Implementacja transformera oraz eksperymenty z wariantami mechanizmu uwagi (Performer, Reformer) w zadaniach klasyfikacji tekstu

Dokumentacja projektu – część II

Bartłomiej Borycki, Michał Iwaniuk

6 listopada 2025

1 Założenia techniki modelowania

Projektowany model opiera się na klasycznej architekturze typu **Transformer Encoder**, stanowiącej bazę dla większości współczesnych modeli językowych (m.in. BERT, RoBERTa, DeBERTa). Struktura jest w pełni modularna, co umożliwia wymianę kluczowych komponentów – w szczególności mechanizmu uwagi – przy zachowaniu wspólnego interfejsu i procesu treningowego. Celem architektury jest uzyskanie elastycznego środowiska badawczego, w którym można porównywać efektywność różnych wariantów atencji przy zachowaniu identycznej reszty toru przetwarzania.

1.1 Blok enkodera

Podstawowym elementem architektury jest powtarzany wielokrotnie **blok enkodera**, który odpowiada za przekształcenie sekwencji wejściowej do reprezentacji semantycznej o stałym wymiarze. Każdy blok składa się z dwóch głównych komponentów:

- **AttentionBlock** – implementacja wielogłowicowego mechanizmu uwagi (*Multi-Head Attention*), którego celem jest modelowanie zależności między tokenami niezależnie od ich położenia w sekwencji. Dla stabilności i efektywnego uczenia stosowane są połączenia rezydualne oraz warstwy normalizacji (`LayerNorm`).
- **MLPBlock** – dwuwarstwowa sieć liniowa z nieliniowością typu `GELU`, pełniąca funkcję projekcji w przestrzeni cech. Umożliwia nieliniową kombinację kontekstu wyuczonego przez uwagę oraz wprowadza dodatkową pojemność modelu.

Każdy blok enkodera jest zakończony połączeniem rezydualnym (*residual connection*) oraz normalizacją warstwy, co poprawia propagację gradientu i stabilność uczenia w głębokich sieciach.

Pozycjonowanie. Aby zachować informację o kolejności tokenów, model obsługuje kilka metod kodowania pozycji:

- *Learned positional embeddings* – wektor pozycji uczony wraz z resztą parametrów modelu,
- *Sinusoidal positional encoding* – deterministyczne, okresowe odwzorowanie pozycji oparte na funkcjach trygonometrycznych,
- *Rotary positional encoding (RoPE)* – metoda rotacyjnego osadzania pozycji bezpośrednio na wektorach zapytań i kluczy (Q,K), pozwalająca na zachowanie relacji odległości w przestrzeni uwagi.

W przypadku wariantu RoPE informacja o położeniu nie jest dodawana do embeddingów, lecz wprowadzana na etapie obliczania uwagi.

Pooling. Po zakończeniu przetwarzania sekwencji przez enkoder, jej reprezentacja jest agregowana do jednego wektora cech, który stanowi wejście dla warstwy klasyfikacyjnej. W zależności od zadania, możliwe są następujące metody poolingu:

- **CLS pooling** - wykorzystanie wektora przypisanego do specjalnego tokenu `[CLS]` (analogicznie jak w BERT),
- **Mean pooling** - uśrednienie reprezentacji wszystkich tokenów niebędących paddingiem,
- **Max/min pooling** - wybór ekstremalnych wartości cech w sekwencji.

Wszystkie warianty uwzględniają maskę `key_padding_mask`, co umożliwia poprawne pominięcie w obliczeniach tokenów `[PAD]`.

1.2 Wymienne warianty uwagi

Jednym z kluczowych założeń projektu jest umożliwienie porównania różnych implementacji mechanizmu uwagi w identycznej architekturze. W ramach systemu zaimplementowano trzy wymienne warianty:

- **Standardowa uwaga (`mha`, SDPA)** – klasyczna wersja wielogłowicowej uwagi softmax, obliczająca pełną macierz wag między wszystkimi tokenami w sekwencji. Maska paddingu jest stosowana w postaci maski addytywnej, a na wektorach zapytań i kluczy może opcjonalnie zostać użyty mechanizm RoPE.
- **Performer (`favor`)** – wariant oparty na aproksymacji softmax przy użyciu dodatnich cech losowych (*FAVOR+*). Pozwala on zredukować złożoność obliczeniową z $\mathcal{O}(n^2)$ do $\mathcal{O}(n)$ przy zachowaniu zbliżonej dokładności. Wariant ten umożliwia również zastosowanie RoPE.
- **Reformer (`lsh`)** – implementacja uwagi opartej na *Locality-Sensitive Hashing (LSH)*, w której tokeny są grupowane w bucketach o podobnych reprezentacjach, a uwaga jest obliczana lokalnie w oknach (typowo 3× `chunk`). Dodatkowo możliwe jest ograniczenie obliczeń do intra-bucket, co further redukuje złożoność. W tym wariancie stosowana jest wspólna projekcja dla Q i K.

Wszystkie powyższe mechanizmy są kompatybilne z wspólnym interfejsem `AttentionBlock`, co umożliwia ich wymianę bez modyfikacji pozostałych komponentów modelu. Dzięki temu możliwe jest bezpośrednie porównanie jakości reprezentacji oraz kosztów obliczeniowych poszczególnych podejść.

1.3 Głowice i zadania

Na wyjściu enkodera zastosowano odrębne moduły (*heads*) odpowiedzialne za specyficzne zadania:

- **SequenceClassificationHead** – moduł klasyfikacyjny operujący na wektorze uzyskanym po poolingu. Implementuje warianty pooler_type zgodne z architekturami BERT, RoBERTa lub bez dodatkowej warstwy projekcji. W etapie fine-tuningu pełni rolę głowicy decyzyjnej w zadaniu klasyfikacji tekstu.
- **MaskedLanguageModelingHead** – moduł przewidujący maskowane tokeny w zadaniu MLM (Masked Language Modeling). W zależności od konfiguracji może współdzielić wagi z embeddingami wejściowymi (*weight tying*), co redukuje liczbę parametrów i sprzyja spójności reprezentacji.

Oba moduły integrują się bezpośrednio z głównym enkoderem i mogą być dynamicznie przełączane w zależności od etapu treningu (*pretraining*, *TAPT*, *fine-tuning*). Takie rozwiązanie umożliwia zachowanie ciągłości uczenia pomiędzy różnymi zadaniami przy minimalnej liczbie zmian architektonicznych.

1.4 Cele treningowe

Pretrening (MLM). Maskowany model językowy, z wykorzystaniem danych z Wikipedii, z polityką maskowania 15% tokenów (80% `[MASK]`, 10% losowy token, 10% oryginał), z funkcją straty *cross-entropy* wyliczaną jedynie dla pozycji maskowanych.

TAPT (Task-Adaptive PreTraining). Dodatkowe dostosowanie reprezentacji poprzez trening MLM na danych z domeny docelowej (IMDB, AG News, ArXiv Classification). Każdy TAPT jest przeprowadzany oddzielnie dla danego zbioru.

Fine-tuning (klasyfikacja). Po etapach pretreningu i TAPT model jest dostrajany pod kątem klasyfikacji nadzorowanej — z nową głowicą klasyfikacyjną, przy zachowaniu architektury enkodera.

2 Ocena narzędzi i technik

Projekt zakłada pełną implementację modelu typu **Transformer Encoder** w środowisku **PyTorch**, z wykorzystaniem jedynie wybranych elementów ekosystemu Hugging Face do obsługi tokenizacji i zarządzania zbiorami danych.

2.1 Biblioteki i środowisko

Hugging Face `datasets` i `tokenizers`. Z ekosystemu Hugging Face wykorzystano wyłącznie moduły wspierające przygotowanie danych wejściowych:

- **`datasets` (v3.0.1)** – do pobierania, wstępnego przetwarzania i cache'owania zbiorów danych.
- **`tokenizers` (v0.21.0)** – do implementacji tokenizacji typu **WordPiece**, z zachowaniem kompatybilności z oryginalnym słownikiem BERT (`bert-base-uncased`). Moduł zapewnia wysoką wydajność oraz możliwość definiowania własnych reguł postprocessingu (`[CLS]`, `[SEP]`, `[MASK]`).

W projekcie celowo zrezygnowano z wykorzystania gotowych modeli dostępnych w bibliotece `transformers`, koncentrując się na implementacji architektury Transformer od podstaw, z pełną kontrolą nad jej komponentami i mechanizmami uwagi.

