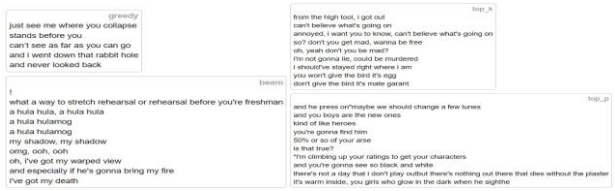

Рисунок 10 – Результати генерування стратегіями greedy пошук, beam пошук, top-p та top-k

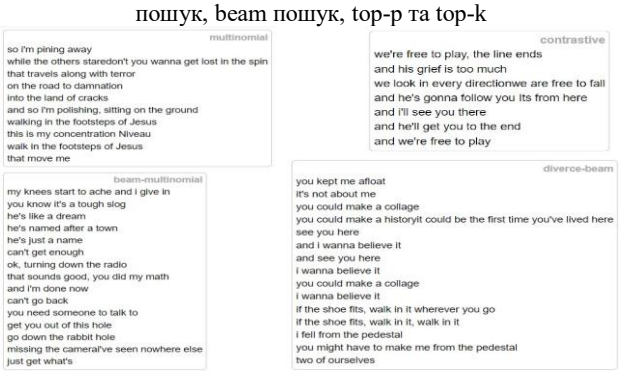

Рисунок 11 – Результати генерування стратегіями вибірки, променевого семплінгу, diverce-beam та контрастивним пошуками

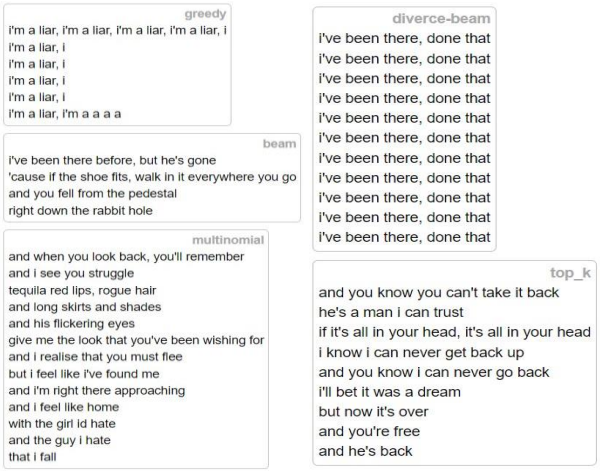

Рисунок 12 – Результати генерування 5 стратегіями з SA моделі

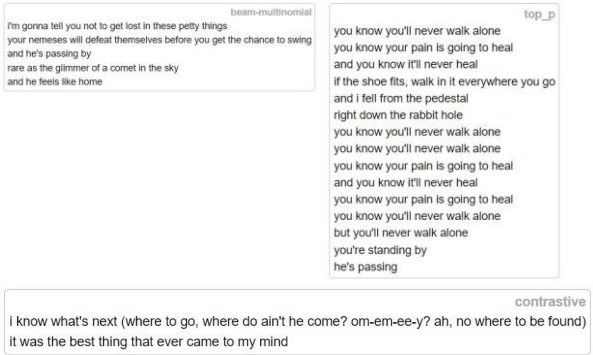

Рисунок 13 – Результати генерування 3 стратегіями з SA моделі

Результати з SA моделі мають видимі ознаки деградації тексту, повторюваність та несумність. Ймовірно, ці проблеми є ознаками перегляду навчального процесу.

6 ОБГОВОРЕННЯ

На рис. 14 для Артиста-1 виведено результати метрик з використанням моделі, що не містить інформації про авторство текстів. Якщо розглядати значення метрик окремо, то BLEU значення діаметрально змінилися, наприклад, двічі зменшилося значення для семплінгу, майже у два рази зменшилося значення для контрастивного методу, проте удвічі збільшилося для diverse-beam пошуку, майже утричі зросли для top-*k*.

DOI 10.15588/1607-3274-2023-4-15

Перш за все, найкращий результат розділяють променевий пошук та променевий семплінг. Результати при жадібному алгоритмі не змінилися. З рис. 14 видно, наскільки зміна стратегії змінює значення метрики. В даному випадку, стратегії променевого семплінгу має найкращі результати.

Значення метрик текстової подібності з моделі SA (рис. 15) стосуються автора з найбільшою кількістю даних на SA моделі. Найкращий результат, відповідно до усіх трьох метрик, досягнуто при променевому пошуку. Найгірший результат залежить від обраної метрики. Значення BLEU є найрізноманітнішим, і його значення значно змінюється, залежно від стратегії. При цьому Rouge метрики мають меншу варіативність, менший розмах значень.

