ДОДАТОК А

Результати роботи моделі на власному наборі даних

Таблиця А.1 – Результати роботи моделі на власному наборі даних

№	Назва файлу	Людський опис зображення	Згенеровані підписи	Оцінка
1	kharkiv_center.jpg	Річка з мостами, багато будівель, церква.	1) «a train traveling over a bridge over a river .» (p=0.000941); 2) «a train traveling over a bridge over a body of water .» (p=0.000094); 3) «a large body of water with a bridge spanning it .» (p=0.000022).	2
2	independence_monument.jpg	Пам'ятник у вигляді дівчини з крилами та вінком у руках, що стоїть зверху на земній кулі; на задньому плані церква.	1) «a building with a clock on the top of it» (p=0.000155); 2) «a building with a clock on top of it .» (p=0.000118); 3) «a statue of a man standing in front of a clock tower .» (p=0.000096).	2
3	rider_monument.jpg	Пам'ятник вершнику, що сидить на коні, на фоні будівель.	1) «a statue of a man riding a horse in front of a building .» (p=0.001347);	5

Продовження таблиці А.1

№	Назва файлу	Людський опис зображення	Згенеровані підписи	Оцінка
3	rider_monument.jpg	Пам'ятник вершнику на коні на фоні будівель.	2) «a statue of a man on a horse in front of a building .» (p=0.000552); 3) «a statue of a man standing on top of a building .» (p=0.000527).	5
4	park_gorkogo.jpg	Парк з колесом огляду, на передньому плані люди.	1) «a group of people standing in a field flying kites .» (p=0.000644); 2) «a group of people standing around a large building .» (p=0.000156); 3) a group of people standing around a large kite . (p=0.000139).	3
5	nure.jpg	Велика будівля, на передньому плані дерева.	1) «a tall building with a clock on top of it .» (p=0.000336); 2) «a large building with a clock on the top of it .» (p=0.000142); 3) «a large building with a clock on the top of it» (p=0.000127).	3

Продовження таблиці А.1

№	Назва файлу	Людський опис	Згенеровані підписи	Оцінка
6	soldier_monument.jpg	Пам'ятник солдату, що високо підійняв рушницю, на тлі високого житлового будинку.	1) «a statue of a man riding a horse in front of a building .» (p=0.000330); 2) «a statue of a man on a horse in the middle of a street .» (p=0.000069); 3) «a statue of a man on a horse in the middle of the street .» (p=0.000059).	2
7	students.jpg	Студенти сидять у лекційній аудиторії.	1) «a group of people sitting around a table .» (p=0.002029); 2) «a group of people sitting around a table with laptops .» (p=0.000317); 3) «a group of people sitting around a table with a laptop .» (p=0.000053).	3
8	lecture.jpg	Лекційна аудиторія з дошкою та трансляцією на екрані.	1) «a man standing in front of a tv holding a game controller .» (p=0.000110); 2) «a man standing in front of a tv holding a wii controller .» (p=0.000082); 3) «a man standing in front of a tv playing a video game .»	2

Продовження таблиці А.1

№	Назва файлу	Людський опис	Згенеровані підписи	Оцінка
9	laboratory.jpg	Столи з комп'ютерами та фотографії на стінах.	1) «a desk with a laptop and a desktop computer .» (p=0.000104); 2) «a desk with a laptop and a monitor» (p=0.000101); 3) «a desk with a laptop and a desktop computer on it .» (p=0.000034).	4
10	zoo_elephants.jpeg	Два слони у загоні на фоні забору.	1) «a group of elephants standing next to each other .» (p=0.005011); 2) «a herd of elephants walking across a dirt field .» (p=0.002721); 3) «a group of elephants standing next to each other in a field .» (p=0.000161).	5
11	zoo_bear.jpg	Ведмідь лежить біля дерев'яної колоди на фоні каміння.	1) «a large brown bear laying on top of a rock .» (p=0.001775); 2) «a large brown bear standing on top of a rock .» (p=0.001311); 3) «a large brown bear laying on top of a pile of rocks .» (p=0.000371).	5

Продовження таблиці А.1

№	Назва файлу	Людський опис	Згенеровані підписи	Оцінка
12	sunflowers.jpeg	Поле з соняшниками на фоні блакитного неба.	1) «a bunch of green and yellow flowers in a field .» (p=0.000060); 2) «a bunch of green and yellow flowers in a field» (p=0.000022); 3) «a bunch of green and white flowers in a field .» (p=0.000017).	5
13	borshch.jpg	Тарілка борщу, поряд з якою зелень, пампушки та ще кілька мисок з їжею.	1) «a bowl of soup and a bowl of soup .» (p=0.000841); 2) «a bowl of soup and a bowl of soup on a table .» (p=0.000436); 3) «a bowl of soup and a plate of food» (p=0.000119).	4
14	girl.jpg	Дівчина у вінку тримає за спиною прапор.	1) «a woman holding an umbrella in her hand .» (p=0.000336); 2) «a woman holding an umbrella in her hands .» (p=0.000274); 3) «a woman holding an umbrella in her hand» (p=0.000082).	3

