ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

УДК 004.85 DOI https://doi.org/10.35546/kntu2078-4481.2024.1.25

К. О. АНТІПОВА
доктор філософії,
старший викладач кафедри інженерії програмного забезпечення
Чорноморський національний університет імені Петра Могили
ORCID: 0000-0002-9012-5290

В. С. РАЛЕНКО
викладач кафедри інженерії програмного забезпечення
Чорноморський національний університет імені Петра Могили
ORCID: 0009-0009-4161-8468

ВИКОРИСТАННЯ РІЗНИХ ВИДІВ ТОКЕНІЗАТОРІВ В ТРАНСФОРМЕРНИХ АРХІТЕКТУРАХ ДЛЯ ЗАДАЧІ МАШИННОГО ПЕРЕКЛАДУ

Токенізація є першим кроком майже для всіх завдань обробки природної мови, і всі сучасні мовні моделі використовують алгоритми токенізації підслів для обробки вхідного тексту. Оскільки різні мови мають унікальні властивості, розробка алгоритму токенізації зазвичай є специфічною для конкретної мови. Попередньо навчені моделі для мов з обмеженими ресурсами для тренування використовують ті ж самі токенізатори, що і моделі для англійської. Вплив алгоритмів токенізації може бути різним для мов з обмеженими ресурсами, де слова можуть мати префікси та суфікси. Крім того, вплив різних методів токенізації не досліджено детально для малоресурсних мов, зокрема для української. В роботі виконується навчання токенізаторів типу WordPiece, BPE та Unigram для дослідження їхньої ефективності з точки зору точності машинного перекладу речень з англійської на українську. Щоб провести експериментальне порівняння роботи токенізаторів для задачі перекладу з англійської на українську, не використовувалася існуюча попередньо підготовлена мовна модель. Натомість було здійснено попереднє навчання власних мовних моделей середнього розміру на основі конфігурації та процедури навчання моделі Marian. Розроблений конвеєр операцій складається зі збору та очищення навчального корпусу пар речень, навчання токенізатора зі словником фіксованої довжини і попереднього навчання глибинної мовної моделі за допомогою обраного токенізатора. Після цього було виконано оцінку точності моделей із використанням таких метрик, як SacreBLEU та ROUGE. Отримані експериментальні результати підкреслюють роль токенізації в мовному моделюванні, зокрема для морфологічно багатих мов. Крім того, вища морфологічна вірогідність токенізації Unigram призводить до кращої продуктивності виконання завдання машинного перекладу природної мови.

Ключові слова: *машинний переклад, мовна модель, токенізація, трансформерна архітектура.*

K. O. ANTIPOVA
Ph.D., Senior Lecturer at the Department of Software Engineering
Petro Mohyla Black Sea National University
ORCID: 0000-0002-9012-5290

V. S. RALENKO
Lecturer at the Department of Software Engineering
Petro Mohyla Black Sea National University
ORCID: 0009-0009-4161-8468

IMPLEMENTATION OF DIFFERENT TYPES OF TOKENIZERS IN TRANSFORMER ARCHITECTURE FOR MACHINE TRANSLATION TASK

Tokenization is the first step for almost all natural language processing tasks, and all modern language models use subword tokenization algorithms to process the input text. Due to the unique characteristics of various languages, the development of a tokenization algorithm typically requires language-specific customization. Pre-trained models for languages with limited training resources use the same tokenizers as models for English. The impact of tokenization algorithms may be different for resource-constrained languages, particularly those where words commonly feature prefixes and suffixes. In addition, the impact of different tokenization methods has not been studied in detail for low-resource languages, including Ukrainian. In this work, we train tokenizers such as WordPiece, BPE, and Unigram to

study their effectiveness in terms of the accuracy of machine translation of sentences from English into Ukrainian. To conduct an experimental comparison of tokenizers for the English to Ukrainian translation task, we did not use an existing pre-trained language model. Instead, we pre-trained our own medium-sized language models based on the configuration and training procedure of the Marian model. The developed pipeline of operations consists of collecting and cleaning the training corpus of sentence pairs, training a tokenizer with a fixed-length dictionary, and pre-training a deep language model using the selected tokenizer. After that, the accuracy of the models was evaluated using such metrics as SacreBLEU and ROUGE. The obtained experimental results emphasize the role of tokenization in language modeling, in particular for morphologically rich languages. In addition, the higher morphological accuracy of Unigram tokenization leads to better performance in natural language machine translation tasks.