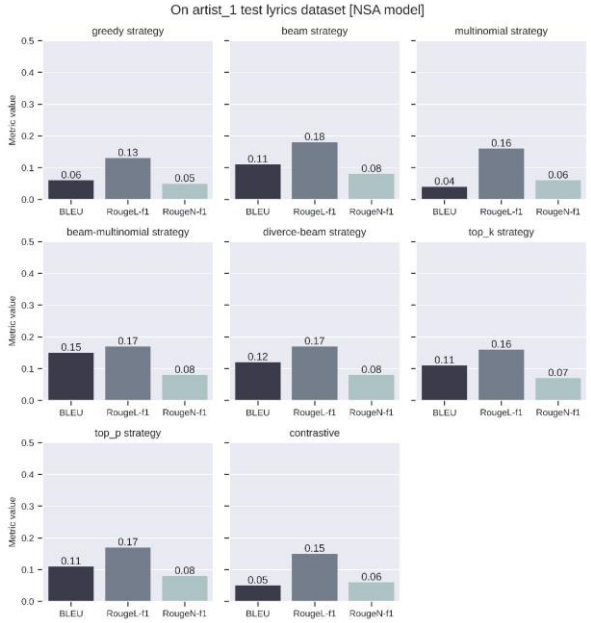

Рисунок 14 – Значення метрик текстової подібності з моделі NSA на даних артиста-1

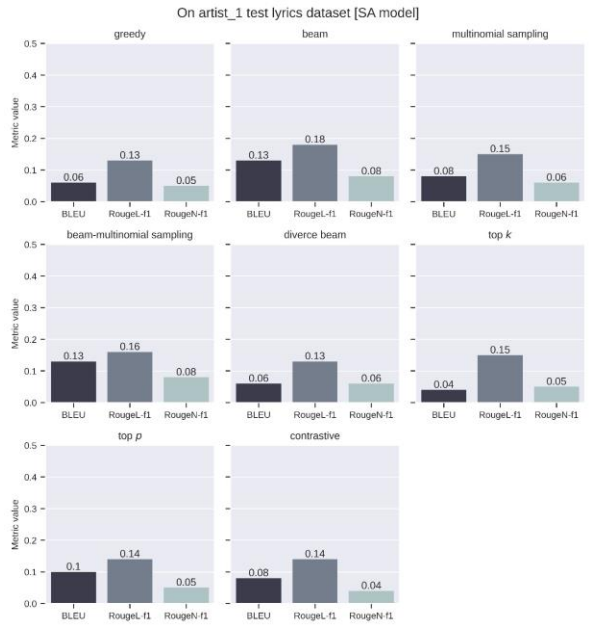

Рисунок 15 – Значення метрик текстової подібності з моделі SA на даних артиста-1

Значення обох Rouge метрик зросли або не змінилися у всіх випадках. Такі результати можу вказувати на те, що навчання на незалежних від авторства текстах, у загальному краще розв'язують задачу генерації, за умови що використано дані, що займають значну частину навчальної вибірки. Для перевірки такої гіпотези необхідно розглянути результати для інших авторів.

Результати для другого автора при вказані його стилю незначно відрізняються від аналогічних для Артиста-1 (рис. 16–17). Найкращі значення знову при використані beam алгоритму. Порівняння результатів Артиста-2:

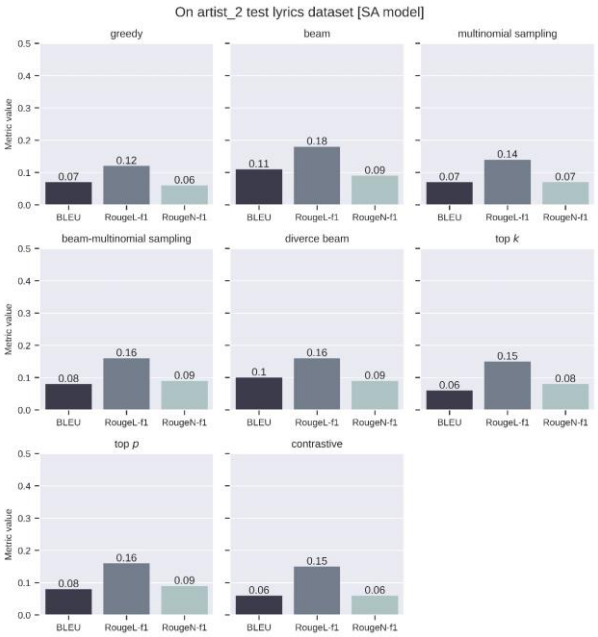

Рисунок 16 – Значення метрик для різних стратегій генерації на даних артиста-2 (модель SA)

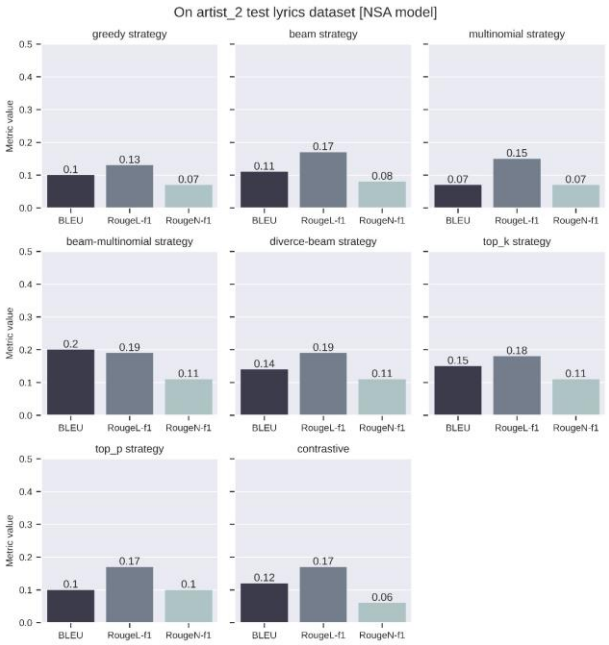

Рисунок 17 – Значення метрик для різних стратегій генерації на даних артиста-2 (модель NSA)

