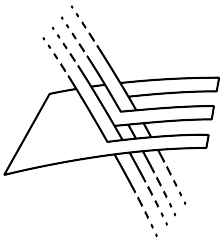

Wydział Elektroniki
i Technik Informacyjnych

Politechnika Warszawska

Wprowadzenie do przetwarzania języka naturalnego

Projekt:

Przystosowanie modelu (RoBERTa) do określania podobieństwa semantycznego dwóch zdań w języku angielskim.

Dominika Ziółkiewicz
Konrad Dumin

Contents

Etap I

1 Treść polecenia do projektu

Przystosowanie modelu (RoBERTa) do określania podobieństwa semantycznego (iSTS - `https://alt.qcri.org/semeval2016/task2/`) dwóch zdań w języku angielskim. Odniesienie się do wyników z obronionej pracy magisterskiej/projektu. 2-3 os.

Język implementacji: Python

Zadania:

1. Zapoznać się z pracą magisterską/projektem i przygotowanym rozwiązaniem.
2. Zapoznać się z wykorzystywanym modelem RoBERTa.
3. Zmodyfikować model RoBERTa zgodnie z informacjami przekazanymi przez prowadzącego.
4. Policzyc miary F dla stworzonego systemu dla wszystkich dostepnych danych z SemEval 2015 Interpretable STS z pojedynczymi relacjami oraz SemEval 2016 - Interpretable STS z rozbudowanymi relacjami.

2 Analiza tematu projektu

2.1 Model RoBERTa

Model językowy **RoBERTa** (z j. ang. *Robustrly Optimized BERT Pretraining Approach* został zaprojektowany jako rozwinięcie nieco starszego narzędzia **BERT** (ang. *Bidirectional Encoder Representations from Transformers*). BERT został zaprezentowany światu w 2018r. przez badaczy z firmy Google i stanowi dwukierunowy transformer wyuczony na tzw. *maskach językowych* oraz z użyciem techniki predykcji następnego zdania **NSP** (z j. ang. *Nest Sentence Prediction*). Uczenie modelu odbywało się na artykułach z Wikipedii oraz wybranych książkach z biblioteki *Toronto Book Corpus*.

Model BERT został zaprojektowany do rozwiązywania realnych problemów z dziedziny ptrzetwarzania języka naturalnego. W zwiąku z tym dokładnie go przetestowano na wielu specjalnie przygotowanych benchmark'ach dla modeli przetwarzania języka naturalnego. Są to m. in. test **GLUE** (*General Language Understanding Evaluation*), test **SQuAD** (*Stanford Question Answering Dataset*) w wersji 1.0 i 2.0 oraz baza danych pytań **SWAG**. We wszystkich testach model BERT poradził sobie znakomicie, osiągając bardzo wysokie wyniki i stając się nowym wyznacznikiem jakości dla przyszłych modeli.

Budowę modelu BERT można wysokopoziomowo podzielić na 3 warstwy:

1. *embedding* - warstwa odpowiedzialna za kodowanie zdań na tablicę wektorów do dalszego przetwarzania
2. *encoding* - warstwa służąca do analizy stworzonych wektorów i ocenie relacji między nimi
3. *un-embedding* - warstwa z powrotem tłumacząca wektory z poprzedniej warstwy na tokeny opisujące relacje pomiędzy badanymi zdaniami

Ostatnia warstwa jest wymagana wyłącznie w fazie pretrenowania modelu. Pomaga dotrenować model tak, aby dostosować go do konkretnego, wybranego zadania. Jeśli dotrenowywanie nie jest potrzebne, można ją bezpiecznie pominąć w działaniu i operować bezpośrednio na wygenerowanych wektorach w celu zaoszczędzenia mocy obliczeniowej.

Model BERT był trenowany z użyciem 2 technik: masek językowych oraz predykcji następnego zdania. Maski językowe polegały na tym, że model próbował przewidzieć, jakie słowo zostało zamaskowane w podanym zdaniu na podstawie kontekstu zdania. Maskowano ok. 15% słów w tekście tak, aby model miał dostatecznie dużo materiału do uczenia się. Predykcja następnego zdania **NSP** (ang. *Next Sentence Prediction*) zaś stawiała przed modelem pary zdań: połowa par to następujące po sobie 2 zdania wzięte z wybranego tekstu, a druga połowa to były pary zdań dobrane losowo z tego samego tekstu, nienastępujące po sobie w oryginalnym dokumencie. Zadaniem modelu było rozróżnienie, które pary należały do którego zbioru.

