Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie

Wydział Informatyki

Praca dyplomowa

Generacja muzyki przy pomocy dużych modeli językowych

Music generation with Lange Language Models

Autor: **Filip Ręka**
Kierunek: **Informatyka — Data Science**
Opiekun pracy: **dr hab. Maciej Smołka prof. AGH**

Kraków, 2024

Tutaj możesz umieścić treść podziękowań. Tutaj możesz umieścić treść podziękowań. Tutaj możesz umieścić treść podziękowań. Tutaj możesz umieścić treść podziękowań. Tutaj możesz umieścić treść podziękowań.

Streszczenie

Duże modele językowe (ang. *Large Language Models* **LLM**) charakteryzują się zdolnością do generacji języka oraz innych zadań w przetwarzania języka naturalnego, takich jak na przykład klasyfikacja. Zdolność tą nabierają podczas czasochłonnego oraz intensywnego obliczeniowego treningu metodami samo odraz pół-nadzorowanego, podczas którego uczą się one relacji z wielkiej ilości dokumentów tekstowych. LLMy mogą zostać wykorzystane do generacji tekstu, formy generatywnej sztucznej inteligencji, poprzez pobieranie tekstu wejściowego i wielokrotne przewidywanie kolejnego tokenu lub słowa w tekście. Strukturę muzyki można porównać struktury tekstu pisanego, gdzie każda nuta odpowiada literze lub słowu, akordy zdaniom a dłuższe i sekwencję paragrafom. Poniższa praca, zamierza zbadać możliwości generacyjne LLMów wytrenowanych na muzycznych zbiorach danych.

Abstract

Abstract in English [1] …

Spis treści

5

Zawartość spisu treści — tytuły rozdziałów oraz ich liczba zależą od tematyki pracy — należy ustalić z opiekunem pracy.

Spis rysunków

Spis tabel

Lista algorytmów

Lista kodów źródłowych

1. Wstęp

Uwaga 1.1. Tytuł oraz strukturę rozdziału należy ustalić z opiekunem pracy.

Wprowadzenie w tematykę pracy.

1.1. Cel i zakres pracy

Streszczenie specyfikacji wymagań Promotora.

2. Część literaturowa

2.1. Cyfrowa reprezentacja muzyki

2.1.1. WAV (ang. *waveform audio format*)

2.1.2. MIDI (ang. *Musical Instrument Digital Interface*)

2.1.3. Podobieństwa reprezentacji muzyki oraz tekstu

2.1.4. Tokenizacja

[2]

2.2. Zbiory danych

2.2.1. Johann Sebastian Bach Chorales

Datset [3]

2.2.2. The MAESTRO v3.0

Dataset [4]

2.2.3. Million Song Dataset

Dataset i takie cytowanko [5]

2.3. STOA

Tutaj nie wiem do końca w jakiej kolejności chciałbym o tym pisać, ponieważ z jednej strony przedstawienie STOA przed czymkolwiek jest ok, ale nie chciałbym pisać o czymś czego jeszcze w pracy nie wprowadziłem.