Продовження таблиці А.1

№	Назва файлу	Людський опис	Згенеровані підписи	Оцінка
15	school.jpg	Ряд шкільних парт, за якими по двоє сидять школярі.	1) «a young boy sitting at a table with a laptop .» (p=0.000042); 2) «a little girl sitting at a table with a baby .» (p=0.000039); 3) «a little girl sitting at a table with a child .» (p=0.000038).	3
16	kashtan.jpg:	На передньому плані квітнуть каштани, на задньому – небо і церква.	1) «a clock tower in the middle of a park .» (p=0.000182); 2) «a clock tower in the middle of a city .» (p=0.000177); 3) «a tall clock tower with a sky background» (p=0.000162).	3
17	football.jpg	Троє хлопців грають у футбол на полі.	1) «a group of young men playing a game of soccer .» (p=0.017758); 2) «a group of men playing a game of soccer .» (p=0.008334); 3) «a group of young men playing a game of frisbee .» (p=0.003716).	5
18	crimea.jpg	Вежа на фоні моря та гір, у морі плаває катер.	1) «a large body of water with a boat in it .» (p=0.000145);	3

Продовження таблиці А.1

№	Назва файлу	Людський опис	Згенеровані підписи	Оцінка
18	crimea.jpg	Вежа на фоні моря та гір, у морі катер.	2) «a large body of water with a boat in the background .» (p=0.000096); 3) «a large body of water with a boat in the water .» (p=0.000094).	3
19	musicians.jpg	Чоловік у білій футболці тримає гітару та стоїть біля чоловіка у чорній сорочці.	1) «a man holding a tennis racquet on a tennis court .» (p=0.000740); 2) «a man holding a tennis racquet on a court .» (p=0.000441); 3) «a man in a white shirt and tie holding a tennis racket .» (p=0.000062).	2
20	cat.jpg	Лежить кішка, у якої на шиї пов'язані стрічки блакитно-жовтого кольору.	1) «a cat that is laying down on a blanket .» (p=0.000378); 2) «a cat that is laying down on a blanket» (p=0.000145); 3) «a cat that is laying down with a toothbrush .» (p=0.000135).	5

ДОДАТОК Б

Слайди презентації

Дослідження методів генерації опису зображення

Виконала:

ст. гр. ІПЗм-17-1 Кальметьєва М.К.

Керівник роботи:

к.т.н. доцент Турута О.П.

Мета роботи

Аналіз існуючих методів генерації опису зображення.

Розробка моделі генерації підписів до зображення з додатковим налаштуванням моделі для підвищення якості її роботи.

2

Актуальність. Варіанти використання:

- допомога людям з вадами зору для розуміння сутності зображення
- пояснення подій, що відбуваються на відео, кадр за кадром
- «розумний» пошук зображень за змістом (content-based image retrieval)
- надання альтернативного тексту для зображень у частинах світу, де мобільні з'єднання є повільними
- джерело інформації для соціальних платформ

3

Наявні датасети та метрики

• UIUC Pascal Sentences: 10 тис. • FLICKR 8K: 8 тис. • FLICKR 30K: 30 тис. • MS COCO: 120 тис. • Conceptual Captions (2018): 3.3 мільйони пар	• BLEU: аналог presicion • ROUGE: аналог recall • METEOR: аналог F1-score з надбудовами (стемінг) • CIDEr та CIDEr-D: косинус подібності • SPICE: семантичний граф

4

Наявні сервіси

Сервіс	Переваги	Недоліки
Microsoft Cognitive Services	• Зручно використовувати в рамках Azure • Імовірно, якість висока	• $2.5 за 1000 • Конкретна точність невідома
Microsoft Caption Bot	• Можна швидко перевірити якість на окремих прикладах	• Не вдасться використати як складову своєї програми
IBM Model Asset eXchange (MAX)	• Містить надбудови, що дозволяють зручно розгортати модель • Можна вносити деякі зміни у реалізацію	• Реалізує класичну модель 2014 року

5

Загальні методи генерації опису до зображення

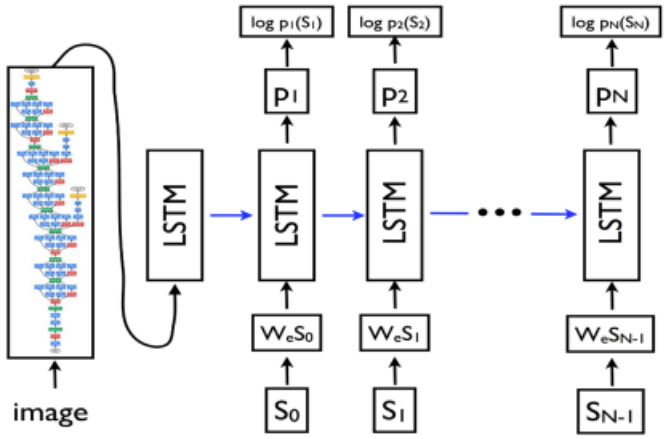
- Створення підпису на основі шаблонів.
- Опис на основі пошуку.
- Використання підходів глибокого навчання.