Model RoBERTa stanowi rozwinięcie koncepcji modelu BERT. Został zbudowany na bazie modelu BERT przez naukowców z zespołu Facebook AI. W celu poprawienia efektywności działania modelu BERT zoptymalizowano wartości niektórych kluczowych parametrów sterujących przebiegiem algorytmu uczenia. Ponadto zmieniono maskę ukrywającą słowa ze statycznej na dynamiczną (generowaną od nowa przy każdym przebiegu algorytmu uczenia). Zmniejszono znaczenie treningu NSP podczas trenowania algorytmu. Zwiększono ilość danych podawanych naraz do algorytmu w jednym przebiegu uczenia oraz zdecydowanie zwiększono wielkość zbioru trenującego. Zastosowane zmiany wpłynęły pozytywnie na skuteczność nowopowstałego modelu, uzyskując jeszcze lepsze rezultaty, niż przed modyfikacjami.

2.2 Zagadnienie iSTS

iSTS (z j. ang. **interpretable Semantic Textual Similarity - interpretowana analiza semantyczna tekstu**) to jedno ze sztandarowych zadań dla modeli przetwarzania języka naturalnego. Polega na ocenie znaczenia semantycznego pary zdań i przyporządkowania tej parze miary podobieństwa znaczenia zawartych w niej zdań. Ocena podobieństwa zdań odbywa się kawałek po kawałku - model rozważa fragmenty obu zdań, a nie całe zdania. Każdej parze fragmentów przpisywana jest etykieta opisująca relację między tymi fragmentami. W tym celu wprowadzono specjalny system klasyfikacji fragmentów, składający się z oceny liczbowej oraz etykiety.

Ocena liczbowa to liczba całkowita z zakresu od 0 włącznie do 5 włącznie. Liczba 0 oznacza, że podane fragmenty nie są z sobą w ogóle powiązane, a liczba 5, że są znaczeniowo identyczne. Podobieństwo częściowe jest wyrażane liczbami od 1 do 4 - im wyższa liczba, tym większy stopień podobieństwa.

Etykieta stanowi objaśnienie, jaka jest dokładnie relacja pomiędzy rozważanymi fragmentami. Wprowadzono następujące etykiety:

- **EQUI** - badane fragmenty są znaczeniowo identyczne
- **OPPO** - fragmenty są znaczeniowo przeciwne
- **SPE1** oraz **SPE2** - fragmenty są podobne znaczeniowo, ale fragment ze zdania odpowiednio 1. albo 2. jest bardziej szczegółowy; podaje więcej informacji
- **SIMI** - znaczenie fragmentów jest podobne, ale nie można powiedzieć, który niesie więcej inforamcji

- **REL** - fragemnty są z sobą znaczeniowo powiązane, ale słabiej, niż w przypadku SIMI

Fragmenty są w obu zdaniach podobnej długości i mapuje się je zawsze 1:1. W związku z tym, dodano również dodatkowe 2 oznaczenia dla specjalnych przypadków, gdzie jedno ze zdań jest dłuższe, niż drugie i pewne fragmenty tego zdania nie posiadają swojego odpowiednika w drugim zdaniu.

- **ALIC** - kiedy jeden fragment zdania posiada więcej, niż jeden pokrewny znaczeniowo odpowiednik w drugim zdaniu, dla celów klasyfikacji zostanie mu przyporządkowany fragment o bliższym mu znaczeniu, drugi zaś otrzyma etykietę ALIC
- **NOALIC** - drugie zdanie jest krótsze i podany fragment w ogóle nie posiada swojego odpowiednika w drugim zdaniu (nie ma do czego go porównać)

Ocena liczbowa i etykieta są ze sobą ściśle powiązane. Dlatego też oprócz powyższych ocen i etykiet dodano również pewne reguły, aby oceny były ze sobą zawsze spójne:

- fragmenty o klasyfikacji ALIC, NOALIC oraz OPPO automatycznie otrzymują ocenę 0
- fragmenty o klasyfikacji EQUI automatycznie otrzymują ocenę 5
- pozostałe etykiety muszą mieć ocenę z przedziału od 1 do 4

3 Plan implementacji rozwiązania - główna część projektu

Naszym zadaniem zadaniem jest powtórzenie pretrenowania modelu RoBERTa na danych z konkursu SemiEval 2015 i 2016.

3.1 Pretrenowanie modelu

- **Model:** RoBERTa z biblioteki PyTorch
- **Architektura sieci:** Korzystając z architektury z projektu RoBERTa_Konopczyński zmodyfikować go tak jak jest to opisane w dokumentacji wstępnej projektu (praca Nitkiewicz, Sadowski). Porównać z architekturami z innych prac magisterkich/projektów.