Key words: *language model, machine translation, tokenization, transformer architecture.*

Постановка проблеми

Глибинні мовні моделі набули популярності з впровадженням маскованого мовного моделювання, в основі якого лежить архітектура трансформеру. Успіх попередньо навчених мовних моделей-трансформерів в обробці природної мови призвів до появи широкого спектру налаштувань тренування моделей. Токенізація є першим кроком майже для всіх завдань обробки природної мови, і всі сучасні мовні моделі використовують алгоритми токенізації підслів для обробки вхідного тексту.

Токенізація підслів, популярна в літературі з нейронного машинного перекладу, створює токени на різних рівнях деталізації, від окремих символів до цілих слів. У результаті рідкісні слова розбиваються на підслова або на символи. Алгоритми токенізації підслів складаються з двох компонентів: процедури побудови словника, яка повертає словник потрібного розміру на основі вхідного тексту, і процедури токенізації, яка застосовує побудований словник до нового тексту, повертаючи послідовність токенів. Процедура токенізації тісно пов'язана з процедурою побудови словника.

Токенізація підслів надає моделі можливість підтримувати відносно невеликий розмір словникового запасу, і при цьому вивчати значущі контекстно-незалежні представлення. Крім того, токенізація підслів надає моделі можливість обробляти нові невідомі слова, розкладаючи їх на відомі підслова.

Оскільки різні мови мають унікальні властивості, розробка алгоритму токенізації зазвичай є специфічною для конкретної мови і вимагає певних лінгвістичних знань. Однак лише декілька з понад 7000 існуючих мов користуються перевагами спеціалізованих алгоритмів токенізації, в той час як для інших мов використовуються стандартизовані токенізатори по пробілах, який не може врахувати тонкощі різних мов.

Багатомовні моделі архітектурно схожі на одномовні моделі на основі трансформерів, але вони попередньо навчаються на багатомовних корпусах текстів. Великі мовні моделі спочатку навчаються на англійській мові, а попередньо навчені моделі для мов з обмеженими ресурсами для тренування використовують ті ж самі токенізатори. Вплив алгоритмів токенізації може бути різним для мов з обмеженими ресурсами, де слова можуть мати префікси та суфікси. Крім того, вплив різних методів токенізації не досліджено детально для малоресурсних мов, зокрема для української.

Аналіз останніх досліджень і публікацій

Сучасні мовні моделі-трансформери використовують алгоритми токенізації підслів, що базуються на кодуванні байтових пар (BPE) або Unigram. Оригінальна модель-трансформер використовує BPE [1].

Кодування байтових пар (Byte-Pair Encoding) було вперше запропоновано в 2015 році для нейронного машинного перекладу рідких слів [2]. BPE має за основу пре-токенізатор, який розбиває тренувальні дані на слова. Unigram – це алгоритм токенізації підслів, представлений у 2018 році для покращення моделей трансляції нейронних мереж [3].

WordPiece – це алгоритм токенізації підслів, який був вперше запропонований для голосового пошуку японською та корейською мовами та є дуже схожим на алгоритм BPE [4]. Алгоритм WordPiece використовується Google для попереднього навчання моделі BERT [5]. ROBERTA та BART застосовують BPE для необроблених байтів замість символів юнікоду [6; 7].

Наведені алгоритми токенізації мають однакову проблему: вважається, що у вхідному тексті пробіли використовуються для розділення слів. Однак не всі мови використовують саме пробіли як роздільники. Щоб загалом вирішити цю проблему, у 2018 році був представлений алгоритм SentencePiece як простий і незалежний від мови токенізатор і детокенізатор підслів для нейронної обробки тексту [8].

Моделі XLNET [9], ALBERT [10], T5 [11] та Marian [12] використовують бібліотеку SentencePiece, яка реалізує як BPE, так і токенізацію Unigram. Вплив токенізації не розглядається в жодній з вищезгаданих робіт, за винятком [6], де зазначено, що WordPiece дає невелику перевагу над BPE в їхньому дослідженні.