"Show And Tell: A Neural Image Caption Generator", 2014

Approach	PASCAL (xfer)	Flickr 30k	Flickr 8k
Im2Text [24]			
TreeTalk [18]			
BabyTalk [16]	25		
Tri5Sem [11]			48
m-RNN [21]		55	58
MNLM [14][5]		56	51
SOTA	25	56	58
NIC	**59**	**66**	**63**
Human	69	68	70

6

Модель «Show And Tell»: CNN + RNN



7

Задача оптимізації

Мета: максимізувати функцію вірогідності формування правильного підпису.

$$\theta^* = \arg max_\theta \sum_{(I,S)} \log p(S|I;\theta)$$

$$\log p(S|I;\theta) = \sum_{t=0}^{N} \log p(S_t|I, S_0, \dots, S_{t-1})$$

Процедура навчання моделі:

$$x_{-1} = \mathrm{CNN}(I)$$

$$x_t = W_e S_t, \ t \in \{0 \dots N-1\}$$

$$p_{t+1} = \mathrm{LSTM}(x_t), \ t \in \{0 \dots N-1\}$$

8

Розробка моделі генерації підпису до зображення

Основа:

- «Show and Tell»

Можливе додаткове налаштування:

- Вибір архітектури CNN
- Режими використання Transfer learning
- Використання аугментацій

Датасет: MSCOCO

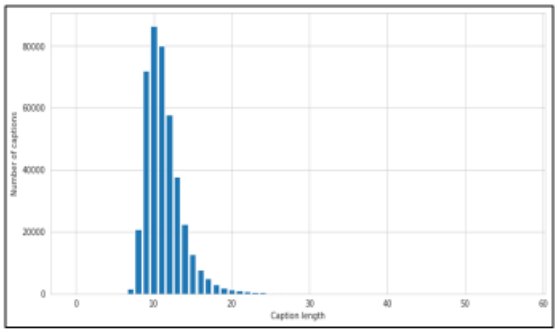
Метрика: CIDEr-D та BLEU

9

Огляд MS COCO

Train size	Validation size	Test size
82783	40504	40775

- Розмір зображень не перевищує 640x640.
- По 5 підписів до кожного зображення.



10

Основні гіперпараметри

Назва гіперпараметру	Значення гіперпараметру
Темп навчання (learning rate)	0.0005
Розмір батча (batch size)	32
Розмір прихованого шару RNN	512
Розмірність векторного простору (embedding) слів та зображень	256
Максимальна допустима довжина згенерованого підпису	20
Ширина пучка променевого пошуку	3
Максимальна кількість епох	50

Функція втрат: cross-entropy

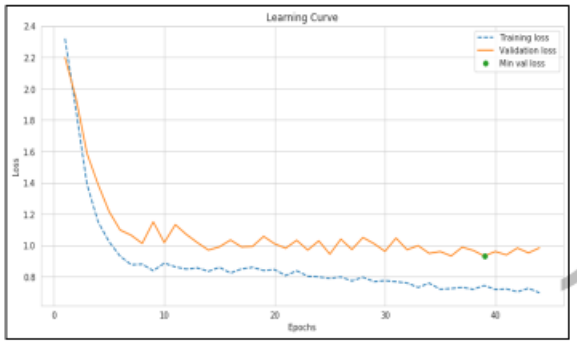
Оптимізатор SGD: Adam

11

Вибір архітектури згорткової частини

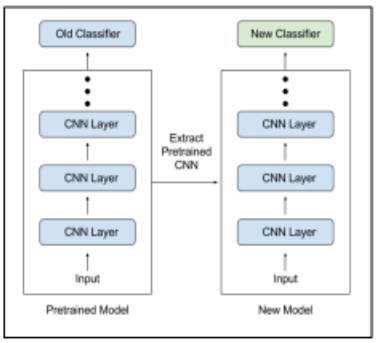
Архітектура CNN	CIDEr-D	BLEU-1	BLEU-4	Кількість епох
InceptionV4, 2016	0.883	70.5	28.9	44
Xception, 2016	0.870	69.9	27.8	41
NASNet Large, 2017	0.885	70.7	29.2	30 (тренування припинене)