2.4. Architekturze transformera

2.4.1. Algorytm uwagi (ang. *attention*)

2.4.2. Warianty mechanizmu uwagi

2.4.2.1. Self attention

2.4.2.2. Multi-headed attention

2.4.2.3. Flash attention

2.4.3. Budowa transformera

2.4.4. Modele tranformerowe

2.4.4.1. *Classic* transformer

2.4.4.2. SeqGAN

2.4.4.3. Mistral

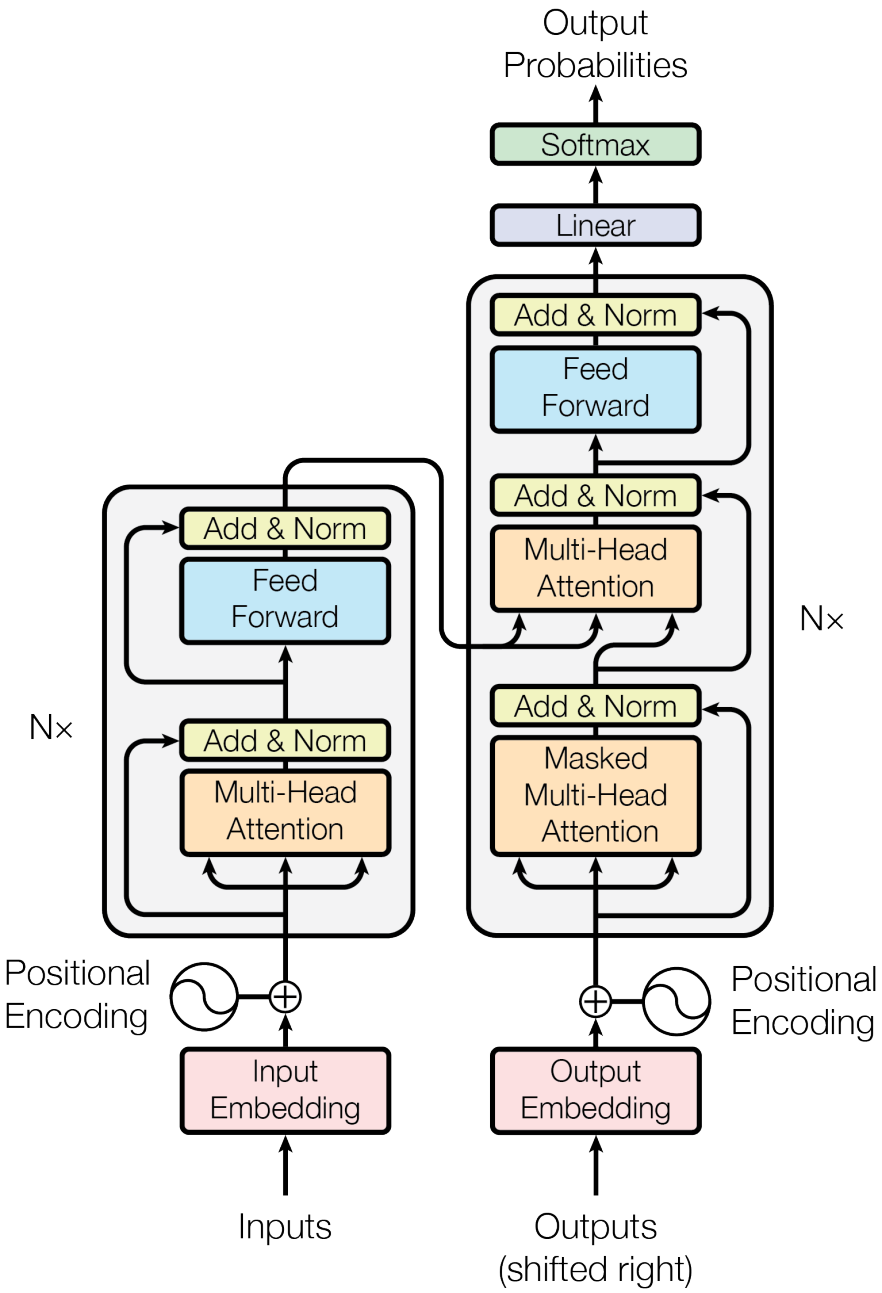
2.5. Architektura *state space*

2.5.1. Mamba

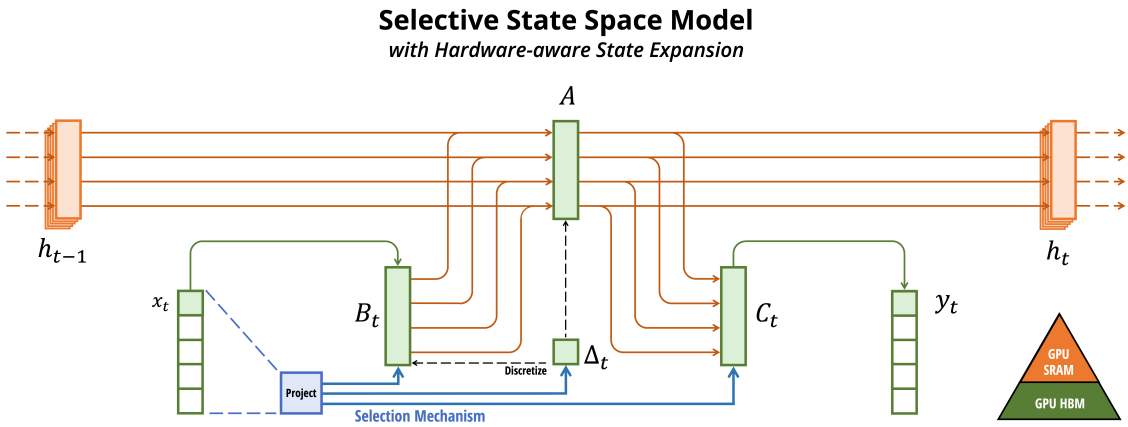
2.5.2. Tutaj się rozdrobnić trzeba

Uwaga 2.1. Tytuł oraz strukturę rozdziału należy ustalić z opiekunem pracy.

