Inżynieria Lingwistyczna

Wykrywanie cyberprzemocy

Michalina Kwolik s23922

Gabriela Olszewska s28652

1 Temat

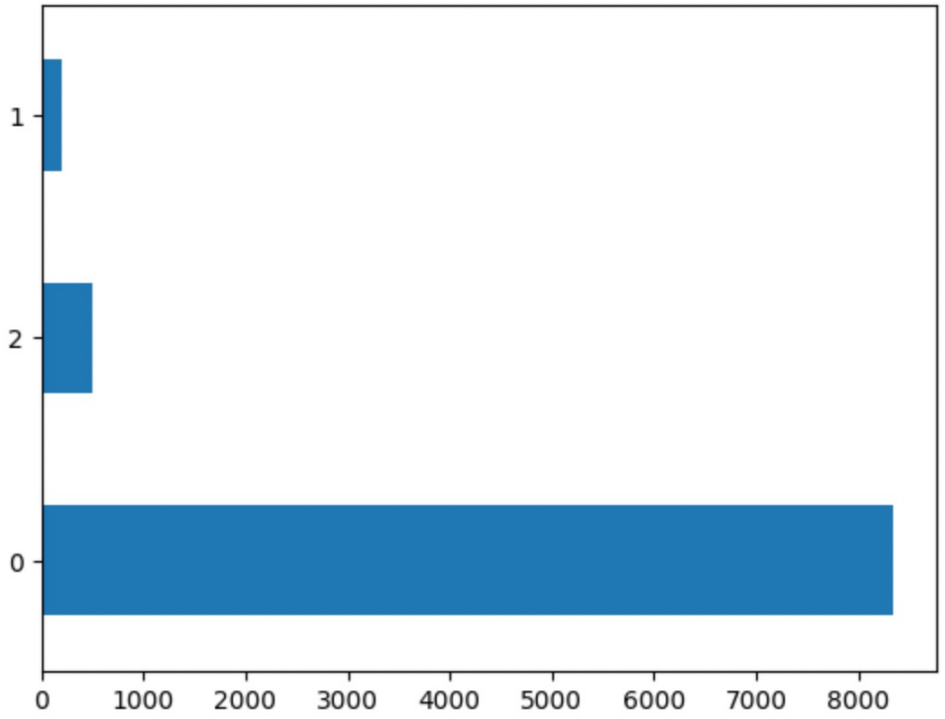
Tematem projektu było użycie modelu językowego BERT do wykrywania hatespeech oraz cyberprzemocy w internecie. Dane do trenowania modelu zostały wzięte z http://2019.poleval.pl/index.php/tasks/task6 i zawierały tweety pobrane z publicznych dyskusji na platformie twitter.com. Wpisy były klasyfikowane w kategorii trzech klas, 0 - wydźwięk neutralny, 1 – cyberbullying oraz 2 – hatespeech. Został pobrany zestaw z danymi treningowymi (podzielony później także na zestaw ewaluacyjny) oraz zestaw testowy.

2 Przygotowanie danych

1. Ściągnięcie dwóch zestawów danych (po dwa pliki każdy, jeden to tekst, drugi to etykiety)
2. Połączenie każdego zestawu w dataframe z kolumnami 'text' i 'labels'
3. Dataframe treningowy podzielony na zestaw **treningowy** liczący **9041** etykiet oraz na zestaw **ewaluacyjny** liczący **1000** etykiet
4. Dataframe z zestawem **testowym** pozostaje bez zmian (**1000** etykiet), przyjęty podział 80-10-10
5. Podjęta próba wyczyszczenia danych lecz przy późniejszym trenowaniu pojawia się błąd (próba obejmowała usunięcia słów "RT" i "@anonymized_account" oraz "emojis")