Крива навчання **InceptionV4**



12

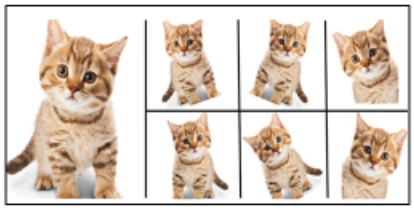
Режими розморозки моделі при переносі навчання

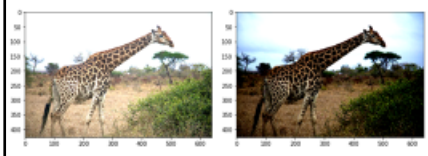


Назва моделі	Cider-D	BLEU-1	BLEU-4	Кількість епох
InceptionV4 з класичним переносом навчання	0.883	70.5	28.9	44
InceptionV4 з відстроченою розморозкою зготкової частини при переносі навчання	0.887	70.9	29.2	43

13

Використання аугментацій





Назва моделі	Cider-D	BLEU-1	BLEU-4	Кількість епох
InceptionV4	0.887	70.9	29.2	43
InceptionV4 з м'якими аугментаціями	0.904	71.6	30.6	46
InceptionV4 зі збільшеними аугментаціями кольору	0.883	70.7	29.1	46
InceptionV4 зі збільшеними просторовими аугментаціями	0.917	71.9	30.8	47

14

Зображення	Згенеровані підписи	Оцінка
	• «a statue of a man riding a horse in front of a building .» (p=0.001347); • «a statue of a man on a horse in front of a building .» (p=0.000552); • «a statue of a man standing on top of a building .» (p=0.000527).	5
	• «a bowl of soup and a bowl of soup .» (p=0.000841); • «a bowl of soup and a bowl of soup on a table .» (p=0.000436); • «a bowl of soup and a plate of food» (p=0.000119).	4
	• «a tall building with a clock on top of it .» (p=0.000336); • «a large building with a clock on the top of it .» (p=0.000142); • «a large building with a clock on the top of it» (p=0.000127).	3

15

Аналіз результатів

Модель	CIDEr-D	BLEU-1	BLEU-4
«Show and Tell», 2014	0.819	-	27.7
Xception	0.861	69.9	27.8
InceptionV4 зі збільшеними просторовими аугментаціями	0.917	71.9	30.8

Середня суб'єктивна оцінка на власному датасеті: 3.45

- Іноді генеруються хибні підписи через те, що модель намагається робити підписи довшими
- Люди в цілому описуються добре
- Зображення з великою кількістю деталей (наприклад, знімки міста зверху) описуються погано

Можливе подальше покращення якості шляхом управління темпом навчання чи використанням механізму «увага».

16

Висновки

- Проведено аналіз існуючих методів, метрик та датасетів для вирішення задачі генерації опису зображення
- Розроблена модель генерації підписів до зображення
- Описано проведені дослідження щодо додаткового налаштування моделі
- Проаналізовано результати досліджень
- Виявлено подальші можливості для розвитку

17

ДОДАТОК В

Наукові публікації

Міністерство освіти і науки України
Прикарпатський національний університет імені Василя Стефаника
Представництво "Польська академія наук" в Києві
Науково-технологічний університет "Гірничо-металургійна академія"
імені Станіслава Сташіца в Кракові
Вінницький національний технічний університет
Харківський національний університет радіоелектроніки
Національний авіаційний університет
Тернопільський національний економічний університет
Фінансово-економічний інститут Таджикистану
Економічна академія "Д.А.Ценов", Болгарія
Лудзький університет, Польща
Штуттгардський університет, Німеччина
Інститут інженерів з електротехніки та електроніки (IEEE), Українська секція
Громадська організація "Івано-Франківський IT кластер"

КОМП'ЮТЕРНІ НАУКИ, ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ ТА СИСТЕМИ УПРАВЛІННЯ

Матеріали
міжнародної науково-технічної конференції
молодих вчених, аспірантів та студентів

28–30 листопада 2018 року
Івано-Франківськ, Україна

COMPUTER SCIENCE, INFORMATION TECHNOLOGIES AND MANAGEMENT SYSTEMS

Proceedings
of the International Scientific Young Scientists Conference

2018, November , 28th to 30th
Ivano-Frankivsk, Ukraine

Івано-Франківськ
Прикарпатський національний університет імені Василя Стефаника
2018

Електронне видання комбінованого
використовування на CD-ROM

УДК 004+005

Наукові редактори: докт. техн. наук, проф. **Л.Б. Петришин** (ПНУ, AGH);
докт. техн. наук, проф. **П. Лебковський** (AGH)

Рецензенти:
д.т.н., проф. **І.Т. Когут**;
д.т.н. **Д. Саля**;
д.т.н., проф. **Заміховський Л.М.**

Комп'ютерні науки, інформаційні технології та системи управління : матеріали Міжнародної науково-технічної конференції студентів, аспірантів та молодих вчених, м. Івано-Франківськ, 28–30 листопада 2018 року / наук. ред. Л. Б. Петришин, П. Лебковський. – Електрон. дані. –Івано-Франківськ : Прикарпатський національний університет імені Василя Стефаника, 2018. – 1 електрон. опт. диск (CD-ROM); 12 см. – Назва з тит. екрана.
ISBN 978-966-640-448-3

Збірник містить матеріали статей Міжнародної науково-технічної конференції з проблем комп'ютерних наук, інформаційних технологій, систем управління та ігрового програмного забезпечення.

