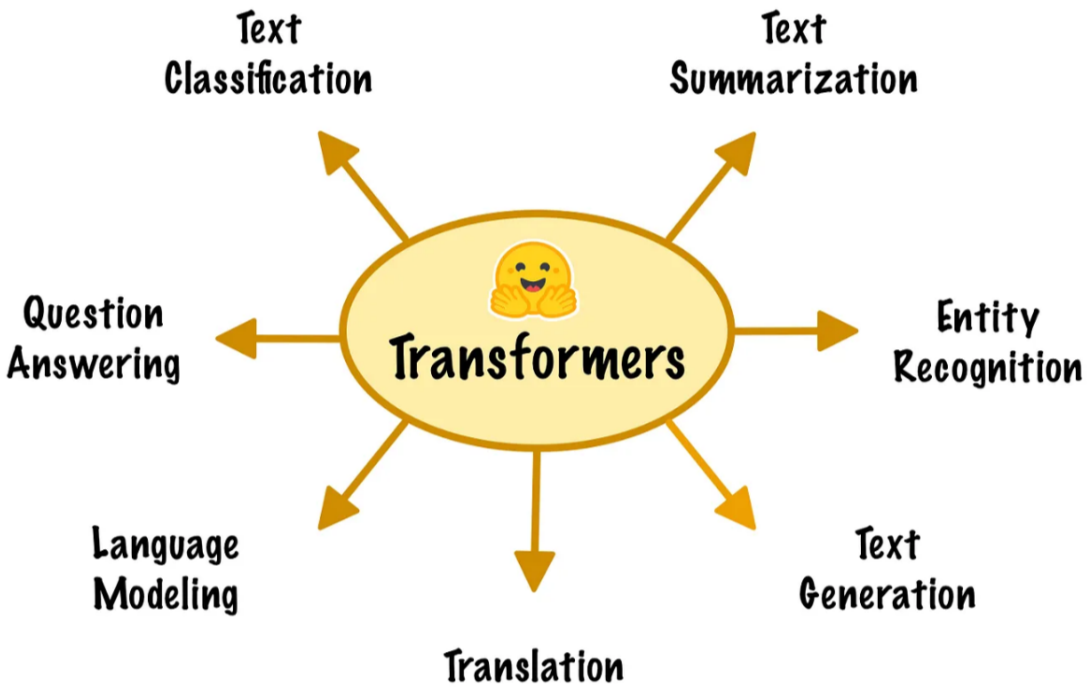
Transformersy	
Kierunek Sztuczna Inteligencja	Termin Czwartek 9:15
Zadanie ZMGSN LAB5	
Prowadzący Mgr inż. Joanna Szołomicka	data 18.01.2024r.
Autor Maciej Wilhelmi	indeks 252938



Rysunek 1: Schemat Transformersów

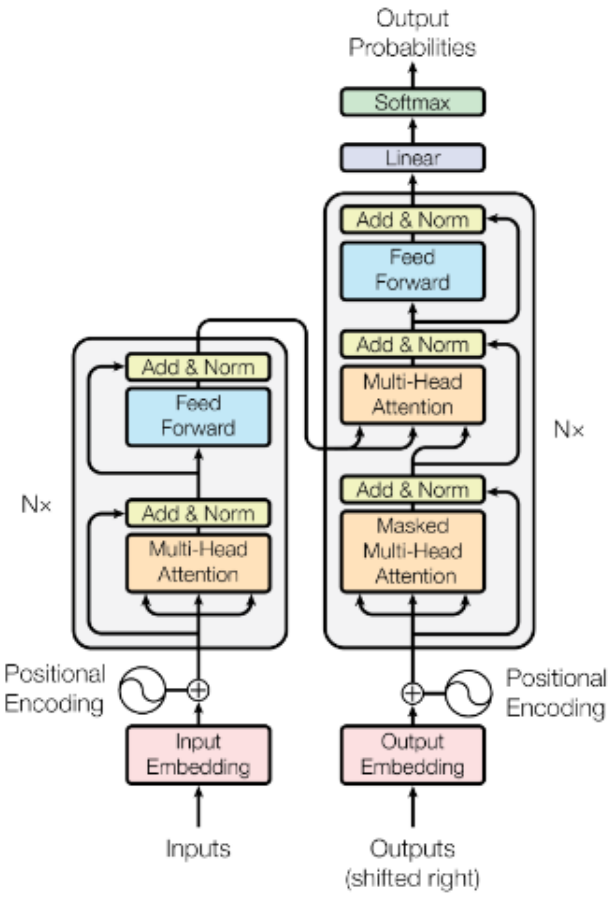
1 Wstęp

1.1 Opis zadania

Celem zadania było zapoznanie się z rodziną modeli typu Transformer, przetestowanie hiperparametrów oraz różnych typów modeli, na następnych stronach znajdują się wyniki eksperymentów.

