УДК 004.8

РОЛЬ ТОКЕНІЗАЦІЇ У МОВНИХ МОДЕЛЯХ ШТУЧНОГО ІНТЕЛЕКТУ

Д. Гребенюк, група КН-23
Науковий керівник: ***В. Резніченко****,* викладач
Центральноукраїнський національний технічний університет

Токенізація – ключовий етап обробки тексту в сучасних мовних моделях штучного інтелекту (ШІ). Вона полягає у розбитті тексту на дрібніші одиниці – токени, які далі перетворюються на числові представлення. Саме завдяки цьому ШІ здатен математично аналізувати природну мову та здійснювати високоточні обчислення, пов’язані з мовними структурами. Цей процес є відправною точкою, з якої починається обробка тексту. Водночас він є і завершальним етапом: спочатку текст перетворюється на токени перед обробкою моделлю, а потім — ці числові представлення декодуються назад у зрозумілу для людини форму після завершення генерації результату [1].

Коректний вибір методу токенізації є критично важливим, адже він впливає не лише на точність вихідного результату, а й на швидкість навчання та розгортання моделі, ефективність використання обчислювальних ресурсів, а також на її здатність розпізнавати слова різних мов, адаптуватися до граматичних і стилістичних відмінностей, працювати з помилками, скороченнями, неологізмами та іншими складнощами живої мови.

У цьому контексті доцільно розглянути основні типи токенізації, які застосовуються в сучасних мовних моделях:

1. Слівна токенізація. Найпростіший тип, коли текст розбивається на окремі слова. Наприклад: "Was the email sent?" → ["Was", "the", "email", "sent", "."]. Цей підхід інтуїтивно зрозумілий, однак має недоліки: погано працює з новими словами, морфологічними формами або мовами, де межі між словами не завжди чіткі (наприклад, китайська) [1].

2. Символьна токенізація. У цьому випадку кожна літера або символ розглядається як окремий токен: ["W", "a", "s", " ", "t", "h", "e", ...]. Такий підхід дозволяє моделі бачити текст у найдрібніших деталях і чудово працює з незнайомими словами чи помилками, але значно збільшує довжину послідовностей, що ускладнює навчання [1].

3. Підслівна токенізація. Найпопулярніший нині підхід. Він розбиває слова на морфеми або значущі частини: "telehealthcare" → ["tele", "health", "care"]. Це дозволяє балансувати між точністю та довжиною послідовності, зберігаючи можливість розуміти нові слова на основі відомих частин [1][2].

Розглянемо актуальні алгоритми токенізації:

- Byte Pair Encoding (BPE). Побудований на ідеї злиття найчастіших пар символів у нові токени. BPE є ефективним компромісом між простотою реалізації та потужністю результату. Його використання дозволяє сформувати відкритий словник, що добре працює з рідкісними словами [3].

- WordPiece. Запропонований у контексті моделі BERT. Алгоритм здійснює пошук найдовших можливих підслів, що існують у попередньо визначеному словнику, забезпечуючи максимально релевантне представлення тексту [2].

- Unigram Language Model. Ґрунтується на ймовірнісному підході: із множини варіантів сегментації обирається той, що має найвищу загальну ймовірність. Реалізована в SentencePiece, добре показує себе в багатомовному середовищі, особливо для мов із рідкісними структурами [4].

Таблиця 1. Порівняльна таблиця алгоритмів токенізації [2-4]

Алгоритм	**Принцип**	**Переваги**	**Недоліки**	**Приклад використання**
BPE	Частотні пари	Добре працює для частих слів	Не завжди ефективний для складних форм	GPT, NMT
WordPiece	Найдовше підслово	Ефективність при обробці морфології	Потребує точного словника	BERT, інші Google-моделі
Unigram	Ймовірності	Гнучкість, підтримка багатьох мов	Вищі обчислювальні витрати	SentencePiece

Важливим фактором у процесі токенізації є гранулярність токенів – тобто їхня довжина або обсяг інформації, що закладено в одному токені. Наприклад, великими токенами можуть бути цілі слова або навіть сталі фрази, тоді як дрібніми – окремі символи чи морфеми. Дрібні токени більш гнучкі, оскільки дозволяють моделі генерувати або розпізнавати нові або незвичні слова. Однак така детальність значно збільшує кількість кроків, які повинна обробити модель, що веде до збільшення обчислювальних витрат. У той час як великі токени дають змогу швидше обробити текст, вони менш ефективні при роботі з новими мовними одиницями [1].

Дослідження також показують, що застосування підслівної регуляризації, тобто випадкового сегментування токенів під час етапу навчання, допомагає покращити узагальнення моделі. Це знижує кількість помилок, особливо у завданнях машинного перекладу, транскрипції або розпізнавання мови для низькоресурсних мов, де бракує об’ємів якісних текстів для навчання [5].

Таким чином, токенізація є критично важливою частиною у функціонуванні мовних моделей. Вибір типу токенізації, відповідного алгоритму та глибини обробки повинен враховувати завдання моделі, специфіку мови, з якою вона працює, і обмеження обчислювального середовища. Сучасна практика демонструє, що саме підслівна токенізація найкраще поєднує гнучкість, точність та ефективність. Вона дозволяє моделі не лише працювати з широким спектром мов, але й легко адаптуватися до появи нових термінів, змін у граматиці або стилі та забезпечує високу якість синтезу і розуміння тексту.

Список літератури

1. What is Tokenization in AI? Usage, Types, Challenges. AI21. URL: https://www.ai21.com/knowledge/tokenization/ (дата звернення: 14.05.2025).

2. WordPiece tokenization - Hugging Face LLM Course. Hugging Face. URL: https://huggingface.co/learn/llm-course/en/chapter6/6 (дата звернення: 14.05.2025).

3. Byte-Pair Encoding tokenization - Hugging Face LLM Course. Hugging Face. URL: https://huggingface.co/learn/llm-course/en/chapter6/5 (дата звернення: 14.05.2025).

4. GitHub - google/sentencepiece: Unsupervised text tokenizer for Neural Network-based text generation. GitHub. URL: https://github.com/google/sentencepiece (дата звернення: 14.05.2025).

5. Subword regularization: An analysis of scalability and generalization for end-to-end automatic speech recognition / E. Lakomkin та ін. Amazon Science. URL: https://www.amazon.science/publications/subword-regularization-an-analysis-of-scalability-and-generalization-for-end-to-end-automatic-speech-recognition (дата звернення: 14.05.2025).