УДК 004+005

ISBN 978-966-640-448-3

Вибір Алгоритму Стемінгу Для Ефективного Повнотекстового Пошуку

Марина Кальметьева
Кафедра Програмної інженерії
Харківський національний університет радіоелектроніки
Харків, Україна
mkalmetieva@gmail.com

Володимир Кобзєв
Кафедра Програмної інженерії
Харківський національний університет радіоелектроніки
Харків, Україна
volodymyr.kobziev@nure.ua

Selecting a Stemming Algorithm for Effective Full-Text Search

Marina Kalmetieva
Software Engineering department
Kharkiv National University of Radioelectronics
Kharkiv, Ukraine
mkalmetieva@gmail.com

Volodymyr Kobziev
Software Engineering department
Kharkiv National University of Radioelectronics
Kharkiv, Ukraine
volodymyr.kobziev@nure.ua

***Анотація*—У роботі визначена роль стемінгу для повнотекстового пошуку та загальні проблеми при використанні стемінгу. Розглянуті різні алгоритми стемінгу для англійської мови, зокрема, наявні у бібліотеці Apache Lucene, виявлені особливості алгоритмів, їх переваги та недоліки. Сформовані принципи для вибору найбільш ефективного алгоритму стемінгу в залежності від наявних умов та вимог до програмної системи.**

***Ключові слова*—повнотекстовий пошук, стемінг.**

***Abstract*—The paper identifies the role of stemming for full text search and common problems with using stemming. Different algorithms of stemming for English language are considered, in particular, available in the library of Apache Lucene, features of algorithms, their advantages and disadvantages are revealed. The established principles for choosing the most effective stemming algorithm depending on the existing conditions and requirements for the software system.**

***Keywords*—full-text search, stemming.**

I. Вступ

На даний момент повнотекстовий пошук є надзвичайно популярним. Він активно використовується у різноманітних інформаційних системах, і можна з впевненістю очікувати, що його розповсюдженість буде ще зростати. Адже при роботі з великою кількістю даних набагато зручніше мати потужний, гнучкий та швидкий інструмент пошуку, схожий на Google, аніж обмежувати себе пошуком лише за точним входженням слова або використовувати регулярні вирази, виконання яких може бути повільним.

Для забезпечення високої швидкості обробки запитів у системах повнотекстового пошуку використовується така структура як інвертований індекс [1]. Інвертований індекс складається зі списку всіх слів, які містяться в будь-якому документі, і для кожного слова зберігається список документів, в яких слово присутнє. Документ у даному випадку – це одиниця даних, котру ми шукаємо, наприклад, наукова стаття чи кулінарний рецепт. Слова, що потрапляють у індекс, називаються термами.

Людська мова є дуже різнобічною, і часто те, що ми шукаємо, може бути присутнім у тексті в різних формах. Наприклад, коли ми шукаємо документи за словом «relax», нам буде все одно, чи написане це слово в документі з малої літери (в середині речення) чи з великої літери (на початку речення). Також, скоріш за все, нам підійдуть документи, де це слово присутнє в третій особі – «relaxes» – чи в іншому часі – «relaxed».

Таким чином, пошук по одному слову насправді може означати пошук по багатьох, хоча й схожих словах. У системах повнотекстового пошуку це вирішується шляхом використання попередньої обробки слів для перетворення їх на терми, що потім будуть записані в інвертований індекс. Попередня обробка може включати:

– Ігнорування стоп-слів, або шумових слів. Це слова, що не несуть смислового навантаження, тому їх користь та роль для пошуку не суттєва. Зазвичай це артиклі (a, an, the), сполучники (and, or), прийменники (in, on), займенники (I, he, she), форми дієслова to be (be, have, had). Ці слова просто не включаються до інвертованого індексу.

– Знаходження синонімів до слова у спеціальному словнику синонімів. Для слова «relax» ми можемо також віднайти слово «rest».

– Перевірка слова на одрук. Це допоможе виявити, що написавши «realx», людина на справді мала на увазі «relax».

– Стемінг, або зведення слова до його кореня. Стемінг перетворить слова «relax», «relaxes» та «relaxed» на один і той же терм «relax», що є коренем цього слова. Таким чином у індекс потрапляє лише обрізана форма «relax», а ось посилання від цього терму в індексі будуть присутні на всі документи, де є слово «relax» в будь-яких особі або часі.