DOI 10.15588/1607-3274-2023-4-15

Рис. 17 демонструє результатні значення метрик для артиста-2 при використанні NSA моделі. Найцікавіший факт цього результату – найбільші значення BLEU метрики серед усіх результатів, що розглядаються у роботі. Значення з рис. 17 є більшими або рівними відповідним з рис. 16. Найімовірніше, що такі аномальні кращі результати для автора, чиї тексти не займають більшість датасету, пов'язані із загальністю та простотою цих текстів. Оскільки NSA модель будує узагальнення серед усіх текстів пісень, що їй надано при навчальному процесі, то ймовірно, що модель вибудували певне усереднення або шаблони для текстів. Через це, тексти пісень, що не є складними, або мають багато повторень або використовують загальні шаблонні фрази є найлегшими для відтворення моделлю. Така властивість моделі не вказує на якість текстів, що вона генерує, а більше на необхідність перегляду процесу її навчання [7].

Порівняння результатів Артиста-3:

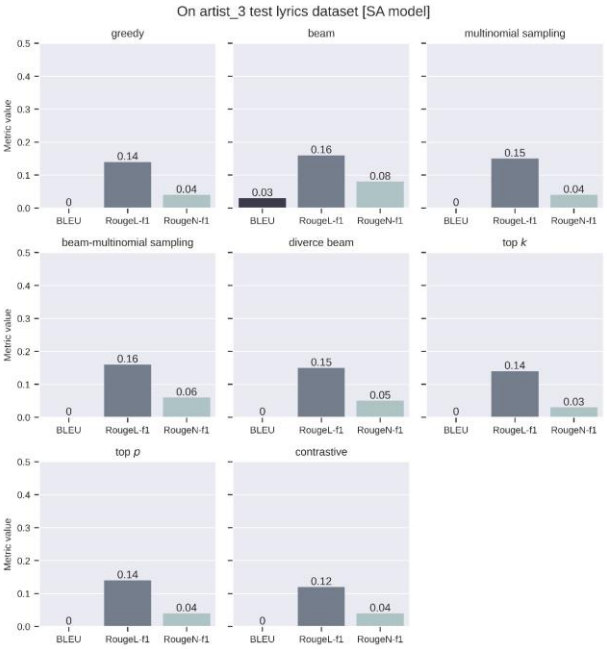

Рисунок 18 – Значення метрик для різних стратегій генерації на даних артиста-3 (модель SA)

З рис. 18, на якому виведено результати метрик для текстів автора з найменшою кількістю екземплярів у датасетах, видно суттєве зниження значень метрики, зокрема для BLEU.

Лише променевий пошук має ненульове значення цієї метрики. При цьому, Rouge метрики показують незначне падіння. Такі результати пов'язані як з кількістю тестових даних, так і з природою текстів (їх унікальністю і різноманітністю). Останні результати, що розглянуто в роботі, продемонстровані на рис. 19, показують, що узагальнене вміння генерувати тексти моделлю NSA є кращим (по значенням метрик) за аналогічне в SA моделі. Знову, лише променевий погук отримав ненульове значення для BLEU. Аналогічно до результатів інших авторів, модель NSA отримала кращі результати за SA. Такі результати можуть вказувати на кілька важливих аспектів, по-перше, необхідно розширювати набір даних, збільшуючи його різноманітність, покращуючи якість. При цьому, збільшення кількості разів оновлення параметрів моделі можуть призвести до того, що модель SA зможе краще запам'ятовувати та відтворювати тексти конкретних авторів.

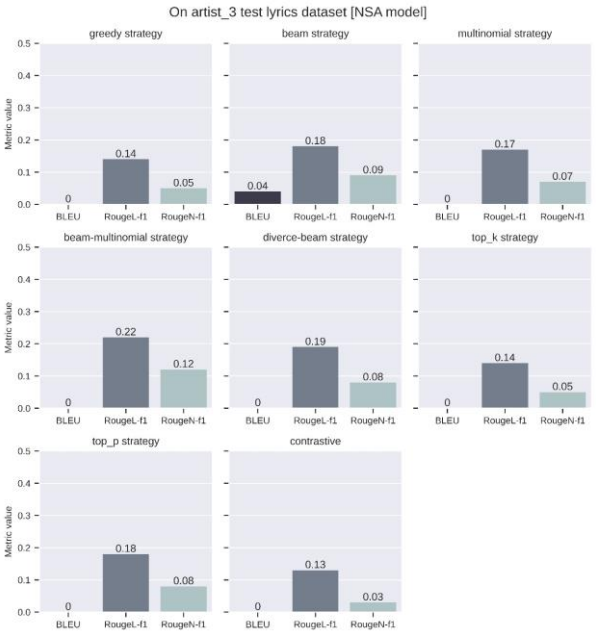

Рисунок 19 – Значення метрик для різних стратегій генерації на даних артиста-1 (модель NSA)

З усіх результатів порівняння текстів видно, що метрично найкращим методом генерації текстів пісень є beam пошук та його варіації, зокрема променевий семплінг. Контрастивний пошук зазвичай перевершував звичайний жадібний підхід. Методи top-*p* та top-*k* не мають однозначної переваги один на одним, і в різних ситуаціях давали різні результати.