Aktualny stan wiedzy, na dany temat, na podstawie dostępnej literatury naukowej oraz specjalistycznej.



Rysunek 2.1.: Schemat transformera.



Rysunek 2.2.: Schemat modelu Mamba.

3. Część badawcza

Uwaga 3.1. Tytuł oraz strukturę rozdziału należy ustalić z opiekunem pracy.

3.1. Opis *pipeline-u*

Tutaj zamierzam opisać w jaki sposób modele zostały stworzone, jakie bilbioteki zostały użyte, jaki sprzęto został użyty podczas treningu

3.2. Porównanie architektur użytych modeli

3.3. Prezentacja otrzymanych wyników

3.4. Porównanie wyników

Celem porównania otrzymanych wyników z dowolnych modeli, jest ocena jakości ich pracy. W kontekście modeli dyskryminujących (ang. *discriminative model*), na przykład klasyfikatorów bądź modeli regresyjnych, ocena ich jakości jest dość prosta, ponieważ istnieje predefiniowana, prawdziwa wartość w zbiorze testowym, którą model próbuje przewidzieć. W takim przypadku ewaluacja jakości modelu to porównanie prawdziwych danych, z tymi przewidzianymi przez model. Porównanie odbywa się przez policzenie metryk np. dla modelu klasyfikującego celności, precyzji, *F1-score* lub dla modelu regresyjnego MSE lub R^2. W przypadku modeli generacyjnych celem treningu jest jak najlepsza aproksymacja rozkładu prawdopodobieństwa danych $P(X_{data})$ lub w przypadku danych oznaczonych łączny rozkład $P(X, Y)$. W przypadku dużej wymiarowości danych, obliczenie obiektywnych metryk takich jak *log-likelihood* lub dywergencja Kullbacka-Leiblera (*KLD*) często jest nieobliczalne. Dla człowieka weryfikacja wyników modeli generacyjnych takich jak *text-to-speech* lub *text-to-image* jest trywialnym zadaniem, jednak nie jest to oczywiste zadanie algorytmiczne. Często odwołuje się do subiektywnych metryk takich jak *MOS ang. mean opinion score*, które oblicza się jako średnią opinię np. w skali od 1-5 braną z zazwyczaj niewielkiej grupy ludzi. Niestety metryka ta często nie jest wiarygodna ze względu na jej subiektywności niewielką

próbę badawczą. Aby wyeliminować te wady serwisy takie jak *HuggingFace* udostępniają narzędzia dla członków społeczności, które pozwalają na ranking modeli, z nadzieją że zgodnie z prawem wielkich liczb, przy wystarczającej liczbie odpowiedzi, uda się otrzymać w miarę obiektywną ocenę. Przykładowym narzędziem tego rodzaju jest *The TTS Arena*[6], która pozwala na ranking modeli *text-to-speech*.

W przypadku muzyki, istnieje możliwość aby zweryfikować poprawność wygenerowanych sekwencji odwołując się do teorii oraz harmonii muzyki. Istnieje kilka narzędzi takich jak *Chordify*, *Hooktheory* lub *Sibelius*, które pozwalają na analizę harmoniczną utworów, dzieki czemu autorzy mogą w prosty sposób analizować i dobierać progresję danej melodii. Niestety większość takich narzędzi jest płatna i nie pozwala na zautomatyzowaną analizę wielu plików. Dodatkowym problemem jest w analizie harmonicznej, szczególnie prowadzonej przez algorytmy, jest rozróżnienie harmonii wertykalnej oraz horyzontalnej. Rozróżnienie to zostało wytłumaczone przez Jacoba Colliera, kilkukrotnego laureata nagród *Grammy*, którego zdaniem akord, który nawet zagrany sam brzmi niepoprawnie, w odpowiednim kontekście i przez odpowiednią progresję w kolejnych fragmentach muzyki, może mieć nadany sens, przez co cała sekwencja nabiera muzycznego piękna[7].