Список методів попередньої обробки не обмежується вищеназваними, однак ми розглянемо детальніше саме стемінг.

Таблиця 1. Використання Porter2 на різних формах слів

Початкове слово	Porter2	Початкове слово	Porter2
marker	mark	added	ad
markers	mark	adds	add
market	market	user's	user
add	add	organization	organ
addition	addit	organ	organ
additive	addit	university	univers
adding	ad	universe	univers

Для більш точного пошуку бібліотека Apache Lucene надає спрощені реалізації алгоритмічних стемерів. Ці стемери містять меншу кількість правил та виконують м'якший стемінг. Розглянемо ці стемери в порядку зменшення їх жорсткості.

Г. Kstem

Цей стемер поєднує у собі алгоритмічний та словниковий підходи. Загалом він використовує лише три прості правила: зведення множини до однини («books» – «book»), перетворення минулого часу на теперішній («looked» – «look») та прибирання закінчення -ing («looking» – «look»). Також цей стемер містить у собі обмежений словник, що дозволяє знаходити відповідність між такими словами як «amplification» та «amplify», «european» та «europe».

Д. Мінімальний (Minimal, EnglishMinimalStemmer)

Цей стемер виконує мінімальну функціональність, що полягає в зведенні до однини: «books» – «book».

Е. Присвійний (Possessive, EnglishPossessiveFilter)

Стемер прибирає ознаки присвійної форми слова: «user's» – «user».

Результати роботи спрощених стемерів на різних формах слів приведені у таблиці 2. Для тестування були використані ті ж самі слова, що і для перевірки алгоритму Porter2 (таблиця 1), відсутні слова, що для усіх стемерів залишилися незмінними після стемінгу.

Таблиця 2. Використання спрощених стемерів на різних формах слів

Початкове слово	Kstem	Minimal	Possessive
markers	marker	marker	markers
adding	add	adding	adding
added	add	added	added
adds	add	add	add
user's	user's	user'	user

Можна помітити, що Мінімальний стемер є дещо грубим, адже присвійна форма «user's» обрізана просто шляхом викидання останньої букви «s», і отриманий терм, до якого увійшов апостроф. Усі три стемери не мають помилки надлишкового стемінгу, адже слова «university», «universe», «organization», «organ» не підпадають під жодне їх правило. Також для стемеру Kstem відсутня проблема недостатнього стемінгу, і всі форми дієслова «add» приводяться до єдиного терму.

IV. Вибір оптимального стемеру

Вибір певного алгоритму стемінгу залежить в першу чергу від того, який ступінь жорсткості обрізання слів є найбільш сприятливим для користувачів програмної системи. При значному розмірі корпусу документів на вибір алгоритму також можуть впливати вимоги до швидкості та наявні технічні ресурси, зокрема пам'ять.

Тут вже постає питання віднайдення певної рівноваги. Жорсткий стемер може призвести до знаходження нерелевантних результатів через змішування змісту різних слів. М'який стемер може знаходити менше результатів, ніж хотілося б, бо буде вважати схожі слова зовсім різними та індексувати окремо, до того ж розмір такого індексу може бути дуже великим.

Використання жорсткого стемеру (Lovins або Porter2) має сенс для максимального угруповання схожих слів у задачі пошуку ключових слів [5]. Більш агресивний стемінг також корисний у випадках, коли результати пошуку направляються на подальшу машинну обробку, наприклад, споживаються алгоритмом кластеризації.

У випадку коли результати пошуку призначені для споживання людиною, м'який стемер зазвичай буде призводити до кращих результатів [6]. Найбільш м'яким серед популярних стемерів є Porter.

Якщо ж програмна система вимагає дуже «тонкого» пошуку, тобто надзвичайно важливо, щоб було знайдене саме те, що шукається, а не схожі слова, тоді можна використовувати стемери зі спрощеним набором правил, що, зокрема, надає Apache Lucene. Прикладом може бути система з пошуком по точних наукових термінах у медичних текстах.

Висновки

Розглянута роль стемінгу для повнотекстового пошуку, проаналізовані популярні стемери, а також спрощені алгоритми, що надають дуже м'який стемінг. Встановлено, що не існує єдиного оптимального алгоритму стемінгу, що підійшов би кожній програмній системі. На вибір алгоритму може впливати специфіка тексту документів, наявні обмеження щодо швидкості пошуку та доступної пам'яті, а також те, як саме використовуються результати пошуку.

Література References

[1] Grainger T., Potter T., "Solr in Action", Manning, 2014, с. 11-15.

[2] Hatcher E., Gospodnetic O., "Lucene in Action", Manning, 2004, с. 102-148.