Враховуючи результати, отримані при проведенні експериментів, прийнято рішення додатково дослідити кілька алгоритмів декодування на залежність значень метрик від параметрів цих алгоритмів. Для прикладу обрано дві стратегії, одну стохастичну – променевий семплінг, та одну детерміністичну – контрастивний пошук. Проблема дослідження променевого пошуку полягає у природній випадковості процесу генерації, що може суттєво впливати на отримані результати генерації. Щоб нівелювати цю проблему при проведені даного алгоритму зафіксовано random seed. Досліджуваними параметрами обрано кількість променів та температуру функції softmax. Метрика оцінювання – BLEU, множина тестового датасету – від Артиста 2.

В результаті повторної генерації та оцінювання, побудовано теплову діаграму залежності значення BLEU метрики від комбінації двох визначених параметрів для тестування NSA, виведену на рис. 20.

DOI 10.15588/1607-3274-2023-4-15

Як видно з рис. 20, збільшення температури, а відповідно згладжування розподілу (тобто створення умов для вибору менш ймовірних слів) погіршує результат, адже зростає різномастість тексту. Аналогічно, при збільшення кількості променів зменшується і значення метрики, проте цей результат може змінитися від наступного процесу генерації.

Аналогічне дослідження проведено для контрастивного пошуку. Проте його детерміністичність дозволяє уникнути проблеми різних значень при повторних генераціях. Параметрами для зіставного пошуку обрано ті, що описані в Розділі 4, а саме: k та α штрафу. Відповідні значення обчислень виведені на рис. 21.

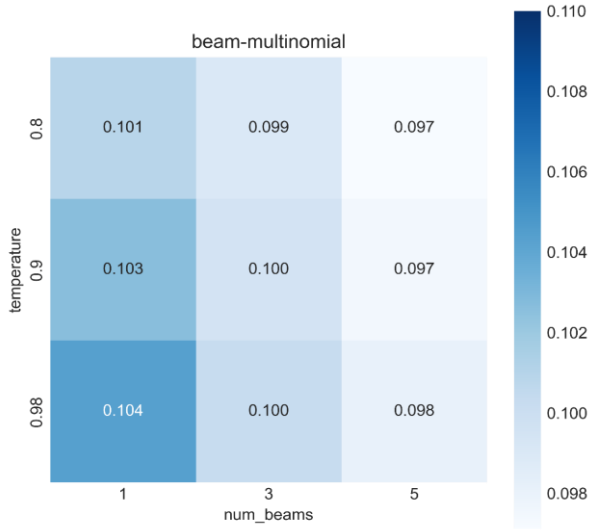

Рисунок 20 – Теплова карта BLEU метрики залежно від комбінації параметрів променевого семплінгу

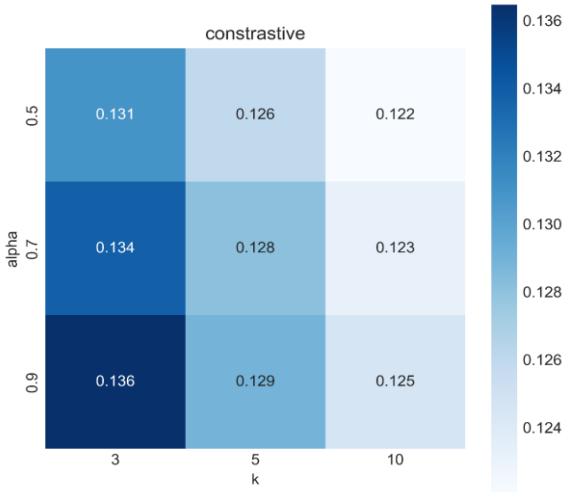

Рисунок 21 – Теплова карта BLEU метрики залежно від комбінації параметрів контрактивного пошуку

З рис. 21 видно, що зменшення штрафу призводить до зменшення значення метрики, що може бути наслідком генерування повторювальних рядків. При цьому збільшення кількості елементів, що враховуються в алгоритмі декодування, теж призводить до спаду значення метрики. Така тенденція зменшення може бути викликана ростом різноманітності та дискримінативності згенерованих стрічок, що збільшує різницю між бажаним результатом. В загальному випадку, не існує кращої стратегії декодування, а її вибір залежить від конкретних умов донавчання, наявного набору даних, можливостей моделі, задачі, та поставлених цілей процесу генерації.

В контексті розробленої ІС дві характеристики є найбільш важливими для аналізу. Перша – оцінка кількості викидів та впливу на екологію планети через використання та навчання моделей на апаратних прискорювачах. Для оцінки цієї характеристики можна використовувати ідею з [8].