1.2 Transformery

Transformer to architektura głębokiego uczenia oparta na mechanizmie uwagi wielogłowicowej. Wyróżnia się tym, że nie zawiera żadnych jednostek rekurencyjnych, a zatem wymaga krótszego czasu uczenia niż poprzednie rekurencyjne architektury neuronowe, takie jak długa pamięć krótkotrwała (LSTM), a jego późniejsza odmiana została powszechnie przyjęta do uczenia dużych modeli językowych na dużych datasetach, takich jak korpus Wikipedii i Common Crawl. Tekst wejściowy jest dzielony na n-gramy zakodowane jako tokeny, a każdy token jest konwertowany na wektor poprzez wyszukiwanie z tabeli osadzania słów. W każdej warstwie każdy token jest następnie kontekstualizowany w zakresie okna kontekstowego z innymi (niezamaskowanymi) tokenami za pomocą równoległego mechanizmu uwagi wielogłowicowej, umożliwiającego wzmocnienie sygnału dla kluczowych tokenów i zmniejszenie mniej ważnych tokenów.



Rysunek 2: Architektura transformera

2 Teoria

Poniżej opisane będą po krótce modele, które użyłem do wykonania eksperymentów i późniejszego porównania.

2.1 BERT

BERT - ang. Bidirectional Encoder Representations from Transformers

Ogromny zbiór danych zawierający 3,3 miliarda słów przyczynił się do ciągłego sukcesu BERT. BERT został specjalnie przeszkolony na Wikipedii (2,5 miliarda słów) i Google BooksCorpus (800 milionów słów). Te duże informacyjne zbiory danych przyczyniły się do głębokiej wiedzy BERT nie tylko o języku angielskim, ale także o naszym świecie!

2.2 RoBERTa

RoBERTa - ang. Robustly Optimized BERT Pretraining Approach

Opiera się na BERT i modyfikuje kluczowe hiperparametry, usuwając cel wstępnego trenowania dla następnego zdania i trenując z dużo większymi mini-partiami i szybkościami uczenia się.

Porównanie z BERTem:

BERT	RoBERTa
Statyczne maskowanie	Dynamiczne maskowanie
Wejście: dwa połączone segmenty dokumentu	Wejście: sekwencje zdań z dokumentów
Przewidywanie następnego zdania	Brak przewidywania następnego zdania
256 przykładów w paczce treningowej	2000 przykładów w paczce treningowej
Tokenizacja: Word-piece	Tokenizacja: byte-pair encoding
Korpus treningowy: BooksCorpus, Wikipedia EN	Dodatkowo: CC-News, OpenWebText, Stories
1 milion kroków uczenia	500 tysięcy kroków uczenia
Początek treningu na krótkich sekwencjach	Trenowanie wyłącznie na pełnych sekwencjach

Rysunek 3: BERT vs RoBERTa

2.3 DistilBERT

DistilBERT oferuje lżejszą wersję BERT; działa o 60% szybciej, zachowując ponad 95% wydajności BERT. DISTILBERT może być następnie dostrojony z dobrą wydajnością w szerokim zakresie zadań, tak jak jego większe odpowiedniki. Podczas gdy większość wcześniejszych prac badała wykorzystanie destylacji do budowania modeli specyficznych dla zadania, wykorzystujemy destylację wiedzy podczas fazy wstępnego szkolenia i pokazujemy, że możliwe jest zmniejszenie rozmiaru modelu BERT o 40%, przy jednoczesnym zachowaniu 97% jego możliwości rozumienia języka i 60% szybszym działaniu. Aby wykorzystać indukcyjne uprzedzenia wyuczone przez większe modele podczas wstępnego szkolenia, wprowadzamy potrójną stratę łączącą modelowanie języka, destylację i straty cosinus-distance. Nasz mniejszy, szybszy i lżejszy model jest tańszy do wstępnego trenowania.