PyTorch. Całość implementacji modelu, mechanizmów uwagi oraz procesów treningowych zrealizowano w ramach frameworka **PyTorch** (v2.3.1) z wykorzystaniem środowiska `Python 3.11`. PyTorch zapewnia:

- imperatywny model programowania, ułatwiający tworzenie i debugowanie niestandardowych modułów sieciowych,
- natywną obsługę akceleracji GPU (CUDA 12.2 / cuDNN 9.1)„
- wsparcie dla trybów obliczeń o mieszanej precyzji (AMP, bfloat16),

Środowisko eksperymentalne. Kod źródłowy został zorganizowany w postaci modułowej struktury Python z konfiguracją w formacie YAML. Umożliwia to pełną automatyzację eksperymentów w trybach: *pretraining*, *TAPT/DAPT* oraz *fine-tuning*. Każdy eksperyment jest definiowany poprzez pojedynczy plik konfiguracyjny zawierający parametry modelu, danych i harmonogramu uczenia, co sprzyja reprodukowalności wyników.

2.2 Techniki przetwarzania tekstu

Tokenizacja: WordPiece. Zastosowano tokenizację typu **WordPiece**, która dzieli słowa na subtokeny według częstości ich występowania w korpusie. Wykorzystano oryginalny słownik modelu BERT, co zapewnia pełną zgodność z rozpoznawalnymi jednostkami językowymi i minimalizuje liczbę nieznanych tokenów (`[UNK]`). Proces tokenizacji został zrealizowany za pomocą własnego `WordPieceTokenizerWrapper`, który integruje funkcje ładowania i enkodowania danych oraz przygotowuje tensorowe reprezentacje wejściowe (`input_ids`, `attention_mask`) w formacie `TensorDataset`.

Przetwarzanie danych tekstowych. Podczas tokenizacji z wykorzystaniem `BertTokenizerFast`, (z którego korzysta `WordPieceTokenizerWrapper`) dane wejściowe są przetwarzane w kilku etapach określonych przez pipeline tokenizera z biblioteki `tokenizers`.

1. Normalizacja tekstu. Na tym etapie wykonywane są operacje zdefiniowane w konfiguracji konkretnego modelu BERT, takie jak:

- konwersja tekstu do małych liter (dla modeli typu *uncased*),
- usuwanie lub zastępowanie znaków niewidocznych,
- podstawowa normalizacja unicode (np. NFD).

2. Pre-tokenizacja. Na tym etapie `BertTokenizerFast` dzieli tekst na wstępne segmenty zgodne z regułami tokenizerów BERT. Obejmuje to:

- podział według białych znaków (spacje, tabulatory, nowe linie),
- wydzielanie znaków niealfanumerycznych jako osobnych segmentów (np. interpunkcji),
- zachowanie offsetów znakowych, umożliwiających późniejsze odwzorowanie tokenów na oryginalny tekst.

Uzyskane segmenty stanowią wejście dla właściwej tokenizacji WordPiece.

3. Tokenizacja WordPiece. Każde słowo jest rozbijane na podtokeny zgodnie ze słownikiem modelu. Fragmenty rozpoczynające się wewnątrz słowa otrzymują prefiks ##. W przypadku braku dopasowania wykorzystywany jest token `[UNK]`.

4. Dodawanie tokenów specjalnych. W zależności od konfiguracji tokenizera automatycznie dodawane są tokeny takie jak `[CLS]` i `[SEP]`.

5. Tokenizator generuje:

- `input_ids` – numeryczne identyfikatory tokenów zgodne ze słownikiem (vocabulary) modelu.
- `attention_mask` – maskę wskazującą tokeny rzeczywiste oraz `[PAD]`.

Podczas tokenizacji stosowane jest przycinanie sekwencji (*truncation*) zależne od maksymalnej długości kontekstu w zadaniu, oraz *dynamiczny padding* po tokenizacji, co pozwala na efektywne wykorzystanie zasobów GPU.

Strategia tokenizacji. W korpusie ArXiv Classification znaczna część dokumentów przekracza kilka tysięcy tokenów - posłuży on nam do testowania mechanizmów atencji na długich sekwencjach.

