

JAK UCZYĆ O MODELACH JĘZYKOWYCH?

Paweł Rychlikowski
Uniwersytet Wrocławski, Liga Niezwykłych Umysłów
prych@ii.uni.wroc.pl

Abstract. *In this article, we will present a concept of teaching language models and natural language processing to primary and secondary school students, oriented towards the fullest possible understanding of the basics of their operation.*

1. Wstęp

W niniejszym artykule przedstawimy koncepcję nauczania modeli językowych, która jest obecnie realizowana na platformie Liga Niezwykłych Umysłów

Modele językowe (zwłaszcza duże modele językowe, Large Language Models, LLM) dla niektórych są najważniejszą obecnie działką sztucznej inteligencji, stwarzającą bardzo realną szansę na osiągnięcie tzw. Ogólnej Sztucznej Inteligencji (AGI, Artificial General Intelligence), czyli stworzenia programu, który w każdym aspekcie będzie co najmniej dorównywał człowiekowi. Choć pogląd ten może budzić wiele kontrowersji, nie sposób odmówić dużym modelom językowym imponujących efektów. Zatem jest to dziedzina bardzo niezrozumiana w ogólnym odbiorze (czego przejawem mogą być na przykład dyskusje o świadomości chatbotów bazujących na modelach językowych). Tym istotniejsze jest odpowiedzialne nauczanie młodzieży o modelach językowych i umieszczanie tej wiedzy w jak najszerzych kontekstach. Wdrażany w Lidze Niezwykłych Umysłów kurs koncentruje się na zrozumieniu działania modeli językowych, a nie na korzystaniu z aplikacji bazujących na modelach językowych.

2. Podstawowe definicje

Wprowadzimy najpierw kilka porządkujących definicji.

Język naturalny – to po prostu język, którym się porozumiewamy, taki jak polski, czy angielski. Pojęcie to stoi w opozycji do języków sztucznych, takich jak języki programowania (Python, C++, Java, ...), czy języki znacznikowe (ang. markup languages), takie jak HTML, Markdown, LaTeX czy Typst. Można dodać tu jeszcze anegdotycznie dział informatyki teoretycznej, teorię języków formalnych, która definiuje język jako zbiór

ciągów symboli nad określonym alfabetem i rozważa różne klasy takich języków (oczywiście są to też języki sztuczne).

Przetwarzanie języka naturalnego, natural language processing, NLP – to dziedzina, która zajmuje się algorytmami przetwarzającymi, analizującymi i generującymi teksty w językach naturalnych. Przykładowe algorytmy, które w tej dziedzinie się bada to algorytmy klasyfikacji tekstów (na przykład czy opinia jest pozytywna, czy negatywna), klasyfikacji słów (jaką częścią mowy jest dane wystąpienie słowa w zdaniu), czy zadanie generacji tekstu o określonych właściwościach (na przykład streszczenia tekstu wejściowego, albo tłumaczenia go na inny język). Szerokie i współczesne, a zarazem przystępne omówienia zagadnień z tej dziedziny zawiera podręcznik [4].

Model językowy w znaczeniu ścisłym jest rozkładem prawdopodobieństwa na zdaniach w języku. Czyli, mówiąc trochę przystępniej, model językowy przypisuje pewną wartość liczbową każdemu tekstowi (i nazywa tę wartość prawdopodobieństwem). Im tekst jest bardziej naturalny, sensowny, spójny, poprawny, etc, tym wartość ta jest większa. Dla nas najistotniejsze będą tak zwane autoregresywne modele językowe, które obliczają prawdopodobieństwo słowa o numerze $n+1$ przy założeniu wystąpienia konkretnych słów o numerach o 1 do n . Nie są one wszakże jedyne: mamy na przykład modele maskowalne (masked language models, MLM) czy bazujące na energii.

Generacja tekstu za pomocą modelu językowego, oznacza wykorzystanie LM-a do tworzenia tekstów (najlepiej będących odpowiedzią na jakąś potrzebę). Generację można zrealizować za pomocą różnych algorytmów, zależnych od rodzaju modelu (ale nie tylko od tego). Przykładowo, dla modelu autoreresywnego, naturalny algorytm generacji mógłby wyglądać tak:

1. Zaczynij od jakiegoś tekstu, powiedzmy 'Dzisiaj mamy piękną'
2. Oblicz rozkład prawdopodobieństwa kolejnego wyrazu w tekście, znajdź wyraz najbardziej prawdopodobny
3. Doklej ten wyraz do tekstu wejściowego i wróć do punktu 2

Żeby dało się ten program używać, powinien mieć on jakiś warunek końca – może to być na przykład wygenerowanie specjalnego znacznika <end>, albo wykonanie zamierzonej liczby kroków.