Так, на момент виконання роботи, враховуючи експерименти та використання моделей, навчання двох T5 моделей на прискорювачі T4 в середовищі Google Colab, усереднено викинуто майже 0,55 кг карбон діоксиду (вуглекислого газу). Друга характеристика напряму пов'язана з першою – це час виконання генерації та декодування тексту з використанням різних стратегій. Для аналізу цього часу, при виконання генерації прикладів, зафіксовано час обчислення та декодування тексту. Звичайно, що цей час залежить від розміру отриманої стрічки, інших процесів ІС тощо. Тому, перед візуалізацією, отриманий час нормалізовано за максимальним значенням. На рис. 22 виведено умовні витрати часу для генерації результатів моделлю NSA, а на рис. 23 – моделлю SA на малих за обсягом вхідних даних.

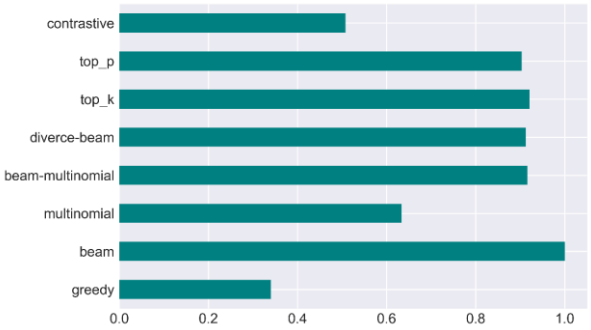

Рисунок 22 – Нормалізований час виконання генерації моделлю NSA

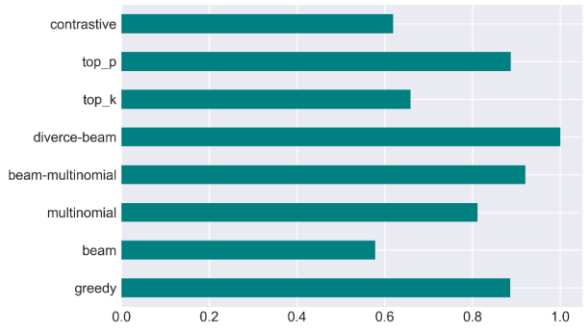

Рисунок 23 – Нормалізований час виконання генерації моделлю SA

DOI 10.15588/1607-3274-2023-4-15

В обох випадках найдовшими за часом виконання є варіації променевого пошуку, а контрастивний пошук – один з найкоротших.

ВИСНОВКИ

При формулюванні контексту даної роботи визначено основну його мету – створення ІС генерації продовження текстів пісень з та баз врахування стилю автора. У результаті виконання цієї роботи цю мету досягнуто, оскільки:

– Проведено системний аналіз ПО. Виявлено основні поняття, процеси та необхідні терміни, що дозволяють створити та реалізувати описану ІС;

– Виконано задачі створення текстового набору даних пар вхідні-вихідні стрічки пісень;

– Проведено fine-tuning двох моделей-трансформерів архітектури T5 для розв'язання задачі штучної генерації тексту проводження пісні з та без врахування стилю автора;

– Проаналізовано вплив процесу навчання на навколишнє середовище, оцінено час виконання процесу генерації, залежно від стратегії декодування;

– Наведено приклади генерації продовження пісень, проаналізовано отриманий текст;

– Кількісно оцінено якість генерації текстів створеними моделями з допомогою метрик порівняння стрічок.

В результаті стаття описує ІТ генерації продовження текстів пісень з допомогою великих мовних моделей, зокрема моделі T5, для прискорення, доповнення та підвищення гнучкості процесу написання текстів до пісень з/без врахування стиля певного автора. Для створення даних відібрано 10 різних артистів, після чого відібрано тексти їх пісень. Загалом отримано 626 унікальних пісень. В результаті розбиття кожної пісні на кілька пар вхідних-вихідних стрічок отримано 1874 навчальних екземплярів та 465 тестових. Проведено донавчання двох мовних моделей NSA та SA для задачі генерації продовження тексту пісень. Для обох моделей як базову обрано t5-base. Ця версія T5 містить 223 мільйони параметрів. Аналіз вихідних даних показав, що модель NSA має менш деградаційні результати, а для моделі SA необхідно збалансовувати кількість тексту для кожного автора. Обраховано кілька текстових метрик як BLEU, RougeL та RougeN для кількісного порівняння результатів моделей та стратегій генерування. Значення метрики BLEU є найрізноманітнішим, і його значення значно змінюється, залежно від стратегії. При цьому Rouge метрики мають меншу варіативність, менший розмах значень. Загалом, для порівняння, у роботі використано 8 різних методик декодування для генерації тексту, що підтримуються бібліотеку transformers, зокрема Contrastive search, Top-p sampling, Top-k sampling, Multinomial sampling, Beam search, Diverse beam search, Greedy search, та Beam-search multinomial sampling. З усіх результатів порівняння текстів видно, що метрично найкращим методом генерації текстів пісень є beam пошук та його варіації, зокрема променевий семплінг. Контрастивний пошук зазвичай перевершував звичайний жадібний підхід. Методи top-p та top-k не мають однозначної переваги один на одним, і в різних ситуаціях давали різні результати.