[3] Lovins J. B., "Development of a stemming algorithm," Mechanical Translation and Computer Linguistic., vol.11, no.1/2, с. 22-31, 1968.

[4] Porter M.F., "An algorithm for suffix stripping". Program, 1980; 14, с. 130-137.

[5] Вуль В.А., "Электронные издания». БХВ-Петербург, 2003, с. 411-412.

Gormley C., Tong Z., "Elasticsearch: The Definitive Guide", O'Reilly Media, Inc., 2015, с. 369-371

ДОДАТОК Г

Лістинг коду

```
# coding: utf-8
# ### 1. COCO API loading

get_ipython().run_line_magic('load_ext', 'autoreload')
get_ipython().run_line_magic('autoreload', '2')

import os
from pycocotools.coco import COCO

# initialize COCO API for instance annotations
dataDir = '/home/marina/_caption/data/coco/'
dataType = 'val2014'
instances_annFile = os.path.join(dataDir,
'annotations/instances_{}.json'.format(dataType))
coco = COCO(instances_annFile)

# initialize COCO API for caption annotations
captions_annFile = os.path.join(dataDir,
'annotations/captions_{}.json'.format(dataType))
coco_caps = COCO(captions_annFile)

# ### 2. Plotting examples
#
# Next, we plot a random image from the dataset, along with its five
corresponding captions.  Each time we run the code cell below, a different
image is selected.
#
# In the project, we will use this dataset to train a model to generate
captions from images.

import numpy as np
```

```
import skimage.io as io
import matplotlib.pyplot as plt
get_ipython().run_line_magic('matplotlib', 'inline')

ids = list(coco.anns.keys())

# pick a random image and obtain the corresponding URL
ann_id = np.random.choice(ids)
img_id = coco.anns[ann_id]['image_id']
img = coco.loadImgs(img_id)[0]
url = img['coco_url']

photo = io.imread(url)
plt.axis('off')
plt.imshow(photo)
plt.show()

# load and display captions
annIds = coco_caps.getAnnIds(imgIds=img['id']);
anns = coco_caps.loadAnns(annIds)
coco_caps.showAnns(anns)

print(f'Image shape: {photo.shape}')

# ### 3. Download nltk module

import nltk
nltk.download('punkt')

# ### 4. Check available model


import pretrainedmodels

print(pretrainedmodels.model_names)
```

```
print(pretrainedmodels.pretrained_settings['inceptionv4'])

print(pretrainedmodels.pretrained_settings['xception'])


print(pretrainedmodels.pretrained_settings['nasnetalarge'])

# ### 4. Check augmentations

import cv2

from albumentations import Compose, HorizontalFlip, RandomBrightness,
RandomContrast, ShiftScaleRotate, HueSaturationValue, Blur, Resize, ToTensor

basic_augmentation = Compose([
    Resize(299, 299),
    Normalize()
], p=1.0)

soft_augmentation = Compose([
    Resize(299, 299),
    RandomContrast(p=0.15, limit=0.20),
    Blur(p=0.33, blur_limit=4),
    ShiftScaleRotate(shift_limit=0.15, scale_limit=0.15, rotate_limit=15,
border_mode=cv2.BORDER_REPLICATE, p=0.25),
    Normalize()
], p=1.0)

hard_color_augmentation = Compose([
    Resize(299, 299),
    RandomContrast(p=0.25, limit=0.20),
    Blur(p=0.33, blur_limit=4),
    ShiftScaleRotate(shift_limit=0.15, scale_limit=0.15, rotate_limit=15,
border_mode=cv2.BORDER_REPLICATE, p=0.25),
    HueSaturationValue(hue_shift_limit=0.15, p=0.25),
    RandomBrightness(p=0.15, limit=0.25),
```

```
    Normalize()
], p=1.0)

hard_space_augmentation = Compose([
    Resize(299, 299),
    RandomContrast(p=0.15, limit=0.20),
    Blur(p=0.33, blur_limit=4),
    ShiftScaleRotate(shift_limit=0.15, scale_limit=0.15, rotate_limit=25,
border_mode=cv2.BORDER_REPLICATE, p=0.25),
    HorizontalFlip(p=0.5)
    Normalize()
], p=1.0)

# ### 5. Load data

from data_loader import get_loader

# Set the minimum word count threshold.
vocab_threshold = 5

# Specify the batch size.
batch_size = 32

# Obtain the data loader.
data_loader = get_loader(transform=soft_augmentation,
                         mode='train',
                         batch_size=batch_size,
                         vocab_threshold=vocab_threshold,
                         vocab_from_file=False,
                         cocoapi_loc='/home/marina/_caption/data/coco')

data_loader.dataset.vocab.__dict__

data_loader.dataset.vocab('a')

# ### 6. Analysing captions
```

```
### Getting longest caption

import numpy as np
import skimage.io as io
import matplotlib.pyplot as plt

get_ipython().run_line_magic('matplotlib', 'inline')

lens = np.array(data_loader.dataset.caption_lengths)
max_len_idx = np.argmax(lens)

d = data_loader.dataset

ann_id = d.ids[max_len_idx]