Ograniczenia długości podczas treningu:

- **IMDB, AG News:** $L_{\max}^{\text{train}} = 512$ tokenów.
- **ArXiv**: $L_{\max}^{\text{train}} = 8192$ tokenów. (Wariant Reformer/LSH; jeśli nie będzie to możliwe w naszej implementacji MHA z powodu zbyt małej ilości pamięci VRAM, wówczas wykorzystamy mechanizm FlashAttention.)

3 Schemat eksperymentów end-to-end

3.1 Opis ogólny

W celu przeprowadzenia pełnego eksperymentu porównawczego zaprojektowano plan uczenia składający się z etapów pretreningu, adaptacji domenowej (TAPT/DAPT) oraz końcowego fine-tuningu. Proces jest realizowany dla trzech wariantów mechanizmu uwagi, przy identycznych parametrach architektury i konfiguracji.

3.2 Etapy przetwarzania

1. **Etap I: Porównanie wariantów uwagi**
   Trzy modele trenowane niezależnie:

   - *Transformer:* klasyczna wielogłowicowa uwaga (**MHA**),
   - *Reformer-style:* uwaga oparta na **LSH** (hashowane okna sekwencji),
   - *Performer:* aproksymacja softmax poprzez **FAVOR+** (cechy losowe).

   Wszystkie warianty korzystają z identycznych hiperparametrów oraz kodowania pozycji RoPE.

2. **Etap II: Pretraining na korpusie ogólnym (Wikipedia)**
   Modele są trenowane od zera na korpusie Wikipedia w zadaniu MLM.

   - Dynamiczne maskowanie tokenów.
   - Zapis checkpointu: `base_rope.pt`.
   - Reset optymalizatora i scheduler'a po zakończeniu etapu.

3. **Etap III: Adaptacja domenowa (DAPT/TAPT)**
   Dla każdego zbioru danych (IMDB, AG News, ArXiv Classification):

   - trening MLM startujący z checkpointu `base_rope.pt`,
   - ponowna inicjalizacja optymalizatora i scheduler'a,
   - możliwość dodania niewielkiej części próbek ogólnych (1–5%) dla ograniczenia *catastrophic forgetting*,

   Wynikiem są osobne checkpointy `base_rope + D_i.pt` dla każdego zbioru domenowego.

4. **Etap IV: Fine-tuning (klasyfikacja)**
   Trening nadzorowany rozpoczyna się z odpowiedniego checkpointu TAPT:

   - trening całego modelu
   - strategia „miękkiego startu": 1–2 epoki z zamrożonymi embeddingami/dolnymi warstwami, następnie pełne odmrożenie.

5. **Etap V: Organizacja eksperymentów**

   - jeden wspólny checkpoint `base_rope.pt` → wiele gałęzi TAPT (per dataset),
   - fine-tuning dla każdego zbioru,
   - powtórzenie całego schematu dla trzech wariantów uwagi (`MHA`, `LSH`, `FAVOR`).

4 Dane i przygotowanie danych

4.1 Korpus Wikipedii do pretreningu MLM

Źródło. `wikimedia/wikipedia 20231101.en`, split `train`. Wybrano podzbiór ok. 200 000 artykułów.

Filtracja. Wybierane dokumenty zawierające przynajmniej jedno ze słów kluczowych:

```
["film", "sport", "business", "science", "technology", "news"]
```

4.2 TAPT: IMDB, AG News i korpus ArXiv

- **IMDB:** recenzje filmowe bez etykiet (MLM); sekwencje o maksymalnej długości 512 tokenów. Dane pochodzą z modułu `datasets` (Hugging Face), split `train`.
- **AG News:** tytuły i streszczenia artykułów prasowych; trening MLM na sekwencjach do 512 tokenów; dane z `datasets` (Hugging Face), split `train`.
- **ArXiv Classification:** Zbiór zawiera streszczenia i teksty naukowe z różnych dziedzin (Algebra przemienna, Widzenie komputerowe i rozpoznawanie obrazów, Sztuczna inteligencja, Systemy i sterowanie, Teoria grup, Inżynieria obliczeniowa (finanse i nauka obliczeniowa), Języki programowania, Teoria informacji, Struktury danych i algorytmy, Obliczenia neuronowe i ewolucyjne, Teoria statystyki), a każdy dokument jest przypisany do jednej z jedenastu kategorii tematycznych ArXiv.