Логічно, що утворена ІС має шляхи розвитку та вдосконалення. Основними перспектива для покращення ІС є такі, що стосуються даних навчання, моделей трансформерів та безпосередньо процесу розробки. Тому можна виділити такі напрямки майбутніх досліджень:

– Використання більш нових чи потужних моделей ніж класична T5;

– Розширення кількості та різноманітності навчальних даних. Заміна або поєднання кількох видів задачі генерації текстів пісень;

– Розширення групи мов, що підтримуються ІС;

– Включення інших структурних елементів пісні (ритм, акорди, розмірність віршового рядка) або музичний супровід відповідної пісні як вхідну інформацію для генерації текстів;

– Вдосконалення процесу навчання моделей, наприклад застосуючи контрастивний процес навчання або refinement, тобто залучення справжніх авторів для оцінки якості та покращення результатів моделей.

Основні перспективи проведення майбутніх досліджень, для поглиблення та вдосконалення аналізу стратегій генерації текстів пісень є:

– Виявлення, створення та оцінка нових чи інших методів декодування тексту для генерації тексту, що може включати і комбінування вже розглянутих алгоритмів;

– Перехід до інших класів мовних моделей-трансформерів (BERT, GPT, Bart, T5v1.1 тощо), та порівняльний аналіз стратегій для цих моделей;

– Огляд або розробка інших метрик, що можуть включати порівняння семантичного чи синонімічного наповнення тексту, риму, чи збереження розміру віршованих текстів.

ЛІТЕРАТУРА

1. Attention Is All You Need / [A. Vaswani, N. Shazeer, N. Parmar et al.] // arXiv. – Access mode: https://arxiv.org/abs/1706.03762
2. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer / [C. Raffel, N. Shazeer, A. Roberts et al.] // arXiv. – Access mode: https://arxiv.org/abs/1910.10683
3. Hugging Face community. Text generation strategies. – Access mode: https://huggingface.co/docs/transformers/v4.29.0/en/generation_strategies
4. von Platen P. How to generate text: using different decoding methods for language generation with Transformers / P. von Platen. – Access mode: https://huggingface.co/blog/how-to-generate
5. Hugging Face community. T5. – Access mode: https://huggingface.co/docs/transformers/model_doc/t5

DOI 10.15588/1607-3274-2023-4-15

6. Diverse Beam Search: Decoding Diverse Solutions from Neural Sequence Models / [A. K. Vijayakumar, M. Cogswell, R. R. Selvaraju et al.] // arXiv. – Access mode: https://arxiv.org/abs/1610.02424
7. Hugging Face community. T5v1.1. – Access mode: https://huggingface.co/docs/transformers/model_doc/t5v1.1
8. Quantifying the Carbon Emissions of Machine Learning / [A. Lacoste, A. Luccioni, V. Schmidt, T. Dandres] // arXiv. – Access mode: https://arxiv.org/abs/1910.09700
9. Google. SentencePiece. – Access mode: https://github.com/google/sentencepiece
10. Sennrich R. Subword Neural Machine Translation / R. Sennrich. – Access mode: https://github.com/rsennrich/subword-nmt
11. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation / [Y. Wu, M. Schuster, Z. Chen et al.] // arXiv. – Access mode: https://arxiv.org/abs/1609.08144
12. Hugging Face community. Transformers. State-of-the-art Machine Learning for PyTorch, TensorFlow, and JAX. – Access mode: https://huggingface.co/docs/transformers/index
13. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems / [M. Abadi, A. Agarwal, P. Barham et al.] // arXiv. – Access mode: https://arxiv.org/abs/1603.04467
14. Shah D. Song Lyrics Dataset / D. Shah // Kaggle. – Access mode: https://www.kaggle.com/datasets/deepshah16/song-lyrics-dataset
15. Swift T. Long story short / T. Swift, A. Dessner // Genius. Taylor Swift Music. – Access mode: https://genius.com/Taylor-swift-long-story-short-lyrics
16. Fan A. Hierarchical Neural Story Generation / A. Fan, M. Lewis, Y. Dauphin // Association for Computational Linguistics : 56th Annual Meeting, Melbourne, Australia, July 2018 : proceedings. – Melbourne: ACL, 2018. – P. 889–898. DOI: 10.18653/v1/p18-1082
17. Chiang T.-R. Relating Neural Text Degeneration to Exposure Bias / T.-R. Chiang, Y.-N. Chen // Analyzing and Interpreting Neural Networks for NLP : the Fourth BlackboxNLP Workshop, Punta Cana, Dominican Republic, November 2021 : proceedings. – Punta Cana: ACL, 2021. – P. 228–239. DOI: 10.18653/v1/2021.blackboxnlp-1.16
18. A Contrastive Framework for Neural Text Generation / [Y. Su, T. Lan, Y. Wang et al.] // arXiv. – Access mode: https://arxiv.org/abs/2202.06417
19. Paulus R. A Deep Reinforced Model for Abstractive Summarization / R. Paulus, C. Xiong, R. Socher // arXiv. – Access mode: https://arxiv.org/abs/1705.04304
20. OpenNMT: Open-Source Toolkit for Neural Machine Translation / [G. Klein, Y. Kim, Y. Deng et al.] // System Demonstrations : Association for Computational Linguistics, Vancouver, Canada, July 2017 : proceedings. – Vancouver: ACL, 2017. – P. 67–72. DOI: 10.18653/v1/p17-4012
21. Murray K. Correcting Length Bias in Neural Machine Translation / K. Murray, D. Chiang // arXiv. – Access mode: https://arxiv.org/abs/1808.10006
22. Mathur N. Tangled up in BLEU: Reevaluating the Evaluation of Automatic Machine Translation Evaluation Metrics / N. Mathur, T. Baldwin, T. Cohn // Association for Computational Linguistics : 58th Annual Meeting, Online, July 2020 : proceedings. – Online: ACL, 2020. – P. 4984–4997. DOI: 10.18653/v1/2020.acl-main.448
23. Lin C.-Y. ROUGE: A Package for Automatic Evaluation of Summaries / C.-Y. Lin // Text Summarization Branches Out : Association for Computational Linguistics, Barcelona, Spain, July 2004 : proceedings. – Barcelona: ACL, 2004. – P. 74–81. Access mode: https://aclanthology.org/W04-1013
24. KerasNLP. Access mode: https://keras.io/keras_nlp/
25. Prokipchuk O. Ukrainian Language Tweets Analysis Technology for Public Opinion Dynamics Change Prediction Based on Machine Learning / O. Prokipchuk, V. Vysotska // Radio Electronics, Computer Science, Control. – 2023. – No. 2 (65). – P. 103–116. DOI: 10.15588/1607-3274-2023-2-11