Uwaga 3.2. W mojej pracy prawdopodobnie zostanie zastosowane podejście MOS dla dość niewielkiej grupy ludzi, jednak jeśli triale oprogramowania pozwola, spróbuję przynajmniej sprawdzić czy taka analiza pozwala na jakieś sensowne obliczenie metryki

4. Zakończenie

Uwaga 4.1. Tytuł oraz strukturę rozdziału należy ustalić z opiekunem pracy.

1. Podsumowanie.
2. Możliwości dalszego rozwoju.
3. Potencjalne obszary zastosowania pracy.

Dodatek A.

Typowe elementy składowe pracy dyplomowej z informatyki

A.1. Tabele

Uwaga A.1.

- Każda tabela powinna być opisana w treści pracy.
- Podpis ma być przed tabelą.

W tabeli A.1 przedstawiono wyniki pomiarów.

Tabela A.1.: Pomiary zużycia energii elektrycznej.

L.p.	**Wartość**
1	12345,6789
	45,89
2	45,678901

Jeżeli tabela zawiera dużą liczbę wierszy i może nie zmieścić się na stronie — patrz tabela A.2 — skorzystaj z pakietu *longtable* [8].

Tabela A.2.: Tabela, która zawiera dużą liczbę wierszy.

	1	2	3	4	5	6	7	8	
Student 1									

	1	2	3	4	5	6	7	8	
Student 2									
Student 3									
Student 4									
Student 5									
Student 6									
Student 7									
Student 8									
Student 9									

Tabele, w których występuje długi tekst, a co za tym idzie może się on nie zmieścić — musi zostać zawinięty, z pomocą przychodzi środowisko 'tabularx' [9] — patrz tabela A.3.

Tabela A.3.: Tabela zawierająca długi tekst.

Wpis wielokolumnowy!		TRZY	CZTERY
jeden	Szerokość tej kolumny zależy od szerokości tabeli.	trzy	Kolumna czwarta będzie zachowywać się w taki sam sposób jak druga kolumna o tej samej szerokości.

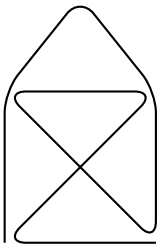
A.2. Rysunki

Uwaga A.2.

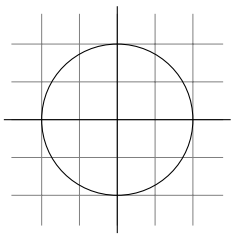
- Rysunki powinny być przerysowane samodzielnie albo używane tylko te, których twórcy zezwolili na ich rozpowszechnianie oraz kopiowanie, czyli np. rysunki objęte licencją Creative Commons.
- Każdy rysunek powinien być opisany w treści pracy.

A.2.1. Wewnętrzne

Klasa *agh-wi*, automatycznie, dołącza pakiet *TikZ* [10] — dostarcza on komend pozwalających na tworzenie grafik. Przykładowe grafiki pokazano na rysunku A.1 oraz A.2.



Rysunek A.1.: Prosty rysunek *TikZ*.



Rysunek A.2.: Bardziej złożony rysunek *TikZ*.

Oprócz rysunków eksponowanych możliwe jest tworzenie grafik będących częścią zdania.

TikZ pozwala również na kreślenie po powierzchni strony, np. możemy narysować strzałki pomiędzy elementami strony.

Ułamek — patrz wzór A.1 — składa się z:

- Licznika
- Mianownika

$$a = \frac{x + y}{y - z} \tag{A.1}$$

A.2.2. Zewnętrzne

Oczywiście możliwe jest również dołączanie rysunków zewnętrznych — pakiet *graphicx* [11] pozwala na wstawianie grafik zapisanych w plikach: '.png', '.jpg' oraz '.pdf'. Rysunek A.3 wstawiono przy użyciu tego pakietu.

Rysunek A.3.: Logo Wydziału Informatyki.

A.3. Kody źródłowe

Najpopularniejszymi pakietami, które umożliwiają składanie kodów źródłowych programów, są:

listings [12] — kod źródłowy jest formatowany bezpośrednio przez LaTeX-a — nie jest używany żaden, zewnętrzny, formater kodu.

Kod źródłowy A.1: Przykładowy kod źródłowy sformatowany za pomocą pakietu 'listings'.

```
1 /* Pierwszy program w C++ */
2
3 #include <iostream>
4
5 int main() {
6 std::cout << "Hello World!";
7 return 0;
8 }
```

minted [13] — formatuje kod źródłowy przy użyciu biblioteki języka Python o nazwie *Pygments* [14].