3 Eksperymenty

3.1 Hiperparametry

Porównałem po 3 parametry w każdym zestawie: kroku uczenia, wielkości paczki danych oraz liczbie epok. Poniżej znajdują się wyniki.

	Lr=0.00001	Lr=0.0001	Lr=0.001	BS=16	BS=32	BS=64	E=5	E=20	E=40
f1 0/1	0.94/0.67	1.00/0.66	0.98/0.87	0.98/0.86	0.97/0.86	0.94/0.67	0.94/0.72	0.99/0.91	0.98/0.91
f1 weighted	0.9	0.93	0.96	0.96	0.96	0.9	0.91	0.98	0.97
F1 macro	0.8	0.87	0.93	0.92	0.92	0.8	0.83	0.95	0.95

Rysunek 4: Hiperparametry

Jak można zauważyć najplepiej zadziałał zestaw parametrów:

- Learning rate: 0.001
- Batch size: 16
- Epochs: 20

Czyli podsumowując, najmniejszy krok uczenia wcale nie był najlepszy, większy zdecydowanie lepiej sobie poradził. Rozmiar pakietu próbek do uczenia najlepszy był najmniejszy, jednak w stosunku do dwa razy większego (32) różnice były nikłe. A liczba epok, po której otrzymałem najlepsze wyniki znajdowała się po środku (20), jednak nie była o wiele lepsza od 40, jednak uczył się model dwukrotnie szybciej.

3.2 Porównanie modeli

Porównując same modele, w tych samych konfiguracjach, najlepiej wypadł BERT, jednak z bardzo niewielką różnicą do każdego innego modelu, który wypróbowałem.

Użyłem tutaj dwóch nowych modeli, pierwszy: *bert-large-uncased*, który jest modyfikacją BERTa z większą liczbą neuronów w warstwach ukrytych (poradził sobie najlepiej z konkurentami) oraz *xlm-roberta-base*, który został wytrenowany na 100 obcych językach na 2.5 TB danych z *CommonCrawl*

	Distilbert	Roberta	xlm-roberta-base	bert-large-uncased	Bert
f1 0/1	0.98/0.87	0.98/0.90	0.98/0.88	0.98/0.91	0.99/0.91
f1 weighted	0.96	0.97	0.97	0.97	0.98
F1 macro	0.93	0.94	0.93	0.95	0.95

Rysunek 5: Porównanie modeli

3.3 Modyfikacje Rozszerzenia

Zmieniając architekturę sieci używaną do douczenia Transformera do zadania klasyfikacji binarnej tekstów w pierwszym wariancie usunąłem warstwę *Dropout*, a w drugim powiększyłem rozszerzenie o dodatkowe warstwy liniowe.
Oba warianty działały gorzej od pierwowzoru, jednak z tych obu lepiej zadziałał wariant z większą ilością warstw

	bez Dropout	Bert więcej warstw
f1 0/1	0.95/0.72	0.96/0.79
f1 weighted	0.92	0.94
F1 macro	0.84	0.88

Rysunek 6: Modyfikacje rozszerzenia

Następnie stworzyłem architektury rozszerzeń oparte na jednowymiarowych warstwach konwolucyjnych oraz LSTM. Konwolucja zadziałała lepiej z obu wariamtów, jednak gorzej niż zwykłe warstwy liniowe.

	Ext Conv	Ext LSTM
f1 0/1	0.95/0.76	0.85/0.51
f1 weighted	0.93	0.81
F1 macro	0.86	0.68

Rysunek 7: Warianty rozzszerzenia

3.4 Parametry tekstu

Jako ostatnie badałem wpływ padowania tekstu na wyniki modelu. Użyłem dla BERT, DistilBERT, RoBERTa 4 różnych wartości używanych jako ograniczenie padowania.