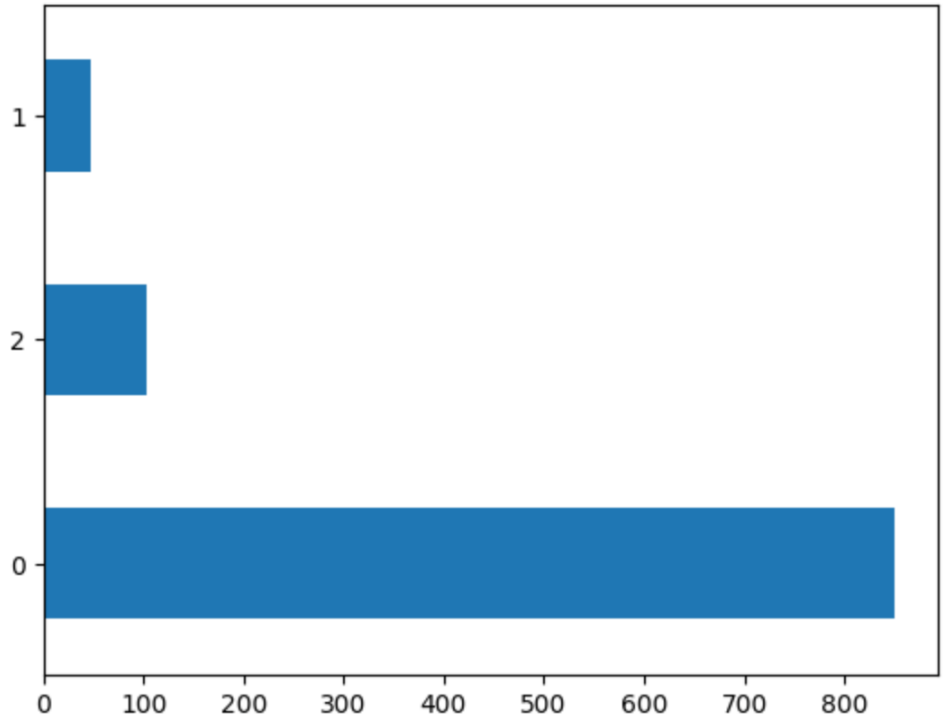
```
def preprocess_dataframe(df):
    df['text'] = df['text'].apply(lambda x: x.replace('@anonymized_account', ''))
    df['text'] = df['text'].apply(lambda x: x.replace('RT', ''))
    df['text'] = df['text'].apply(lambda x: re.sub(r'[^\w\s\d\s]+', '', x))

    return df

df = preprocess_dataframe(df)
```

Dane treningowe:



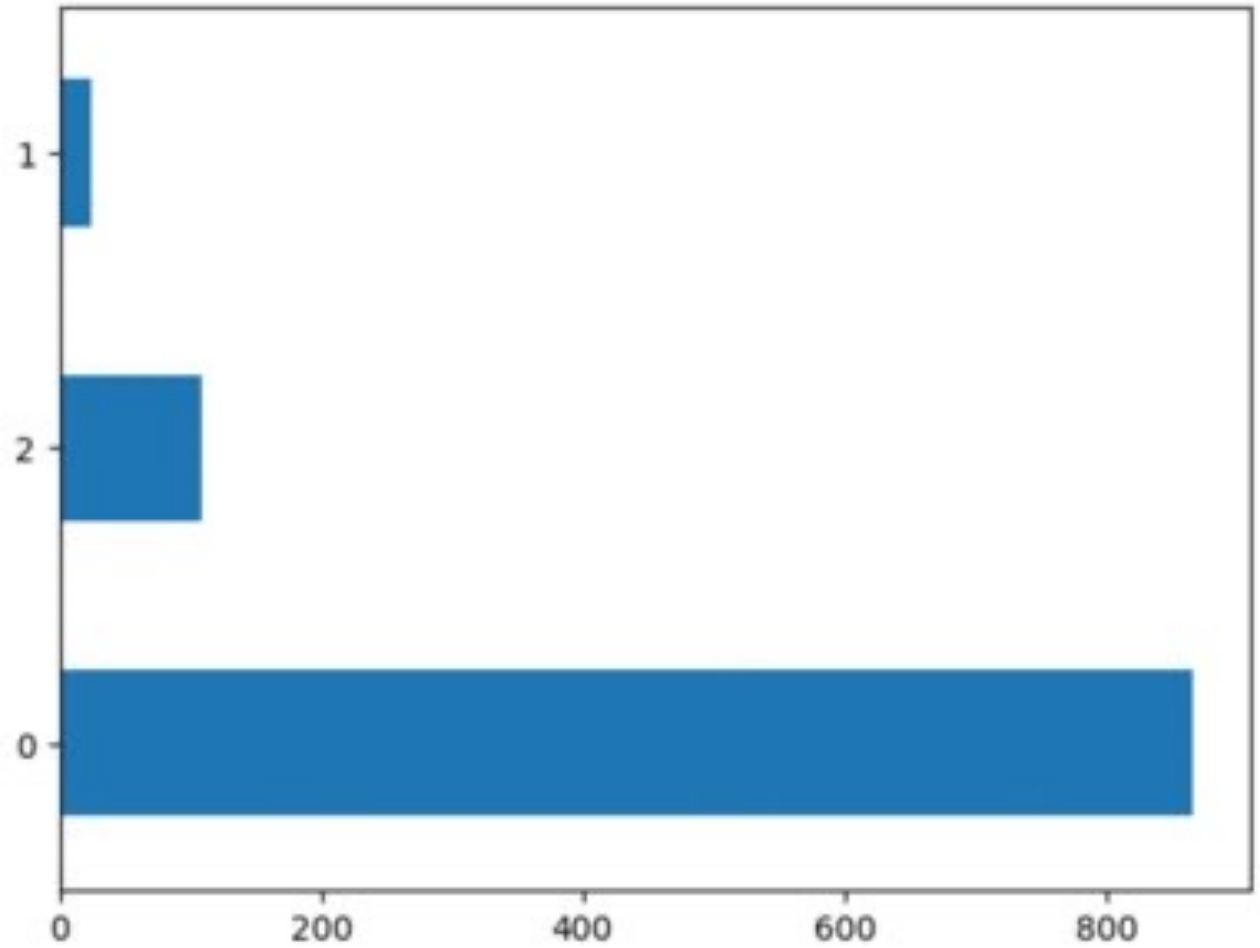

```
Value count for training data:
0    8340
2     495
1     206
Name: labels, dtype: int64
```