Kod źródłowy A.1.: Przykładowy listing sformatowany za pomocą pakietu 'minted'.

```
1 /* Pierwszy program w C++ */
2
3 #include <iostream>
4
5 int main() {
6 std::cout << "Hello World!";
7 return 0;
8 }
```

Uwaga A.3.

- Podpis ma być przed kodem źródłowym.
- Proszę używać tylko jednego z tych pakietów; w przeciwnym razie otrzymasz taki efekt, jak w przykładowej pracy — obydwa listingi mają ten sam numer.

Kod źródłowy w C++ sformatowany przy użyciu pakietu *listings*, pokazano na listingu A.1; sformatowany przy użyciu pakietu *minted*, pokazano na listingu A.1.

A.4. Algorytmy

Pakiet *algorithm2e* [15] to jeden z kilku, które pozwalają zapisywać algorytmy w formie pseudokodu — patrz algorytm 1.

Uwaga A.4. Podpis ma być przed algorytmem.

Algorytm 1: Disjoint decomposition.

input : A bitmap Im of size $w \times l$
output: A partition of the bitmap

```
1  special treatment of the first line;
2  for i ← 2 to l do
3  |  special treatment of the first element of line i;
4  |  for j ← 2 to w do
5  |  |  left ← FindCompress(Im[i, j − 1]);
6  |  |  up ← FindCompress(Im[i − 1, ]);
7  |  |  this ← FindCompress(Im[i, j]);
8  |  |  if left compatible with this then // O(left,this)==1
9  |  |  |  if left < this then Union(left,this);
10 |  |  |  else Union(this,left);
11 |  |  end
12 |  |  if up compatible with this then                    // O(up,this)==1
13 |  |  |  if up < this then Union(up,this);
   |  |  |  // this is put under up to keep tree as flat as possible
14 |  |  |  else Union(this,up);
   |  |  |  // this linked to up
15 |  |  end
16 |  end
17 |  foreach element e of the line i do FindCompress(p);
18 end
```

A.5. Wzory

LaTeX bardzo dobrze sprawdza się w przypadku prac dyplomowych zawierających wzory matematyczne[1].

[1] W przypadku złożonych wzorów warto zastosować pakiet *amsmath* [16].

A.5.1. Przykłady

Wzór $E = mc^2$ jest częścią zdania.

$$\left|\sum_{i=1}^{n} a_i b_i\right| \leq \left(\sum_{i=1}^{n} a_i^2\right)^{1/2} \left(\sum_{i=1}^{n} b_i^2\right)^{1/2} \tag{A.2}$$

Wartości zmiennej opisano wzorem A.3.

$$x = \begin{cases} y & \text{dla } y > 0 \\ \frac{z}{y} & \text{dla } y \leq 0 \end{cases} \tag{A.3}$$

Wzór A.4 to wzór wielowierszowy.

$$\begin{aligned} 2x^2 + 3(x-1)(x-2) &= 2x^2 + 3(x^2 - 3x + 2) \\ &= 2x^2 + 3x^2 - 9x + 6 \\ &= 5x^2 - 9x + 6 \end{aligned} \tag{A.4}$$

Uwaga A.5. Należy używać tylko dwóch rodzajów wzorów:

1. „W linii”.
2. Eksponowane, numerowane.

A.6. Twierdzenia i podobne struktury

Twierdzenie nr 1 opublikował, w roku 1691, francuski matematyk Michel Rolle.

Twierdzenie 1 (Rolle’a) *Jeśli dana funkcja f:* $\mathbb{R} \to \mathbb{R}$ *jest:*

1. *ciągła w przedziale* $[a, b]$
2. *jest różniczkowalna w przedziale* (a, b)
3. *na końcach przedziału* $[a, b]$ *przyjmuje równe wartości:* $f(a) = f(b)$,

to w przedziale (a, b) *istnieje co najmniej jeden punkt* c *taki, że* $f'(c) = 0$.