  W pracy RoBERTa_Konopczyński (RoBERTa-for-iSTS-task/modeling/roberta_ists.py) została zaimplementowana architektura w której uzyskuje się predykcje wartości STS oraz (oddzielnie) wyjaśnienia. Ogólny model systemu przedstawia się w taki sposób:

  1. Zakodowane fragmenty zdań jako wejście do modelu
  2. RoBERTa jako ekstraktor cech
  3. Uśredniony wektor z modelu o długości 1024
  4. Rozdzielenie w zależności od problemu:
     - Regresja dla wartości STS
     - Klasyfikacja wartośći warstwy objaśniającej (explanatory layer)

  Żeby uzyskać STS, wyjscie z roberta_large przechodzi przez warstwę *dropout -> linear -> relu -> droput -> linear (output).* Natomiast uzyskanie drugiego wyjścia predykowanego, czyli objaśnienia, mamy kolejno *dropout -> linear -> relu -> linear -> softmax (output).*

Natomiast w dokumentacji wstępnej do naszego projektu (czyli pracy Nitkiewicz, Sadowski) struktura wygląda nieco prościej ponieważ wyjście z roberta_large przechodzi przez warstwę liniową, dropout (z prawdopodobieństem 10%) oraz kolejną warstwę liniową.

- **Planowane uczenie modelu:**
  - Funkcja straty:
    - dla predykcji wartości STS - średni błąd kwadratowy (MSE)
    - dla predykcji wyjasnienia - *negative log likelihood loss* (NLLL)
  - Algorytm optymalizacji: ADAM
  - Liczba epok: 10
- **Dane:** Pobranie danych ze strony konkursu SemiEval (Interpretable STS) i zapisanie w formie csv. Dostępne zbiory to:
  - "Images Captions" - podpisy obrazków (2015 r.) - (970 par treningowych, 943 par testowych)
  - "News Headlines" - nagłówki informacji prasowych (2015 r.) -(2188 par treningowych, 1172 par testowych)
  - "Answers-Students" - odpowiedzi uczniów z systemu dialogowego (2016 r.) - (919 par treningowych, 933 par testowych)
- **Technologia:**
  - Język programowania: Python 3
  - Środowisko: Google Colaboratory
  - Biblioteki:
    - PyTorch (model roberta_large, architektura sieci)
    - matplotlib, seabor (wizualizacja)
    - sklearn (analiza wyników, wykorzystanie np. dostępnych metryk F1)
    - NumPy (operowanie na danych)

Kod główny projektu będzie wykonywany przy użyciu środowiska jakim jest Google Colaboratory. Środowisko to jest w formie internetowego notatnika (Jupyter Notebook) z udostępnionymi zasobami CPU i GPU, który jest łatwo udostępnialny. Idealnie nadaje się do projektów kooperacyjnych z takich dziedzin jak machine learning i data science.

Dodatkową zaletą notatnika Google Colab jest możliwość korzystania z najnowszych wersji wszelkich bibliotek do języka Python bez martwienia się o ich pobieranie na własny komputer i konfigurację środowiska.

3.2 Uzyskanie wyników

- Wyniki ewaluacji miar F1 bezpośrednio z modelu

Oddzielnie będą obliczane miary dla każdego ze zbiorów (Images, Headlines i Answear-Students) oraz dla zbioru łącznego je wszystkie.

Miara F1 - zarówno dla oceny (score) jak i typu relacji (type) - jest obliczana ze wzoru:

$$\mathrm{F1} = \frac{2 * \mathrm{TP}}{2 * \mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

Figure 1: Źródło: sklearn.metrics.f1_score

TP, TN, FP, FN to wartości z tablicy pomyłek. W głównej mierze F1 jest średnią harmoniczną wartości "prezycji" (*precision*) oraz "zupełność" (*recall*). Jej zadaniem jest podanie pojedynczej wartości z przedziału od 0 do 1, która określać będzie skuteczność klasyfikatora. Dzięki średniej harmonicznej dodatkowo premiuje się zbliżony wynik dla tych dwóch wskaźników, dlatego jeśli np. klasyfikator uzyska dużą precyzję ale niską zupełność jego ostateczna ocena (F1) nie będzie wysoka.

- Wykresy macierzy pomyłek
- Uzyskanie miar F1 ze skryptów *evalF1_penalty* i *evalF1_no_penalty* napisanymi w języku Perl
  - F1 score - z uwzględniem oceny (od 0 do 5)
  - F1 type - z uwzględnieniem typu (etykiety)
  - F1 type + score