Dane ewaluacyjne:




```
Value count for evaluation data:
0    850
2    103
1     47
Name: labels, dtype: int64
```

Dane testowe:




```
Value count for test data:
0    866
2    109
1     25
Name: labels, dtype: int64
```

Wnioski:

- Etykiety 1 i 2 są w każdym dataset w mniejszości (zwykle stanowią niecałe 10% wszystkich etykiet)
- Taki nierównomierny rozkład danych może wpływać na skuteczność modelu w wykrywaniu hatespeech lub cyberbullying
- Model może mieć tendencję do dominacji klasy 0 z powodu dużo większej liczby jej wystąpień
- Można ten problem rozwiązać za pomocą oversampling lub undersampling
- Nierównomierny rozkład klas może wpłynąć na jakość oceny modelu (np. wysoka precyzja dla dla klasy 0, a słabszy wynik dla klas 1 i 2)

4 Trenowanie modelu

- Wszystkie próby (udane i nieudane) zapisane były na Wandb.
- Pierwsze cztery główne próby zakończyły się błędami już gdy przechodziła prawie cała epoka (po około godzinie). Wynikało to na początku z powodu kodu na czyszczenie danych (wspomniane wcześniej), a później problem pojawiał się z kodem na liczenie F1, accuracy, precision i recall. Mimo nieudanych prób, argumenty do trenowania były zmieniane żeby zobaczyć jak model zachowuje się w trakcie trenowania. Zmieniana była liczba epok (pierwsze cztery próby i tak nie dokańczały pierwszej epoki), learning rate oraz training i evaluation batch size. Jedyne wykresy z tych przebiegów to training loss, global step oraz learning rate.