В публікації [8] представлено метод токенізації Unigram в контексті машинного перекладу і показано, що цей метод за продуктивністю схожий на BPE. В роботі [13] наведені подальші експерименти для дослідження впливу токенізації на результати нейронного машинного перекладу, але в усіх експериментах використовувався спільний словник BPE. В роботі [14] представлено порівняльний аналіз алгоритмів сімейства BPE, але вони не порівнюються з Unigram.

Формулювання мети дослідження

Метою дослідження є порівняння ефективності різних алгоритмів токенізації з точки зору точності машинного перекладу речень з англійської на українську. Для того, щоб оцінити ефективність алгоритмів, тренуються різні види токенізаторів, а також виконується попереднє навчання моделей на основі процедури попереднього навчання моделі Marian. В якості вхідних даних виступає набір з речень англійською та відповідних речень українською.

Виклад основного матеріалу дослідження

Токенізація – це важливий етап попередньої обробки тексту для підготовки вхідних токенів для глибинних мовних моделей. WordPiece і BPE – це методи, які використовуються в таких відомих моделях, як BERT і GPT. Однак вплив токенізації може бути іншим для морфологічно багатих мов, де багато слів можна утворити додаванням префіксів і суфіксів. З токенізацією пов'язані такі операції, як згортання регістру, нормалізація, стеммінг, лематизація, видалення стоп-слів тощо. Алгоритми, що використовуються для виконання цих операцій, не є універсальними для всіх мов, оскільки кожна мова має свої унікальні особливості і відрізняється від інших лексичною, семантичною та морфологічною складністю.

Кодування байтових пар – це часто використовуваний токенізатор для попередньо навчених мовних моделей. Гранулярність BPE можна вважати середньою між символьним і словесним рівнями, так що токени здебільшого є підсловами, залежно від обсягу словника. У цьому методі спочатку витягуються всі унікальні слова. Потім будується базовий словник з усіх символів, що зустрічаються в унікальних словах. BPE вважається неоптимальним для тренування мовних моделей, оскільки цей метод неефективно використовує словниковий простір [15].

Подібно до BPE, WordPiece також базується на об'єднанні символів у тексті. Його основна відмінність від BPE полягає в тому, що WordPiece об'єднує символи з метою максимізації оцінки правдоподібності мовного моделювання, тобто коли ймовірність об'єднаних символів, поділена на індивідуальні ймовірності символів, є більшою, ніж у будь-якій іншій парі символів. WordPiece і BPE – це методи, в основі яких лежить розрахунок частоти появи слів або підслів, що потрібно для покращення здатності представляти окремі одиниці мови. Ці методи також допомагають розбити слова, які токенізер не вивчав під час тренування.

Токенізація на основі статистики появи слів призводить до того, що представлення даних залежать від того, наскільки поширеними є слова, а не від їх семантики. З іншого боку, пропонується застосовувати регуляризацію підслів шляхом використання декількох сегментацій підслів для підвищення стійкості моделей нейронного машинного перекладу, як це зроблено в SentencePiece [8].

Всі методи токенізації реалізують наступні кроки:

– нормалізація (будь-яке очищення тексту, яке вважається необхідним, наприклад, видалення пробілів або наголосів, нормалізація Unicode тощо);

– попередня токенізація (розбиття вхідної послідовності на слова);

– проведення вхідних даних через модель (з використанням попередньо токенізованих слів для створення послідовності токенів);

– постобробка (додавання спеціальних токенів токенізатора, створення маски уваги та ідентифікаторів типу токенів).

Після попередньої токенізації створюється набір унікальних слів і визначається частота появи кожного слова в тренувальному наборі даних. Далі BPE створює базовий словник, що складається з усіх символів, які зустрічаються в наборі унікальних слів, і вивчає правила поєднання для формування нового символу з двох символів базового словника. Остаточний словник будується шляхом об'єднання символів відповідно до частот послідовних символів або підслів. Оскільки BPE оперує з байтовими представленнями, словник може охоплювати лексеми з різних мов і неформальні послідовності символів, такі як емодзі.