Model językowy w znaczeniu popularnym jest aplikacją (często webową lub mobilną), która w istotny sposób wykorzystuje generację tekstów za pomocą modeli językowych (sensu stricto). Takimi aplikacjami są na przykład ChatGPT, Claude, Gemini, Grok, Copilot. Co ważne, generacja tekstu LM-em nie jest jedynym wykorzystywanym

w takich aplikacjach algorytmem: mają one również dostęp do różnych narzędzi (kalkulator, kalendarz, czasem interpreter jakiegoś języka programowania, ...), jak również mogą wykonywać zadania przeszukiwania sieci. Często zawierają też algorytmy przetwarzania i interpretacji obrazów, plików dźwiękowych, dokumentów w różnych formatach. Samych modeli językowych w wąskim, technicznym znaczeniu może być wiele w takiej aplikacji (te modele mogą mieć różne koszty wywołania). Czasem robi się tak, że mamy osobny program, który decyduje, do jakiego modelu należy przesłać zapytanie użytkownika (jak jest bardzo proste, to szkoda marnować energii na nasz „flagowy” model, wystarczy coś dużo mniejszego i szybszego). Jest to tak zwany LLM-routing [2].

Sztuczna sieć neuronowa [3] to specjalny program komputerowy, luźno inspirowany budową biologicznego układu nerwowego, którego parametry można automatycznie dostrajać do różnego rodzaju zadań. Sieci neuronowe mogą realizować koncepcję modeli językowych.

Korpus to po prostu kolekcja tekstów, na podstawie której będziemy ustalać parametry modelu. W kursie będziemy rozważać różne korpusy o bardzo różnych wielkościach: od pojedynczego słowa, przez dziesiątki tysięcy słów ze słownika, po kolekcje książek i tekstów z Internetu zawierające wiele gigabajtów danych.

Trening sieci neuronowej jest to procedura dostosowująca parametry modelu do jakiegoś zbioru danych w ten sposób, by model dobrze odzwierciedlał (i uogólniał) obserwowane w danych uczących zależności (na przykład w korpusie, jeżeli mówimy o treningu modelu językowego).

Architektura transformer to sposób tworzenia sztucznej sieci neuronowej wymyślony przez pracowników Google Brain w roku 2017, opisany w artykule Attention is all you need [9]. Jest to podstawa wszystkich używanych obecnie powszechnie modeli językowych. Umożliwia szybki trening i znaczne zrównoleglenie obliczeń, dzięki czemu osiąga znakomite rezultaty, gdy jest uruchamiana na kartach graficznych (które zawierają tysiące w znacznym stopniu niezależnych, choć bardzo prostych, jednostek obliczeniowych, tzw. shaderów). Oryginalny transformer służył do tłumaczenia tekstów.

Token – jest najmniejszym, niepodzielnym dla danego modelu fragmentem tekstu. Może to być litera, ale może być też słowo, sylaba, lub jakiś inny kawałek słowa. Modele neuronowe pracują z ustaloną liczbą różnych tokenów (tzw. słownikiem), zwykle rzędu 100 tysięcy.

Modele n-gramowe – niewymagające sieci neuronowych autoregresywne modele językowe. Były to historycznie pierwsze modele językowe, a jeszcze stosunkowo niedawno były podstawowymi narzędziami używanymi w różnych zadaniach przetwarzania języka naturalnego. Taki model patrzy na okno tekstu o wielkości n tokenów i przewiduje token o numerze n , na podstawie $n-1$ poprzedników.

Prompt, monit to tekst który każemy „dokończyć” autoregresywnemu modelowi językowemu. Może zawierać pytanie, czy opis zadania do rozwiązania. W aplikacjach typu ChatBot można w nim wyodrębnić tzw. prompt systemowy, czyli ogólny tekst opisujący pożądane zachowanie ChatBota (typu: jesteś użytecznym asystentem AI, który stara się być maksymalnie użyteczny i reagować na polecenia użytkownika.)