	Max text	Max=10	Max=50	longest
f1 0/1	0.93/0.69	0.87/0.51	0.95/0.75	0.95/0.75
f1 weighted	0.90	0.82	0.93	0.92
F1 macro	0.81	0.69	0.85	0.85

Rysunek 8: padding BERT

	Max text	Max=10	Max=50	longest
f1 0/1	0.99/0.91	0.94/0.70	0.99/0.92	0.99/0.92
f1 weighted	0.98	0.91	0.98	0.98
F1 macro	0.95	0.82	0.95	0.95

Rysunek 9: padding DistilBERT

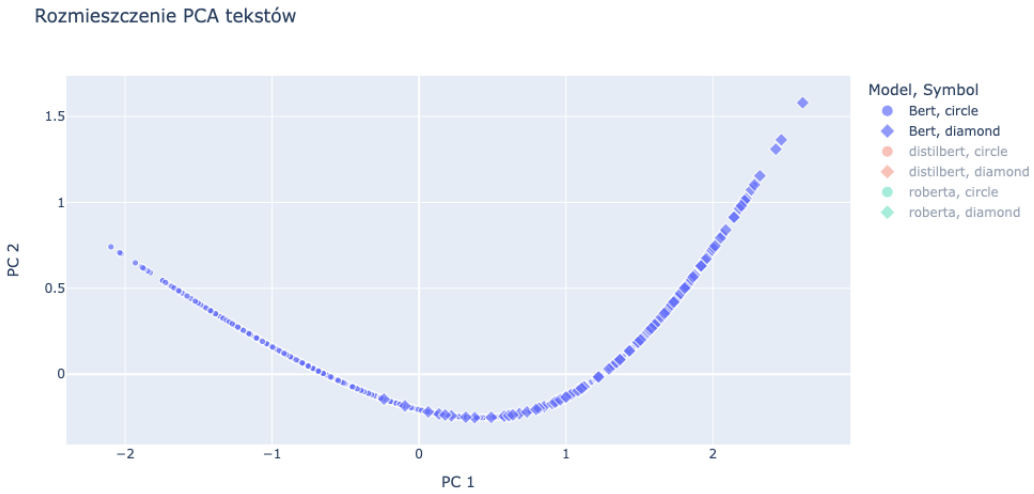
	Max text	Max=10	Max=50	longest
f1 0/1	0.98/0.89	0.97/0.82	0.99/0.92	0.99/0.91
f1 weighted	0.94	0.95	0.98	0.98
F1 macro	0.97	0.90	0.95	0.95

Rysunek 10: padding RoBERTa

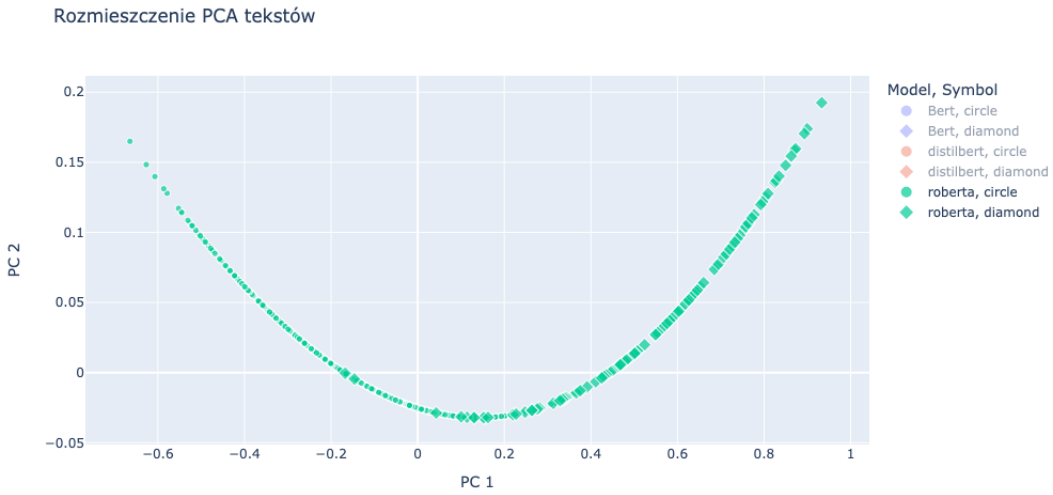
Ostatecznie Dla wszystkich modeli najlepiej sprawowały się max=50 tokenów oraz longest (najdłuższy w sekwencji tekstów). Jednak dla modelu DistilBERT równie dobrze zadziałał Max text (czyli maksymalna długość tekstu wśród wszystkich tekstów w zbiorze., tak samo dla RoBERTy. BERT natomiast o wiele gorzej radził sobie z tym parametrem.