4.3 Fine-tuning: klasyfikacja

- **IMDB:** klasyfikacja binarna sentymentu (pozytywny/negatywny).
- **AG News:** klasyfikacja czteroklasowa (World, Sports, Business, Sci/Tech).
- **ArXiv Classification:** klasyfikacja streszczeń naukowych według dziedziny.

4.4 Aspekty danych: źródła, licencje, przetwarzanie, etyka i bias

Źródła i licencje. Wszystkie zbiory danych są pozyskiwane wyłącznie poprzez **Hugging Face Datasets**: `imdb`, `ag_news`, `ccdv/arxiv-classification`, `wikimedia/wikipedia`.

- **Wikipedia (`wikimedia/wikipedia`)**: treści objęte licencjami **CC BY-SA 3.0** oraz **GFDL**. Wykorzystanie w projekcie ma charakter badawczy; zachowujemy atrybucję i nie redystrybuujemy surowych fragmentów tekstów poza cytowaniami i metadanymi.
- **ArXiv (`ccdv/arxiv-classification`)**: dane pochodzą z serwisu arXiv; obowiązują warunki arXiv dla treści (*non-exclusive license to distribute*). Wykorzystanie wyłącznie do celów naukowo-badawczych; nie publikujemy pełnych tekstów.
- **IMDB (`imdb`)**, **AG News (`ag_news`)**: korzystamy z gotowych ładowarek i podziałów udostępnianych na HF; zasady licencyjne zgodnie z kartami danych na Hugging Face. Dane używane są wyłącznie badawczo; bez redystrybucji surowych przykładów.

Wstępne przetwarzanie danych.

- **Normalizacja:** podstawowa normalizacja z pipeline tokenizera (sekcja 2.2): m.in. obróbka białych znaków, znaków niewidocznych; bez agresywnej modyfikacji semantyki.
- **Ograniczenie długości:** maksymalna długość sekwencji określone w sekcji 2.2; dynamiczny *padding* w batchu.
- **Wikipedia:** dodatkowy filtr słów kluczowych (`film`, `sport`, `business`, `science`, `technology`, `news`).

Kwestie etyczne i bias.

- **Stronniczość treści:** Wikipedia odzwierciedla preferencje społeczności redaktorów; IMDB zawiera subiektywne opinie; AG News obejmuje wybrane źródła medialne; ArXiv ma nierówny rozkład dziedzin.
- **Transparentność:** odwołujemy się do kart danych na HF i nie modyfikujemy etykiet/ splitów dostarczanych przez autorów.

5 Eksploracyjna analiza danych

Analiza eksploracyjna została przeprowadzona dla czterech korpusów: IMDB, AG News, ArXiv Classification oraz Wikipedia. Podziały zbiorów (`split`) są predefiniowane podczas ładowania z **Hugging Face**.

5.1 Model bazowy

Wszystkie eksperymenty korzystają z klasyfikatora bazowego opartego na wektorach TF–IDF i regresji logistycznej (*baseline*). Konfiguracja modelu przedstawia się następująco:

```python
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline

pipeline = Pipeline(
    steps=[
        (
            "tfidf",
            TfidfVectorizer(
                max_features=20000,
                ngram_range=(1, 2),
                min_df=2,
                max_df=0.9,
            ),
        ),
        (
            "clf",
            LogisticRegression(
                max_iter=1000,
                solver="lbfgs",
                multi_class="auto",
                verbose=0,
            ),
        ),
```

```
    ]
)
```

Uwaga: W dalszych eksperymentach wykorzystamy dodatkowy **baseline transformacyjny** w postaci **DistilBERT-base-uncased** (fine-tuning) w celu porównaniania wyników z powszechnie stosowanym punktem odniesienia.

5.2 Zbiór IMDB

Opis. Zbiór IMDB obejmuje 50 000 recenzji filmowych z etykietami `positive` lub `negative`, oraz dodatkowy podzbiór 50 000 przykładów bez etykiet (*unsupervised*).

Statystyki długości (liczba słów).

Split	avg	std	median	min	max
train	233.8	173.7	174.0	10	2470
test	228.5	168.9	172.0	4	2278
unsupervised	234.4	173.8	175.0	9	2367

Statystyki po tokenizacji (liczba tokenów).