Стаття надійшла до редакції 15.09.2023.
Після доробки 25.10.2023.

UDC 004.9

SONGS CONTINUATION GENERATION TECHNOLOGY BASED ON TEST GENERATION STRATEGIES, TEXTMINING AND LANGUAGE MODEL T5

Mediakov O. – Post-graduate student of Information Systems and Networks Department, Lviv Polytechnic National University, Lviv, Ukraine.

Vysotska V. – PhD, Associate Professor of Information Systems and Networks Department, Lviv Polytechnic National University, Lviv, Ukraine.

ABSTRACT

Context. Pre-trained large language models are currently the driving force behind the development of not only NLP, but also deep learning systems in general. Model transformers are able to solve virtually all problems that currently exist, provided that certain requirements and training practices are met. In turn, words, sentences and texts are the basic and most important way of communication between intellectually developed beings. Of course, speech and texts are used to convey certain emotions, events, etc. One of the main ways of using language to describe experienced emotions is songs with lyrics. However, often due to the need to preserve rhyme and rhyming, the dimensions of verse lines, song structure, etc., artists have to use repetition of lines in the lyrics. In addition, the process of writing texts can be long.

Objective of the study is to develop information technology for generating the continuation of song texts based on the T5 machine learning model with (SA, specific author) and without (NSA, non-specific author) consideration of the author's style.

Method. Choosing a decoding strategy is important for the generation process. However, instead of favoring a particular strategy, the system will support multiple strategies. In particular, the following 8 strategies: Contrastive search, Top-p sampling, Top-k sampling, Multinomial sampling, Beam search, Diverse beam search, Greedy search, and Beam-search multinomial sampling.

Results. A machine learning model was developed to generate the continuation of song lyrics using large language models, in particular the T5 model, to accelerate, complement and increase the flexibility of the songwriting process.

DOI 10.15588/1607-3274-2023-4-15

Conclusions. The created model shows excellent results of generating the continuation of song texts on test data. Analysis of the raw data showed that the NSA model has less degrading results, while the SA model needs to balance the amount of text for each author. Several text metrics such as BLEU, RougeL and RougeN are calculated to quantitatively compare the results of the models and generation strategies. The value of the BLEU metric is the most variable, and its value varies significantly depending on the strategy. At the same time, Rouge metrics have less variability, a smaller range of values. For comparison, 8 different decoding methods for text generation, supported by the transformers library, were used. From all the results of the text comparison, it is clear that the metrically best method of song text generation is beam search and its variations, in particular beam sampling. Contrastive search usually outperformed the conventional greedy approach. The top-p and top-k methods are not clearly superior to each other, and in different situations gave different results.

KEYWORDS: text generation, T5 language model, Transformers, author's style, Contrastive search, Top-p sampling, Top-k sampling, Multinomial sampling, Beam search, Diverse beam search, Greedy search, and Beam-search multinomial sampling.