RUN 1
LR: 4e-5
EPOCHS: 1
TRAIN BATCH SIZE: 32
EVAL BATCH SIZE: 32

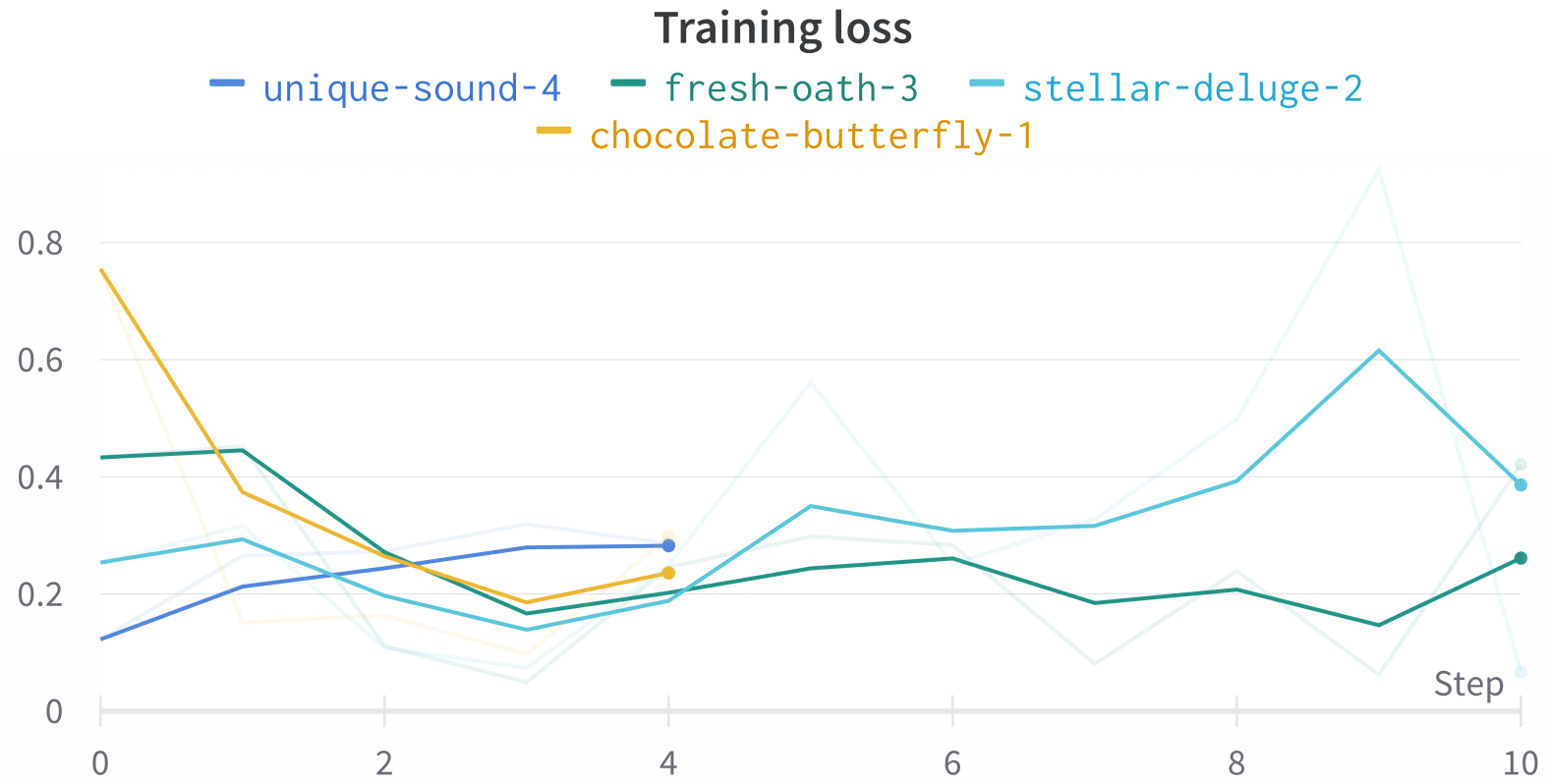
RUN 2
LR: 1e-5
EPOCHS: 1
TRAIN BATCH SIZE: 16
EVAL BATCH SIZE: 32

RUN 3
LR: 1e-5
EPOCHS: 1
TRAIN BATCH SIZE: 16
EVAL BATCH SIZE: 32

RUN 4
LR: 1e-5
EPOCHS: 2
TRAIN BATCH SIZE: 32
EVAL BATCH SIZE: 32

Training loss:

- Wykres ten przedstawia postęp uczenia się modelu (oś Y to wartość straty treningowej) dla przebiegów 1, 2, 3 oraz 4
- Strata treningowa powinna mieć trend malejący (najlepiej wypadł przebieg stellar-deluge-2)



Udane przebiegi:

- Ostatnie cztery przebiegi (5, 6, 7, 8) zakoczyły się bez żadnych błędów. Tak jak w wypadku pierwszych czterych, parametry były zmieniane oraz ostatnie trzy (6, 7, 8) miały dodaną metrykę F1, precision, recall i accuracy).
- Argumenty w ostatnich przebiegach:

RUN 5
LR: 1e-4
EPOCHS: 1
TRAIN BATCH SIZE: 16
EVAL BATCH SIZE: 32

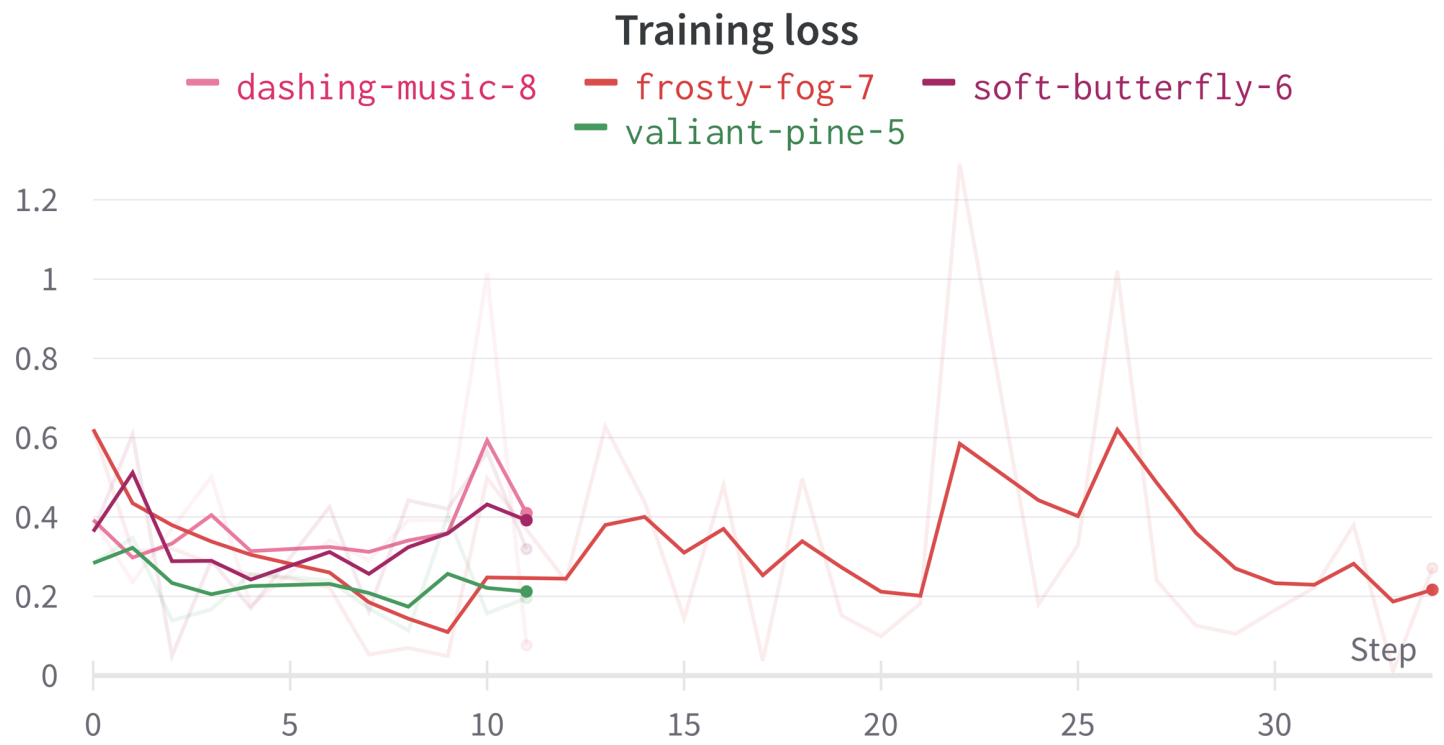
RUN 6
LR: 1e-5
EPOCHS: 2
TRAIN BATCH SIZE: 32
EVAL BATCH SIZE: 32

RUN 7
LR: 2e-5
EPOCHS: 3
TRAIN BATCH SIZE: 16
EVAL BATCH SIZE: 32

RUN 8
LR: 1e-3
EPOCHS: 2
TRAIN BATCH SIZE: 32
EVAL BATCH SIZE: 32

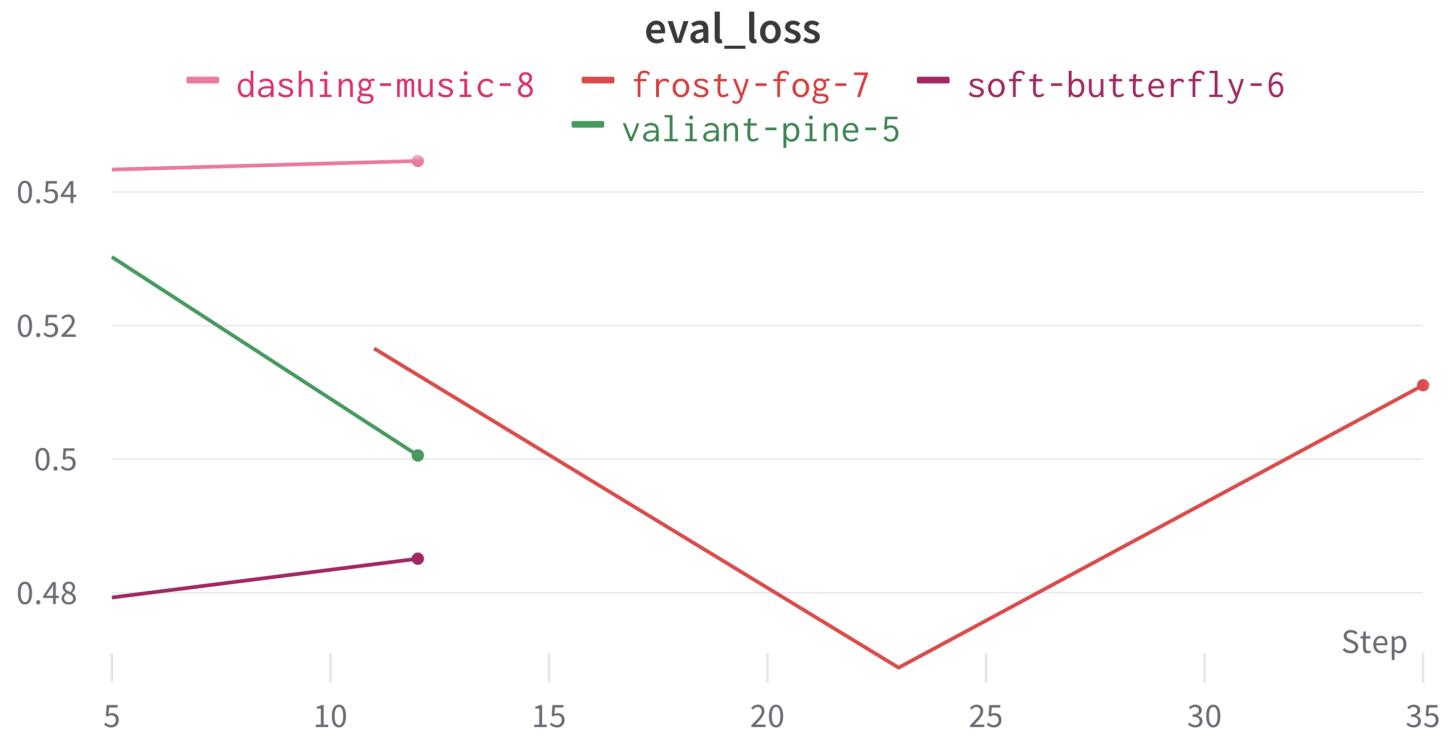
Training loss:

- Trening nr 7 ma najwięcej kroków ze względu na 3 epoki
- Strata treningowa powinna mieć trend malejący (najlepiej wypadł przebieg stellar-deluge-2)



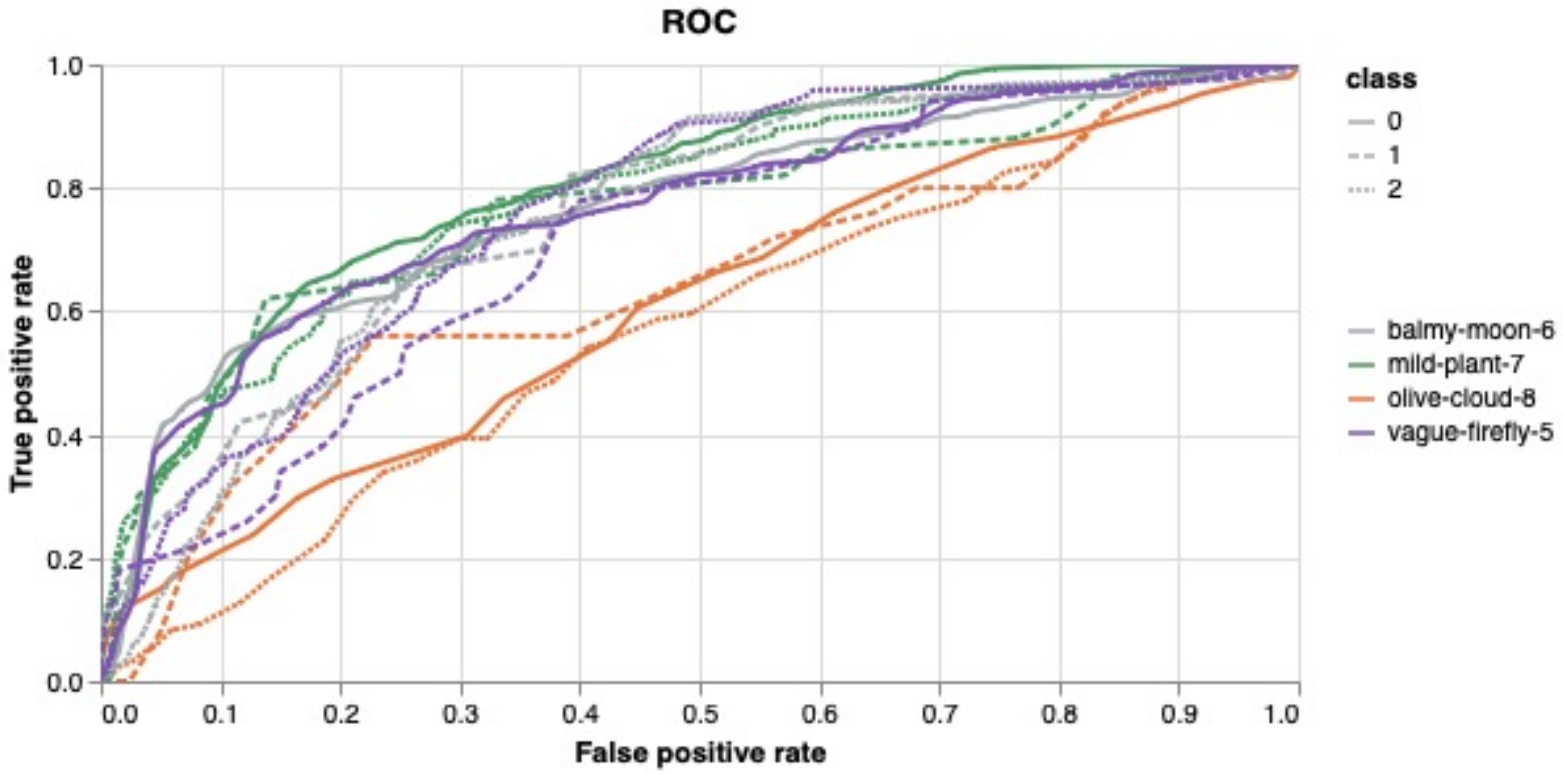
Evaluation loss:

- Wykres ten przedstawia jak dobrze model sobie radzi z danymi z ewaluacji
- Tak jak w przypadku traning loss, strata dla ewaluacji powinna mieć trend malejący
- Najlepiej wyszło w przebiegu valiant-pine-5, który jako jedyny miał 1 epokę (reszta 2-3). Może to świadczyć o tym, że już więcej epok powoduje przeuczenie modelu (widać wzrost w przebiegu 6, 7 i 8)



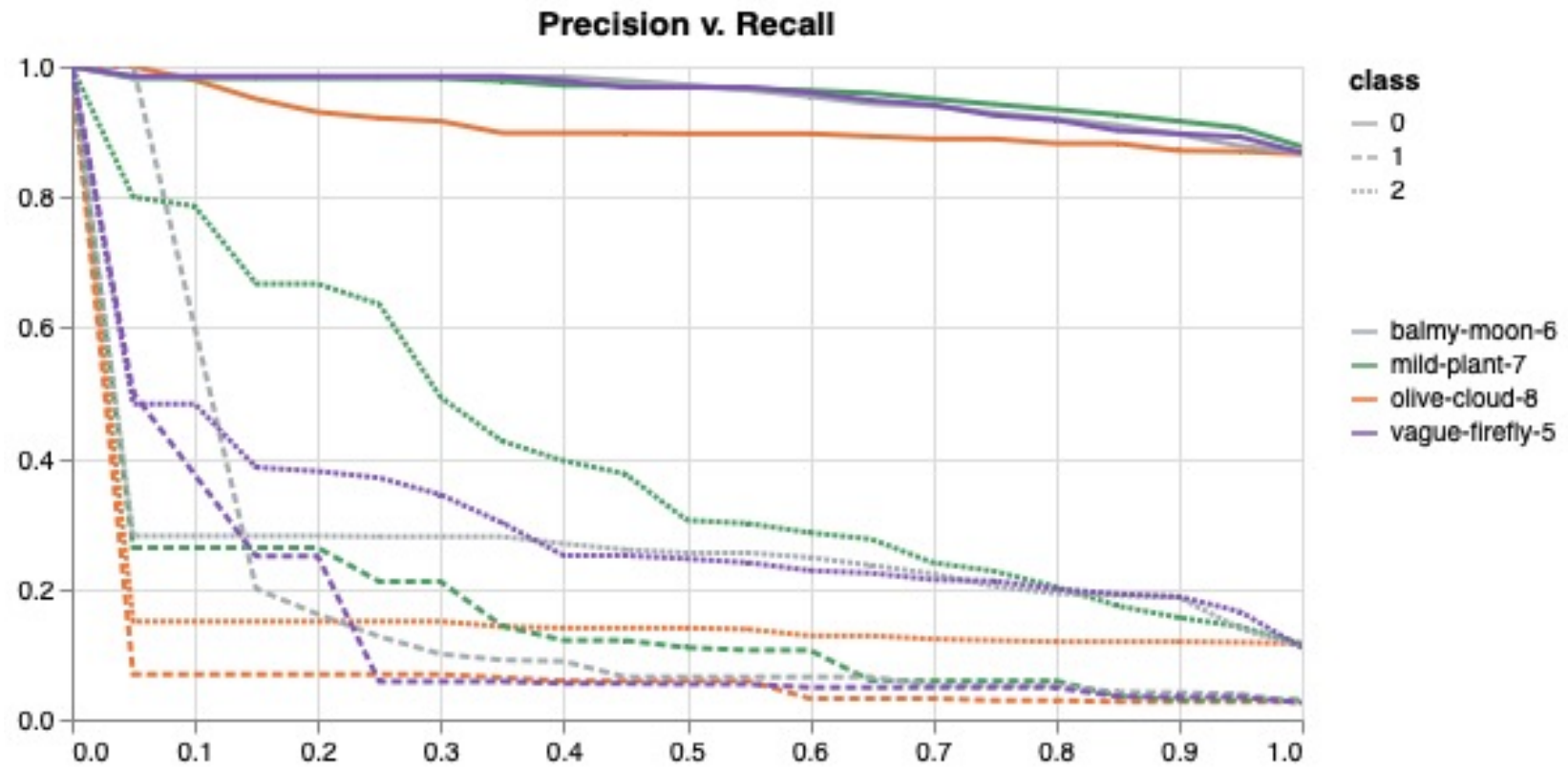
ROC:

- Wykres ROC ilustruje zdolność modelu do rozróżniania między klasami (0, 1, 2).
- Im wyżej położony jest punkt na ROC curve, tym wyższa jest wartość TPR, co oznacza lepszą zdolność klasyfikatora do identyfikowania pozytywnych próbek przy minimalnej liczbie fałszywie pozytywnych klasyfikacji.
- Dobry klasyfikator powinien się łączyć od (0,0) do (1,1), co widać na poniższym wykresie głównie dla przebiegu 5 i 7. Ponownie przebieg 8 (3 epoki) spisał się najgorzej.



Precision vs Recall:

- Precyzja (oś X) oraz czułość (oś Y).
- Idealny przypadek to punkt zbilansowany między oba parametrami.



6 TF-IDF

- Użycie TF-IDF oraz klasyfikatora (logistyczna regresja) za pomocą Sklearn aby porównać wyniki otrzymane z trenowania modelu
- Kroki:
  - Przekształcenie danych treningowych i ewaluacyjnych na wektory za pomocą TF-IDF
  - Trening klasyfikatora na danych treningowych i odpowiadających etykietach
  - Dokonanie predykcji na danych ewaluacyjnych i porównanie danych z predykcji i prawdziwymi etykietami
  - Otrzymane wyniki były porównywalne do otrzymanych z modelu BERT

```
F1: 0.7892922844235707
Accuracy: 0.853
Precision: 0.7783611670020122
Recall: 0.853
```

Czemu DistillBERT?

Tips:
•DistilBERT doesn't have `token_type_ids`, you don't need to indicate which token belongs to which segment. Just separate your segments with the separation token `tokenizer.sep_token` (or `[SEP]`).
•DistilBERT doesn't have options to select the input positions (`position_ids` input). This could be added if necessary though, just let us know if you need this option.
•Same as BERT but smaller. Trained by distillation of the pretrained BERT model, meaning it's been trained to predict the same probabilities as the larger model. The actual objective is a combination of:
    •finding the same probabilities as the teacher model
    •predicting the masked tokens correctly (but no next-sentence objective)
    •a cosine similarity between the hidden states of the student and the teacher model

This model was contributed by victorsanh. This model jax version was contributed by kamalkraj. The original code can be found here.

Nie można jednoznacznie stwierdzić, że DistilBERT jest lepszy od Herberta w kontekście klasyfikacji cyberbullyingu, ponieważ ostateczna skuteczność modelu zależy od wielu czynników, takich jak zbiór danych, architektura modelu, hiperparametry, itp.
Herbert to model BERT dostosowany przez Allegro na dużym polskim korpusie tekstów. Jest szkolony na szerokim spektrum zadań, w tym na zadaniami, które wymagają rozumienia kontekstu społecznego i emocji, co może być przydatne w przypadku analizy cyberbullyingu.
Z drugiej strony, DistilBERT to zoptymalizowana wersja BERTa, która ma mniejszą liczbę parametrów, co przekłada się na mniejsze wymagania obliczeniowe i szybsze działanie. Może być bardziej wydajny w przypadku zastosowań, gdzie czas i zasoby obliczeniowe są ograniczone.
W praktyce warto eksperymentować z różnymi modelami, takimi jak Herbert, DistilBERT, a nawet inne architektury, aby znaleźć ten, który najlepiej radzi sobie z danym problemem klasyfikacji cyberbullyingu na konkretnym zbiorze danych. Warto również dostosować hiperparametry i dobrze przygotować zbiór treningowy, aby osiągnąć jak najlepsze wyniki.

4 Wyniki

5 Wnioski