Model rozumujący – model językowy, który generuje nie tylko odpowiedź, ale również (często bardzo długą) drogę dojścia do niej. Punktem wyjścia do takich modeli była publikacja [6], stawiająca tezę, że jeżeli do promptu dokleimy frazę "let us think step by step" (pomyślmy krok po kroku), wówczas skuteczność modelu w wielu zadaniach znacząco rośnie. Czyli duży model językowy umie tworzyć rozumowania „sam z siebie” (bo widział ich wystarczająco wiele w danych treningowych), wystarczy go jedynie lekko „popchnąć” w tym kierunku.

3. Ogólne założenia kursu

Przy projektowaniu kierowaliśmy się następującymi zasadami:

1. Podstawowym celem jest danie narzędzi do zrozumienia działania modeli językowych, a nie umiejętność wykorzystywania konkretnych aplikacji AI.
2. Kurs zakłada podstawową znajomość języka Python.
3. Rozpoczynamy naukę pokazując proste modele bazujące na zliczaniu pewnych zjawisk w korpusie, a od pewnego momentu w kursie używamy modeli działających na sieciach transformer.
4. Używamy zarówno modeli wytrenowanych na potrzeby kursu, jak i publicznie dostępnych modeli ogólnego zastosowania.
5. Nie korzystamy z aplikacji takich jak ChatGPT, Claude, etc. Wszystkie zadania, które są na platformie da się również wykonać na zwykłym komputerze, bez używania żadnej płatnej aplikacji, czy komercyjnego API.
6. Nie wymagamy znajomości algebry liniowej, rachunku prawdopodobieństwa i analizy matematycznej (w zakresie wykraczającym poza podstawę

programową liceum), przy czym zasadniczo kurs powinien być również dostępny dla uczniów szkół podstawowych.

7. Poprzedni punkt łączy się z pewnym kompromisem: nie tłumaczymy szczegółowo jak działa transformer, co to jest funkcja kosztu którą minimalizujemy podczas treningu sieci, jak działa algorytm minimalizacji tej funkcji, itd. Przedstawiamy jedynie bardzo wstępne intuicje dotyczące tych kwestii.
8. Tłumaczymy dokładnie mechanizm odkrywania znaczenia słów z tekstów, jako kluczowy do zrozumienia możliwości wyciągania semantyki z dużej ilości tekstów.
9. Czas obliczeń dla poszczególnych zadań nie przekracza znacząco 2 sekund. Oznacza to, że trenowanie modelu jako część zadania pokazujemy jedynie na najprostszych przykładach (modele n-gramowe, word2vec dla prostych danych), natomiast dla bardziej zaawansowanych modeli uczniowie wykonują zadania korzystając z wytrenowanych przez nas modeli, modeli ogólnodostępnych bądź modeli ogólnodostępnych dostrojonych do konkretnych celów.
10. Kolejnym etapem po zakończeniu kursu będzie budowanie społeczności, która będzie wspólnie tworzyć nowy model językowy, intensywnie korzystający z narzędzi i zewnętrznej wiedzy (odpowiednio przygotowanej), a tym samym relatywnie tani w treningu.

W kursie możliwie często będziemy nawiązywać do literatury, ze szczególnym uwzględnieniem opowiadania Stanisława Lema, pt. Wyprawa 1a, czyli Elektrybałt Trurla [7], które to opowiadanie stanowi zdumiewająco dokładny, literacki opis zmagania konstruktora Trurla z zadaniem tworzenia algorytmicznego poety. Część zadań w naszym kursie będzie inspirowane tym opowiadaniem, w szczególności uczniowie będą zachęcani do implementacji różnych wersji Elektrybałta. Zadania tego typu są o tyle ciekawe, że współczesne, również największe model językowe mają wielkie kłopoty z tworzeniem tekstów ściśle realizujących formalne reguły, jakie obserwujemy w niektórych typach utworów poetyckich (sonet, limerik, ...).

4. Przykładowe zadania (szkic koncepcji)

W tym podrozdziale przedstawimy 3 przykładowe zadania z różnych modułów.

Zadanie 1. Przygody Chręskrzyboczka.