caption = d.coco.anns[ann_id]["caption"]
print(caption)
print(lens[max_len_idx])
img_id = d.coco.anns[ann_id]["image_id"]

img = d.coco.loadImgs(img_id)[0]
url = img['coco_url']

# print URL and visualize corresponding image
print(url)
I = io.imread(url)
plt.axis('off')
plt.imshow(I)
plt.show()

len(data_loader.dataset.caption_lengths)

### Plotting caption length

import numpy as np
import seaborn as sns
```

```
import matplotlib.pyplot as plt

get_ipython().run_line_magic('matplotlib', 'inline')
sns.set_style('whitegrid')

lens = np.array(data_loader.dataset.caption_lengths)
lens_counts = np.bincount(lens)

plt.figure(figsize=(12,6))
plt.bar(np.arange(len(lens_counts)), lens_counts)
# ax.set_yscale('log')
plt.xlabel('Caption length')
plt.ylabel('Number of captions')
plt.show()

from collections import Counter

# Tally the total number of training captions with each length.
counter = Counter(data_loader.dataset.caption_lengths)
lengths = sorted(counter.items(), key=lambda pair: pair[1], reverse=True)
for value, count in lengths:
    print('value: %2d --- count: %5d' % (value, count))

# ### 7. Check caption preprocessing

sample_caption = 'A crowd of bicycle riders are going down the street.'

import nltk

sample_tokens = nltk.tokenize.word_tokenize(str(sample_caption).lower())
print(sample_tokens)

sample_caption = []

start_word = data_loader.dataset.vocab.start_word
print('Special start word:', start_word)
```

```
sample_caption.append(data_loader.dataset.vocab(start_word))
print(sample_caption)

sample_caption.extend([data_loader.dataset.vocab(token) for token in
sample_tokens])
print(sample_caption)

end_word = data_loader.dataset.vocab.end_word
print('Special end word:', end_word)

sample_caption.append(data_loader.dataset.vocab(end_word))
print(sample_caption)

import torch

sample_caption = torch.Tensor(sample_caption).long()
print(sample_caption)

# ### 8. Explore vocabulary

# Preview the word2idx dictionary.
print (dict(list(data_loader.dataset.vocab.word2idx.items())[:10]))

# Print the total number of keys in the word2idx dictionary.
print('Total number of tokens in vocabulary:',
len(data_loader.dataset.vocab))

# Minimum word count threshold.
vocab_threshold = 5

# Obtain the data loader.
data_loader = get_loader(transform=soft_augmentation,
                         mode='train',
                         batch_size=batch_size,
                         vocab_threshold=vocab_threshold,
                         vocab_from_file=False,
```

```
                           cocoapi_loc='/home/marina/_caption/data/coco')
# Print the total number of keys in the word2idx dictionary.
print('Total number of tokens in vocabulary:',
len(data_loader.dataset.vocab))

unk_word = data_loader.dataset.vocab.unk_word
print('Special unknown word:', unk_word)

print('All unknown words are mapped to this integer:',
data_loader.dataset.vocab(unk_word))
print ("For example:")
print("'blahblah' is mapped to", data_loader.dataset.vocab('blahblah'))

# ### 9. Check model

# Import EncoderCNN and DecoderRNN.
from model import EncoderCNN, DecoderRNN

# Specify the dimensionality of the image embedding.
embed_size = 256

# Initialize the encoder. (We can add additional arguments if necessary.)
encoder = EncoderCNN(embed_size)

# Move the encoder to GPU if CUDA is available.
if torch.cuda.is_available():
    encoder = encoder.cuda()

# Move the last batch of images from Step 2 to GPU if CUDA is available
if torch.cuda.is_available():
    images = images.cuda()
# Pass the images through the encoder.
features = encoder(images)

print('type(features):', type(features))
print('features.shape:', features.shape)
```

```
# Specify the number of features in the hidden state of the RNN decoder.
hidden_size = 512

# Store the size of the vocabulary.
vocab_size = len(data_loader.dataset.vocab)

# Initialize the decoder.
decoder = DecoderRNN(embed_size, hidden_size, vocab_size)
# Move the decoder to GPU if CUDA is available.
if torch.cuda.is_available():
    decoder = decoder.cuda()

# Move the last batch of captions (from Step 1) to GPU if cuda is availble
if torch.cuda.is_available():
    captions = captions.cuda()
# Pass the encoder output and captions through the decoder
outputs = decoder(features, captions)

print('type(outputs):', type(outputs))
print('outputs.shape:', outputs.shape)
```

ДОДАТОК Д

Електронні матеріали (CD)