WordPiece спочатку ініціалізує словник, щоб включити кожен символ, присутній серед тренувальних даних, і поступово вивчає задану кількість правил поєднання. На відміну від BPE, WordPiece вибирає не найпоширенішу пару символів, а ту, яка максимізує ймовірність того, що тренувальні дані будуть додані до словника.

На відміну від BPE або WordPiece, Unigram ініціалізує свій базовий словник великою кількістю символів і поступово видаляє символи, щоб отримати менший словник. Базовий словник може, наприклад, відповідати всім попередньо лексемованим словам і найпоширенішим підрядкам.

Оскільки, на відміну від WordPiece, Unigram не базується на правилах поєднання, алгоритм має декілька способів токенізації нового тексту після навчання. Крім збереження словника Unigram зберігає ймовірність кожного токена в тренувальному наборі даних, щоб ймовірність кожної можливої токенізації можна було обчислити після завершення процесу навчання. Алгоритм вибирає найбільш ймовірну токенізацію, але також пропонує можливість вибірки можливої токенізації відповідно до її ймовірності. Ці ймовірності визначаються втратою, на якій навчено токенізатор.

Таблиця 1

Приклади результату роботи різних токенізаторів

WordPiece	BPE	Unigram
'system', '##atical', '##ly'	'system', 'atic', 'ally'	'–system', 'atic', 'ally'
'standard', '##ized'	'Ġstandard', 'ized'	'–standard', 'ized'
'illust', '##rate'	'Ġillustr', 'ate'	'–illustrate'
'complete', '##ness'	'comp', 'let', 'eness'	'–complete', 'ness'
'одно', '##вимір', '##ні', 'масив', '##и'	'Ð²Ð°', 'ĠÐ¾Ð´Ð½', 'Ð¾Ð²Ð¸Ð¹', 'Ð¼ÑĸÑĢ', 'Ð½Ñĸ', 'ĠÐ¼Ð°Ñģив', 'Ð²Ð¸Ð¹'	'–одно', 'вимірн', 'і', '–масив', 'и'
'спеціалі', '##зо', '##ван', '##ии'	'Ñģ', 'Ð¿ÐµÑĨÑĸ', 'Ð°Ð»ÑĸÐ·', 'Ð¾Ð²Ð°Ð½Ð¸Ð¹'	'–', 'спеціалізован', 'ии'
'на', '15', '-', 'му', 'повер', '##сі'	'Ð½Ð°', 'Ġ15', '-', 'Ð¼Ñĥ', 'ĠÐ¿Ð¾Ð²ÐµÑĢ', 'Ñģі'	'–на', '–15', '-', 'му', '–по', 'вер', 'сі'

Таблиця 2

Розрахунок метрик для моделей

Модель	Метрики		
	Accuracy	SacreBLEU	ROUGE-1
Власна, WordPiece	85.91	42.75	76.3
Власна, BPE	86.4	43.12	79.59
Власна, Unigram	86.52	43.63	83.21
T5-base	86.27	28.0	42.05

SentencePiece розглядає вхідні дані як необроблений вхідний потік, таким чином включаючи пробіл у набір символів для використання. Після чого застосовується алгоритм Unigram для створення відповідного словника. SentencePiece, як алгоритм токенізації для попередньої обробки тексту, можна використовувати з будь-якою з існуючих моделей обробки природної мови. Цей алгоритм розглядає текст як послідовність символів Unicode та замінює пробіли спеціальним символом «_». Всі моделі трансформерів, які використовують SentencePiece, використовують його в поєднанні з Unigram.

Щоб провести експериментальне порівняння цих трьох методів для задачі перекладу з англійської на українську, не використовувалася існуюча попередньо підготовлена мовна модель. Натомість було здійснено попереднє навчання власних мовних моделей середнього розміру на основі конфігурації та процедури навчання моделі Marian [16]. В якості базової моделі для порівняння результатів роботи власних моделей була використана модель T5-base, тренована на датасеті C4, що складається з 364 613 570 речень.

Вхідні дані представлені датасетом kde4, який складається з 829 789 пар речень [17]. Після токенізації отримано три словника на 23 769 (WordPiece), 24 700 (BPE) та 24 989 (Unigram) токенів відповідно.