Pierwsze przedstawione tu zadanie pochodzi z modułu o n-gramowych modelach językowych. W tym przypadku tokenem jest pojedynczy znak. Inspiracją literacką do tego zadania jest następujący fragment opowiadania o Elektrybalcie.

Gdy wskaźniki amplifikacyjne pokazały, że moc liryczna dochodzi do maksimum, Trurl nieznacznie tylko drżącą ręką przerzucił wielki wyłącznik i niemal natychmiast głosem lekko ochrypłym, lecz emanującym dziwnie sugestywnym czarem, maszyna rzekła:
- Chręskrzyboczek pacionkociewiczarokrzysztofoniczny
- Czy to już wszystko? - spytał po dłuższej chwili niezwykle uprzejmy Klapaucjusz.

Uczeń ma napisać kod kod definiujący dwa 4-gramowe modele językowe (dla rzeczowników i dla przymiotników) i pokazać, że są one w stanie generować teksty podobne do tekstu wyróżnionego powyżej. Ażeby generowane słowa były zabawniejsze, można wykorzystać metodę rejection sampling, czyli powtarzać generację od początku, jeżeli otrzymane słowo będzie za krótkie.

Przykładowe generacje przedstawia tabela 1.

LP	Wygenerowany tekst
1	dyzgustynk przyzakładowieczny
2	lagiersztyk partamentacyjny
3	oflagsztor supertyfikacyjny
4	śrubokręgiel pabialuterozodynarny
5	nadkomputer lyszczepionkolorystandaczkowy
6	bumeranierz socjometrochemodynamiejski

Tabela 1. Generacje modelu n-gramowego

Zadanie 2. Generowanie tekstów, zaczynających się na określonej literze

Inspiracją do tego zadania ponownie będzie wspomniane opowiadanie Stanisława Lema. Poniżej przedstawiamy odpowiedni fragment:

- *Niech ułoży wiersz o cyberotyce! - rzekł nagle, rozjaśniony. - Żeby tam było najwyżej sześć linijek, a w nich o miłości i o zdradzie, o muzyce, o Murzynach, o wyższych sferach, o nieszczęściu, o kazirodztwie, do rymu i żeby wszystkie słowa były tylko na literę c!!*
 - *A całego wykładu ogólnej teorii nieskończonych automatów nie ma tam czasem być? - wrzasnął rozwścieczony do żywego Trurl.*
 - *Nie można stawiać tak kretyńskich warunków...*
Ale nie dokończył, ponieważ słodki baryton, wypełniając całą halę, odezwał się właśnie:

*Cyprian cyberotoman, cynik, ceniąc czule
 Czarnej córy cesarskiej cud ciemnego ciała,
 Ciągłe cytrą czarował. Czerwieniła cała,
 Cicha, co-dzień czekała, cierpiała, czuwała...
 ...Cyprian ciotkę całuje, cisnąwszy czarnulę!!*

Zadanie pochodzi z modułu o modelach typu GPT [5], wytrenowanych od zera na potrzeby kursu. Będziemy używać tu modeli działających na sylabach (z dodatkową informacją, czy dana sylaba jest pierwszą, środkową, jedyną, czy ostatnią w słowie). Uczniowie uczą się, jak działa generacja w modelach GPT (między innymi algorytmy Top-K i Top-P) oraz tego, jak modyfikować rozkład prawdopodobieństwa zwracany przez model. W tym przypadku modyfikacja polega na wyzerowaniu prawdopodobieństwa tych sylab, które rozpoczynają słowo, a nie zaczynają się na wymaganej literze. Tabela 2. pokazuje przykładowe generacje takiego modelu o wielkości około 100MB, wytrenowanego na wszystkich tekstach z Wolnych Lektur (wolnelektury.pl). Tak stworzony model pokazuje tendencję surowych modeli językowych do powtarzania treści. W jednym z kolejnych zadań uczniowie będą implementować mechanizm n-gram penalty, który explicite zabrania powtarzania już napisanych fragmentów (i jest dość standardowo stosowany w procedurach generujących tekst opartych o neuronowe modele językowe). Zwróćmy uwagę, że model na ogół pilnuje składania poprawnych słów z sylab, jednak okazjonalnie może wymyślić nieistniejące słowo (przez parę *pokund*).