Teraz coś z informatyki …

Definicja 1 *Bit to najmniejsza jednostka informacji w komputerze.*

Definicja 2 *Bajtem nazywamy ciąg ośmiu bitów.*

Uwagi Autora

- Aktualna wersja klasy jest dostępna pod adresem https://github.com/polaksta/LaTeX/tree/master/agh-wi[1].
- Skoro Twoja praca dyplomowa powstała w LaTeXu, to zachęcam Cię również do przygotowania prezentacji (na obronę pracy magisterskiej) w tym języku. Najpopularniejszą klasą do tworzenia tego typu dokumentów jest *beamer* [17].
- Pod adresem https://github.com/polaksta/LaTeX/tree/master/beamerthemeAGH[2] możesz znaleźć, stworzony przeze mnie, nasz uczelniany szablon dla prezentacji LaTeX Beamer.
- Treść wszystkich rozdziałów tej, przykładowej, pracy dyplomowej znajduje się w jednym pliku — **nie jest to polecane rozwiązanie**. W przypadku pisania własnej pracy warto umieścić zawartość każdego z rozdziałów w osobnych plikach, a następnie dołączać je do dokumentu głównego — patrz opis na stronie https://www.dickimaw-books.com/latex/thesis/html/include.html.
- Jeżeli pewne elementy mają być wyróżniane w jednakowy sposób, to proponuję nie używać bezpośredniego stylowania, tzn.

```
\colorbox{red!50}{jednakowy} \colorbox{red!50}{sposób}
```

  ale zdefiniować własną komendę stylującą, np. `\alert`,

```
\newcommand{\alert}[1]{\colorbox{red!50}{#1}}
```

  a następnie użyć jej w dokumencie.

```
\alert{jednakowy} \alert{sposób}
```

  Dzięki temu, jeżeli będziesz chciał / chciała zmienić sposób stylowania tych elementów, np. niebieskie tło zamiast czerwonego, to wystarczy zmodyfikować, tylko, definicję komendy, zamiast zastępować, w tekście pracy dyplomowej, wybrane (niekoniecznie wszystkie!) wystąpienia tekstu `red`, tekstem `blue`.

[1] W przypadku Overleaf-a jest ona pod adresem https://www.overleaf.com/read/fnvcvqjyrbyw#5ac622
[2] W przypadku Overleaf-a jest on pod adresem https://www.overleaf.com/read/fkjdthnbrfhj#9c6184

Stanisław Polak

Bibliografia

[1] Ashish Vaswani i in. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL].

[2] Nathan Fradet i in. „MidiTok: A Python package for MIDI file tokenization". W: *Extended Abstracts for the Late-Breaking Demo Session of the 22nd International Society for Music Information Retrieval Conference*. 2021. URL: https://archives.ismir.net/ismir2021/latebreaking/000005.pdf.

[3] Darrell Conklin. *Bach Chorales*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5G

[4] Curtis Hawthorne i in. „Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset". W: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=r1lYRjC9F7.

[5] Thierry Bertin-Mahieux i in. „The Million Song Dataset". W: *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*. 2011.

[6] *The TTS Arena*. URL: https://huggingface.co/blog/arena-tts.

[7] Jacob Collier. *That's not a wrong note, you just lack confidence*. URL: https://www.youtube.com/watch?v=meha_FCcHbo.

[8] *The longtable package*. URL: http://mirrors.ctan.org/macros/latex/required/tools/longtable.pdf.

[9] *The tabularx package*. URL: http://mirrors.ctan.org/macros/latex/required/tools/tabularx.pdf.

[10] *The TikZ and PGF Packages*. URL: http://mirrors.ctan.org/graphics/pgf/base/doc/pgfmanual.pdf.

[11] *Packages in the 'graphics' bundle*. URL: http://mirrors.ctan.org/macros/latex/required/graphics/grfguide.pdf.

[12] *The Listings Package*. URL: http://mirrors.ctan.org/macros/latex/contrib/listings/listings.pdf.

[13] *The minted package: Highlighted source code in LaTeX*. URL: http://mirrors.ctan.org/macros/latex/contrib/minted/minted.pdf.

[14] *Strona WWW biblioteki „Pygments"*. URL: https://pygments.org/.

[15] *algorithm2e.sty — package for algorithms*. URL: http://mirrors.ctan.org/macros/latex/contrib/algorithm2e/doc/algorithm2e.pdf.

[16] *User's Guide for the amsmath Package*. URL: http://mirrors.ctan.org/macros/latex/required/amsmath/amsldoc.pdf.

[17] *The beamer class*. URL: http://mirrors.ctan.org/macros/latex/contrib/beamer/doc/beameruserguide.pdf.