REFERENCES

1. Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser Ł., Polosukhin I. Attention Is All You Need, *arXiv*. Access mode: https://arxiv.org/abs/1706.03762
2. Raffel C., Shazeer N., Roberts A., Lee K., Narang S., Matena M., Zhou Y., Li W., Liu P. J. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, *arXiv*. Access mode: https://arxiv.org/abs/1910.10683
3. Hugging Face community. Text generation strategies. Access mode: https://huggingface.co/docs/transformers/v4.29.0/en/generation_strategies
4. von Platen P. How to generate text: using different decoding methods for language generation with Transformers. Access mode: https://huggingface.co/blog/how-to-generate
5. Hugging Face community. T5. Access mode: https://huggingface.co/docs/transformers/model_doc/t5
6. Vijayakumar A. K., Cogswell M., Selvaraju R. R., Sun Q., Lee S., Crandall D., Batra D. Diverse Beam Search: Decoding Diverse Solutions from Neural Sequence Models, *arXiv*. Access mode: https://arxiv.org/abs/1610.02424
7. Hugging Face community. T5v1.1. Access mode: https://huggingface.co/docs/transformers/model_doc/t5v1.1
8. Lacoste A., Luccioni A., Schmidt V., Dandres T. Quantifying the Carbon Emissions of Machine Learning, *arXiv*. Access mode: https://arxiv.org/abs/1910.09700
9. Google. SentencePiece. Access mode: https://github.com/google/sentencepiece
10. Sennrich R. Subword Neural Machine Translation. Access mode: https://github.com/rsennrich/subword-nmt
11. Wu Y., Schuster M., Chen Z., Le Q. V., Norouzi M., Macherey W., Krikun M., Cao Y., Gao Q., Macherey K., Klingner J., Shah A., Johnson M., Liu X., Kaiser Ł., Gouws S., Kato Y., Kudo T., Kazawa H., Stevens K., Kurian G., Patil N., Wang W. , Young C., Smith J., Riesa J., Rudnick A., Vinyals O. , Corrado G., Hughes M., Dean J. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation, *arXiv*. Access mode: https://arxiv.org/abs/1609.08144
12. Hugging Face community. Transformers. State-of-the-art Machine Learning for PyTorch, TensorFlow, and JAX. Access mode: https://huggingface.co/docs/transformers/index
13. Abadi M., Agarwal A., Barham P., Brevdo E., Chen Z., Citro C., Corrado G. S., Davis A. , Dean J., Devin M., Ghemawat S., Goodfellow I., Harp A., Irving G., Isard M., Jia Y., Jozefowicz R., Kaiser L., Kudlur M., Levenberg J., Mane D., Monga R., Moore S., Murray D., Olah C., Schuster M., Shlens J., Steiner B., Sutskever I., Talwar K., Tucker P., Vanhoucke V., Vasudevan V., Viegas F., Vinyals O., Warden P., Wattenberg M., Wicke M., Yu Y., Zheng X. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, *arXiv*. Access mode: https://arxiv.org/abs/1603.04467
14. Shah D. Song Lyrics Dataset, *Kaggle*. Access mode: https://www.kaggle.com/datasets/deepshah16/song-lyrics-dataset
15. Swift T., Dessner A. Long story short, *Genius. Taylor Swift Music*. Access mode: https://genius.com/Taylor-swift-long-story-short-lyrics
16. Fan A., Lewis M., Dauphin Y. Hierarchical Neural Story Generation, *Association for Computational Linguistics : 56th Annual Meeting, Melbourne, Australia, July 2018 : proceedings*. Melbourne, ACL, 2018, pp. 889–898. DOI: 10.18653/v1/p18-1082
17. Chiang T.-R., Chen Y.-N. Relating Neural Text Degeneration to Exposure Bias, *Analyzing and Interpreting Neural Networks for NLP : the Fourth BlackboxNLP Workshop, Punta Cana, Dominican Republic, November 2021 : proceedings*. Punta Cana, ACL, 2021, pp. 228–239. DOI: 10.18653/v1/2021.blackboxnlp-1.16
18. Su Y., Lan T., Wang Y., Yogatama D., Kong L., Collier N. A Contrastive Framework for Neural Text Generation, *arXiv*. Access mode: https://arxiv.org/abs/2202.06417
19. Paulus R., Xiong C., Socher R. A Deep Reinforced Model for Abstractive Summarization, *arXiv*. Access mode: https://arxiv.org/abs/1705.04304
20. Klein G., Kim Y., Deng Y., Senellart J., Rush A. OpenNMT: Open-Source Toolkit for Neural Machine Translation, *System Demonstrations : Association for Computational Linguistics, Vancouver, Canada, July 2017 : proceedings*. Vancouver, ACL, 2017, pp. 67–72. DOI: 10.18653/v1/p17-4012
21. Murray K., Chiang D. Correcting Length Bias in Neural Machine Translation, *arXiv*. Access mode: https://arxiv.org/abs/1808.10006
22. Mathur N., Baldwin T., Cohn T. Tangled up in BLEU: Reevaluating the Evaluation of Automatic Machine Translation Evaluation Metrics, *Association for Computational Linguistics : 58th Annual Meeting, Online, July 2020 : proceedings*. Online, ACL, 2020, pp. 4984–4997. DOI: 10.18653/v1/2020.acl-main.448
23. Lin C.-Y. ROUGE: A Package for Automatic Evaluation of Summaries, *Text Summarization Branches Out : Association for Computational Linguistics, Barcelona, Spain, July 2004 : proceedings*. Barcelona, ACL, 2004, pp. 74–81. Access mode: https://aclanthology.org/W04-1013
24. KerasNLP. Access mode: https://keras.io/keras_nlp/
25. Prokipchuk O., Vysotska V. Ukrainian Language Tweets Analysis Technology for Public Opinion Dynamics Change Prediction Based on Machine Learning, *Radio Electronics, Computer Science, Control*, 2023, No. 2(63), pp. 103–116. DOI: 10.15588/1607-3274-2023-2-11

DOI 10.15588/1607-3274-2023-4-15