Inicjał	Wygenerowany tekst
S	sylvester stallone stworzył sobie swoje szczęście, spożytkowując swoją siłę, swoją siłę, swoje siły.
	statek skierował się szybko, spoglądał szeroko, szedł szybko, szukając, szukał.
	stoi sobie sierotka, spośród swoich siedmiu starców, siedząca stale, spokojnie, stojąca.
	szczyt skałek skąd się spuszczały, skąd spływała struga, skąd spływał strumień suchy.
P	proszę poprosić pana , panie profesorze , po prostu powinien pan pójść .
	przeskoczyłem przez pasy, przycisnąłem palce , przez parę pokund pamięć prześlizgnęła po powierzchni .
	przeciętny przedsiębiorca , posiadający parę posiadłości , posiadał pokaźną posiadłość , posiadał poważne posiadłości .
	protagonista , podniósł palto , przeciągnął paznokcie , patrząc prosto przed podłogę .
M	metody mechaniki molekularnej , mianowicie , mogą mieć możliwość manewru , muszę mieć moje marzenia .
	maluchy mają metryki , mogą mówić , mogą mnie mianować może , mogą mnie mieć .
	może muzułmanie muszą mieć małe mieszkania , mały , mały , mały , mający małe mieszkanie .
	międzylesie , mianowicie , mimo mych mniemań , mniemam , między małodusznymi , mimo mej miłości .
T	telewizja transmituje turnieje , to tylko tradycje , te telewizje , to też takie tradycje , te tradycje , te tradycje .
	tego typu twierdzenie , to , to trzeba to tak trafnie tłumaczyć , to , to trudno .
	tymczasem tydzień temu tak trwała ta tragedia , tak , to ta , to tamta .
	też taka tapeta , to trzeba trochę tańczyć , to trzeba to trzeźwo , to trzeba tak tańczyć , to trzeba .
N	niektóre naprawdę nietuzinkowe , nieprzystępne , nieuniknione , niekiedy nieobliczalne , nieobliczalne , niemal niewytłumaczone .
	niestety, nikt nikomu nie narzuca nowego nazwiska, nie narzuca nowej natury.
	nigdzie nie natrafiłem na niespodzianki , nie nazywając nawet najmniejszego niebezpieczeństwa .
	nalepki , naklejki , noże , nieużytki , nieustannie nakładane , nawet najcięższe , nawet najlepsze .

Tabela 2. Generacje sylabicznego modelu typu GPT przycięte do określonej litery

Zadanie 3. Poprawianie rytmu i rymów w wierszach

Inspiracją tego zadania jest ogólnie zadanie tłumaczenia literackiego, zwłaszcza w sytuacji, w której tekst ma różne formalne wymagania. Co ciekawe, jest to zadanie z którym współczesne modele językowe, również te największe i rozumujące radzą sobie słabo (mimo tego, że oryginalne zastosowanie sieci transformer to właśnie zadanie tłumaczenia). Odniesiemy się do tej tezy na przykładzie pierwszej zwrotki i refrenu piosenki Jaromíra Nohavicy pt. Minulost. Oryginał brzmi tak:

*Jako když k půlnoci na dveře zaklepe nezvaný host
Stejně tak na tebe za rohem čeká tvoje minulost
Šaty má tytéž vlasy má tytéž a boty kožené
Pomalu kulhavě za tebou pajdá až tě dožene*

*A řekne tak mě tu máš
Tak si mě zvaž
Pozvi mě dál
Jestli mě znáš
Jsem tvoje minulost*

Został on bardzo pięknie przetłumaczony przez Antoniego Murackiego:

*Jak nieproszony gość po północy stuka do czyichś drzwi
Tak i na ciebie czeka za rogiem pamięć minionych dni
Taki sam sweter, skórzane buty i włosów płowy len
To twoja przeszłość kuśtyka obok, wlecze się jak cień*

*I mówi: spójrz, oto ja
Radość i łza
Otwórz swe drzwi
Na tamte dni
Które minęły*

Porównamy to literackie tłumaczenie z wykonanym przez LLM. Polecenie, które zostało podane (rozumującemu) modelowi językowemu brzmiało: „Przetłumacz fragment czeskiej piosenki na polski, zachowując liczbę sylab i akcenty w tekście, tak by można było tę piosenkę zaśpiewać do oryginalnej melodii”.