4 Ewaluacja

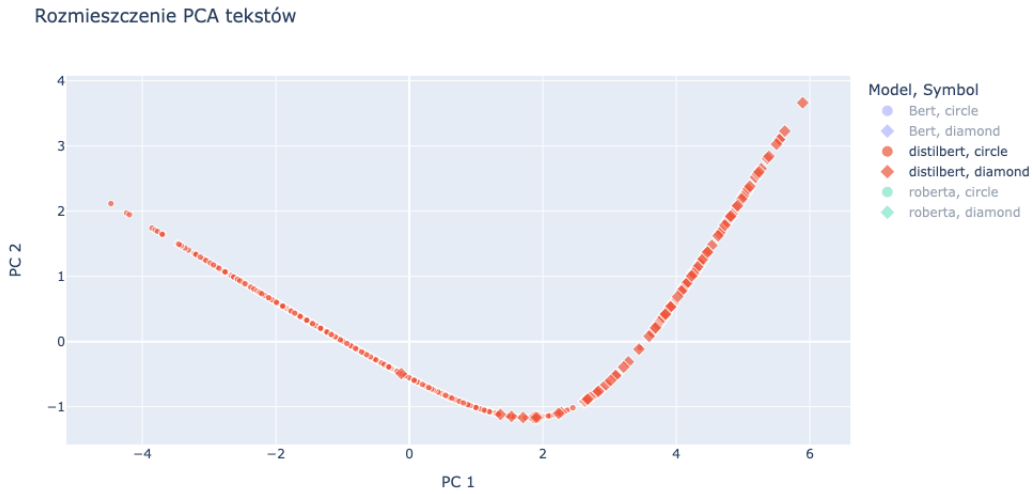
W ewaluacji posłużyłem się przeniesieniem reprezentacji tekstów wyciągniętych z wyuczonych modeli i przedstawienie ich na wykresie. Przedstawione są one poniżej:



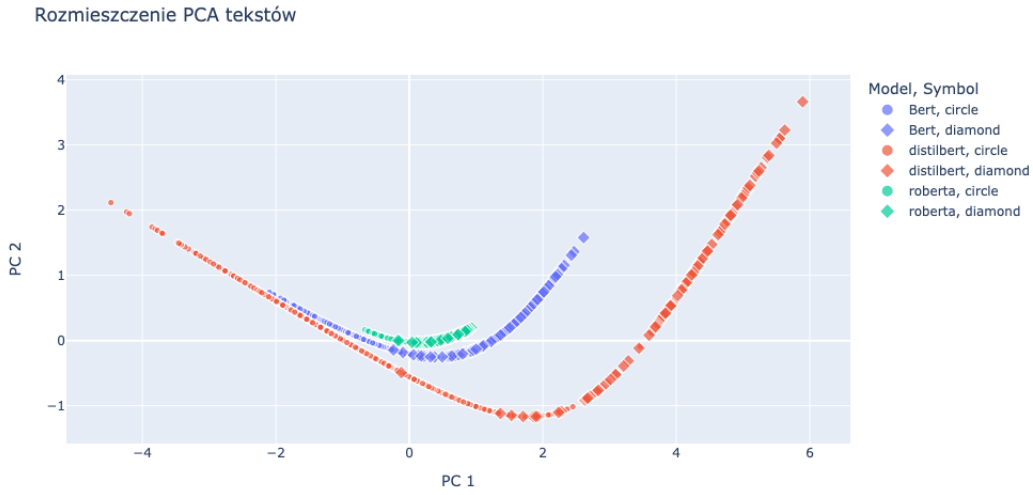
Rysunek 11: Repr BERT



Rysunek 12: Repr RoBERTa



Rysunek 13: Repr DistilBERT



Rysunek 14: Repr all

Wszystkie charakterystyki są do siebie zbliżone. Podobnie mają rozłożone przykłady obu klas, jedna w większości na lewym ramieniu paraboli, a drugie na prawym. Jednak różnią się delikatnie kształtem paraboli i przedziałem wartości. Właściwości te wynikają z tego, że wszystkie bazują na tej samej architekturze, jednak z niewielkimi zmianami.