Split	avg_tokens	std_tokens	median_tokens	min	max
train	313.9	234.3	233.0	13	3127
test	306.8	227.9	230.0	10	3157
unsupervised	314.8	234.5	234.0	13	3446

Rozkład etykiet (zbiór treningowy).

Etykieta	Liczba próbek	Udział [%]
negative	12 500	50.00
positive	12 500	50.00

Wyniki modelu bazowego (zbiór testowy).

Klasa	Precision	Recall	F1	Support
negative	0.90	0.89	0.89	12 500
positive	0.89	0.90	0.90	12 500
Accuracy		0.8950		25 000

5.3 Zbiór AG News

Opis. Zbiór AG News zawiera 127 600 artykułów wiadomości w czterech kategoriach: `world`, `sports`, `business`, `sci/tech`. Podział: 120 000 próbek uczących oraz 7 600 testowych.

Statystyki długości (liczba słów).

Split	avg	std	median	min	max
train	37.8	10.1	37.0	8	177
test	37.7	10.1	37.0	11	137

Statystyki po tokenizacji (liczba tokenów).

Split	avg_tokens	std_tokens	median_tokens	min	max
train	53.2	19.1	51.0	15	379
test	52.7	18.2	50.0	18	277

Rozkład etykiet (zbiór treningowy).

Etykieta	Liczba próbek	Udział [%]
business	30 000	25.00
sci/tech	30 000	25.00
sports	30 000	25.00
world	30 000	25.00

Wyniki modelu bazowego (zbiór testowy).

Klasa	Precision	Recall	F1	Support
world	0.93	0.90	0.92	1900
sports	0.95	0.98	0.97	1900
business	0.89	0.88	0.89	1900
sci/tech	0.89	0.90	0.90	1900
Accuracy		0.9166		7600

5.4 Zbiór ArXiv Classification

Opis. Zbiór ArXiv Classification obejmuje artykuły naukowe z serwisu arXiv w 11 kategoriach tematycznych. Podziały: `train` – 28 388, `validation` – 2 500, `test` – 2 500 próbek.
Statystyki długości (liczba słów).

Split	avg	std	median	min	max
train	10 560	10 184	8424	441	553 206
validation	10 506	8260	8431	733	125 454
test	10 234	7781	8198	740	95 584

Statystyki po tokenizacji (liczba tokenów).
Obliczone bez ***split=train****. Maksymalna długość ograniczona do 32768 tokenów*

Split	avg_tokens	std_tokens	median_tokens	min	max
validation	15 012.7	8385.6	12 885.0	1373	32 768
test	14 745.9	8257.8	12 631.0	1268	32 768

Rozkład etykiet (zbiór treningowy).

Etykieta	Liczba próbek	Udział [%]
cs.DS	3527	12.42
cs.IT	2768	9.75
cs.SY	2631	9.27
math.GR	2599	9.16
math.ST	2581	9.09
cs.AI	2569	9.05
cs.NE	2560	9.02
math.AC	2456	8.65
cs.PL	2443	8.61
cs.CV	2137	7.53
cs.CE	2117	7.46

Wyniki modelu bazowego (zbiór testowy).

Klasa	Precision	Recall	F1	Support
math.AC	0.95	0.94	0.95	212
cs.CV	0.80	0.82	0.81	194
cs.AI	0.69	0.58	0.63	205
cs.SY	0.85	0.88	0.87	233
math.GR	0.95	0.96	0.95	234
cs.CE	0.78	0.73	0.75	196
cs.PL	0.88	0.93	0.91	241
cs.IT	0.86	0.85	0.85	236
cs.DS	0.88	0.90	0.89	318
cs.NE	0.75	0.74	0.75	219
math.ST	0.81	0.88	0.84	212
Accuracy		0.8436		2500

5.5 Zbiór Wikipedia

Opis. Zbiór Wikipedia obejmuje pierwsze 200 000 artykułów zawierających co najmniej jedno ze słów kluczowych: `film`, `sport`, `business`, `science`, `technology`, `news`.

Statystyki długości (liczba słów).

avg	std	median	min	max
931.3	1351.6	547.0	7	48 059

Statystyki po tokenizacji (liczba tokenów).
Obliczone na podstawie pierwszych 20000 artykułów. Maksymalna długość ograniczona do 4096 tokenów

avg_tokens	std_tokens	median_tokens	min	max
1364.6	1226.1	897.0	16	4096