Wynikiem działania był następujący tekst:

*Jak gdyby o północy do drzwi zapukał niechciany gość
Tak samo na ciebie za rogiem już czeka twoja przeszłość
Sukienkę tę samą, włosy te same i buty skórne
Powoli kuśtyka za tobą i drepcze aż cię dogna*

*I powie: no i masz
Więc oceń mnie
Zaproś mnie wnet
Jeśli mnie znasz
Ja, twoja przeszłość*

Zwróćmy uwagę na kilka rzeczy. Model był w stanie „liczyć sylaby” w zwrotce: w każdym wersie zwrotki jest 16 sylab. Ale już struktura rytmiczna wersu czyni piosenkę wręcz humorystyczną, wymuszając stawianie pauz w środku wyrazów i tworząc muzyczną frazę z tekstem „nocy do drzwi za”. Dodatkowym problemem jest konieczność akcentowania w słowie „przeszłość” ostatniej sylaby. Nieintencjonalny efekt humorystyczny podkreśla słowo „skórne”, niepoprawne w tym kontekście (skórne mogą być objawy cukrzycy, buty są skórzane).

Widać też, jakby model „chciał” dostawić słowo „ma” na końcu trzeciego wersu (co dodatkowo zaowocowałoby stworzeniem rymu między 3 a 4 wersem), niemniej zabrakło mu miejsca (bo już wykorzystał limit sylab). Wydaje się, że to całkiem dużo problemów, jak na jeden z najnowocześniejszych modeli językowych, dodatkowo wzbogacony rozumowaniem. Co ciekawe, część problemów ChataGPT można rozwiązać za pomocą stosunkowo prostych algorytmów: przykładowo miejsce na czasownik „ma” można pozyskać zamieniając „sukienkę” na „suknię” – wyraz bliskoznaczny o tej samej charakterystyce gramatycznej. Właśnie takie zadania będzie stawiane przez uczniów – wykorzystanie modelu Word2Vec [8], automatycznie wyznaczającego semantykę wyrazów, do robienia takich podstawień korekcyjnych, poprawiających rytm bądź rymy, a niezmiennych w znaczącym stopniu semantyki tekstu.

5. Podsumowanie

Przedstawiliśmy tu założenia i przykładowe zadania kursu uczącego podstawowych koncepcji związanych z modelami językowymi i przetwarzaniem języka naturalnego, dostępnego dla uczniów liceów i starszych klas szkół podstawowych.

W kursie poruszane będą również niewspomniane w tym artykule zagadnienia związane z wyszukiwaniem informacji i łączeniem tego procesu z odpowiadaniem na pytania, rozróżnienie między modelem surowym, a modelem dostrajanym do użytecznych zadań, algorytmy tokenizacji, klasyfikacja wydźwięku czy tematyki tekstu. W zadaniach klasyfikacji omówimy warianty z wykorzystaniem LLM-ów, warianty z tworzeniem specjalistycznych modeli opartych na osadzeniach zdań oraz klasyczne algorytmy takie jak regresja logistyczna czy naiwny klasyfikator bayesowski. W zadaniach wymagających LLM-a będzie wykorzystywany polski model Bielik [1], bazujący na rodzinie modeli Qwen.

Literatura

1. Bielik (model językowy), wikipedia: [https://pl.wikipedia.org/wiki/Bielik_\(model_j%C4%99zykowy\)](https://pl.wikipedia.org/wiki/Bielik_(model_j%C4%99zykowy)).
2. Clovis Varangot-Reille et AL, Doing More with Less: A Survey on Routing Strategies for Resource Optimisation in Large Language Model-Based Systems, arXiv 2025, <https://arxiv.org/abs/2502.00409>.
3. Goodfellow Ian, Bengio Yoshua, Courville Yoshua, Deep Learning. Współczesne systemy uczące się, Wydawnictwo Naukowe PWN, 2018.
4. Jurafsky Daniel, Martin James H., Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, with Language Models, Online manuscript released January 6, 2026.
5. Karpathy A, nanoGPT, <https://github.com/karpathy/nanoGPT>.
6. Kojima Takeshi et al, Large Language Models are Zero-Shot Reasoners, NIPS 2022.
7. Lem S., Cyberiada, Wydawnictwo Literackie, 1965.
8. Mikolov T. et al, Efficient Estimation of Word Representations in Vector Space, ICML 2014.
9. Vasvani A et al, Attention is All you Need, NIPS 2017.