Процес токенізації відбувався у три етапи: застосування нормалізації для очищення тексту від неприпустимих символів, навчання токенізатора відповідно до заданого розміру словника, і обробка тексту за допомогою навченого токенізатора для отримання токенізованих даних попереднього навчання. Також було застосовано згортання регістру та нормалізацію NFC.

Розроблений конвеєр операцій складається зі збору та очищення навчального корпусу пар речень, навчання токенізатора зі словником фіксованої довжини і попереднього навчання глибинної мовної моделі за допомогою обраного токенізатора. Після цього було виконано оцінку точності моделей та ефективності токенізаторів.

Метрика SacreBLEU – це різновид BLEU (Bilingual Evaluation Understudy), що є орієнтованою на точність метрикою, яка вимірює перекриття між згенерованим та еталонним текстом. ROUGE (Recall Oriented Understudy for Gisting Evaluation) – це метрика, який вимірює перекриття між згенерованим і еталонним текстом з точки зору відтворення.

Отже, власні треновані моделі, які є значно меншими за сучасні моделі, дають не менший рівень точності. Моделі, навчені із використанням токенізації Unigram, забезпечують більшу точність, ніж моделі з BPE або WordPiece.

Висновки

В роботі досліджено вплив методів токенізації для речень українською, яка є мовою з обмеженою кількістю попередньо навчених мовних моделей. Для виконання цього завдання, були треновані мовні моделі невеликого розміру на основі моделі Marian з різними алгоритмами токенізації. Виконано порівняння трьох токенізаторів на різних рівнях гранулярності: їхні вихідні дані варіюються від символів до слів різної довжини. Було проведено тренування наступних токенізаторів: BPE, WordPiece, Unigram. Отримані експериментальні результати підкреслюють роль токенізації в мовному моделюванні, зокрема для морфологічно багатих мов. Крім того, вища морфологічна вірогідність токенізації Unigram призводить до кращої продуктивності виконання завдання перекладу природної мови.

Список використаної літератури

1. Vaswani, A. et al. (2017). Attention Is All You Need. *31st Conference on Neural Information Processing Systems (NIPS)*. https://doi.org/10.48550/arXiv.1706.03762
2. Sennrich, R., Haddow, B., & Birch, A. (2016). Neural Machine Translation of Rare Words with Subword Units. In *54th Annual Meeting of the Association for Computational Linguistics* (pp. 1715–1725). Association for Computational Linguistics (ACL).
3. Kudo, T. (2018). Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 66–75).
4. Schuster, M., & Nakajima, K. (2012). Japanese and Korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 5149–5152). IEEE.
5. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
6. Liu, Y., Ott, M., et al. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
7. Lewis, M., Liu, Y., et al. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.
8. Kudo, T., & Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. *Conference on Empirical Methods in Natural Language Processing*.
9. Zhilin Yang, Zihang Dai, et al. (2019). XLNet: Generalized autoregressive pretraining for language understanding. arXiv preprint arXiv:1906.08237.
10. Zhenzhong Lan, Mingda Chen, et al. (2019). ALBERT: A lite BERT for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942.
11. Colin Raffel, Noam Shazeer, et al. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv preprint arXiv:1910.10683.
12. MarianNMT. (n.d.). https://marian-nmt.github.io/
13. Domingo, M., García-Martínez, M., Helle, A., Casacuberta, F., & Herranz, M. (2019). How much does tokenization affect neural machine translation?. In *International Conference on Computational Linguistics and Intelligent Text Processing* (pp. 545–554). Cham: Springer Nature Switzerland.
14. Matthias Gall´e. (2019). Investigating the effectiveness of BPE: The power of shorter sequences. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (pp. 1375–1381). Association for Computational Linguistics.
15. Bostrom, K., & Durrett, G. (2020). Byte pair encoding is suboptimal for language model pretraining. *arXiv preprint arXiv:2004.03720*.
16. Wolf, T., Debut, L., Sanh, V., Chaumond, J., et al. (2019). HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv, abs/1910.03771*.
17. Hugging Face. (n.d.). https://huggingface.co/datasets/kde4