Выполняла: Акимова Анастасия

Цель работы:

Построить и обучить модель, которая по тексту определяет, какому автору он принадлежит, используя датасет writerextnd с платформы Kaggle.

Этапы работы:

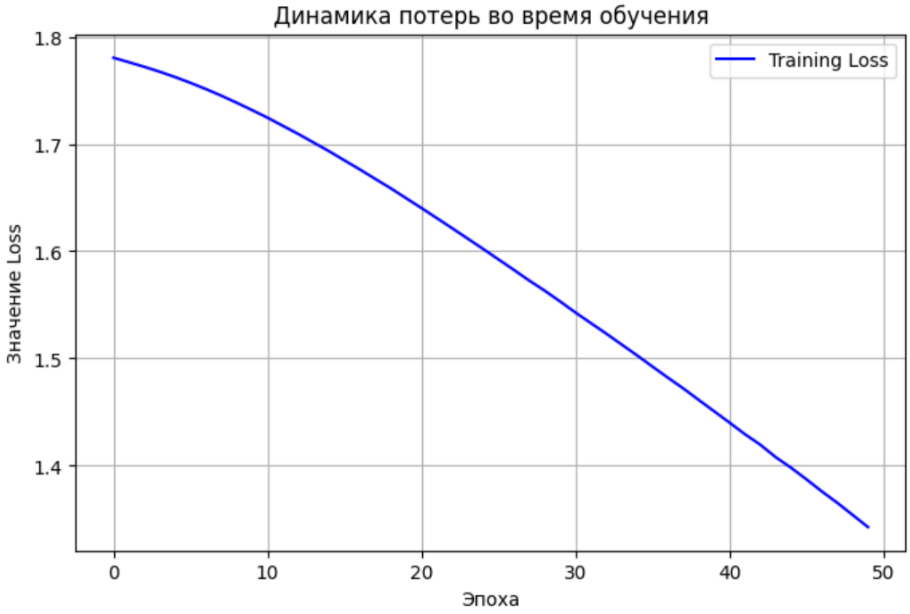
1) Подготовка данных. Я загрузила архив данных с сайта Kaggle и разархивировала его. Затем объединила все тексты авторов в одну таблицу для дальнейшей обработки. При анализе текстов у каждого автора, я выяснила, что у многих авторов очень мало записей и они не подходят для обучения модели, поэтому я оставила только тех авторов, на которых модель может хорошо обучится. В конце этого этапа я произвела очистку тестов (все тексты были приведены к нижнему регистру, удалены все лишние символы и знаки препинания), чтобы одинаковые слова не воспринимались как разные из за регистра.
2) Преобразование текста в числовой формат. Для преобразования текста в числовой формат я выбрала метод TF-IDF, так как он хорошо работает с текстами средней длины и позволяет учитывать частоту слов и их уникальность. Я ограничила количество признаков до 5000, чтобы модель обучалась быстрее, но при этом сохраняла смысл. После векторизации, я разделила данные на обучающую и тестовую выборку в соотношении 80\20.
3) Построение модели. Для классификации я выбрала простую полносвязную нейросеть, поскольку задача — базовая многоклассовая классификация, и сложные архитектуры не нужны на данном этапе. Входной слой: 5000 признаков TF-IDF. Скрытый слой: 128 нейронов, активация ReLU — для нелинейности модели. Dropout: 30% для снижения переобучения. Выходной слой: 6 классов авторов, с активацией softmax (через CrossEntropyLoss). Такая архитектура позволяет построить быструю базовую модель для многоклассовой классификации.
4) Обучение модели. Я использовала функцию потерь CrossEntropyLoss, так как задача многоклассовая. Для оптимизации — Adam, скорость обучения выбрана 0.001 для постепенного и стабильного обучения. Число эпох я выбрала 50, определила это экспериментальным путём: я начала с меньшего количества эпох, но увидела, что увеличение до 50 даёт улучшение качества модели.
5) Оценка модели. После обучения, я оценила модель на тестовой выборке. Итоговая точность составила 0.5507, а F1-score — 0.4956. Базовая модель решает задачу лучше случайного угадывания, но требует дальнейшего улучшения.

Выводы и рекомендации по улучшению модели:

Модель обучилась не очень хорошо, итоговая точность (accuracy) только 0.5507. Возможные причины: простая архитектура модели и маленький объем данных. Модель обучилась бы лучше на большей обучающей выборке.

Графики, визуализации и интерпретации ошибок:

1) <u>График обучения (Loss function)</u>
   На графике видно, как функция потерь постепенно снижается с каждой эпохой, это говорит о том, что модель обучается.



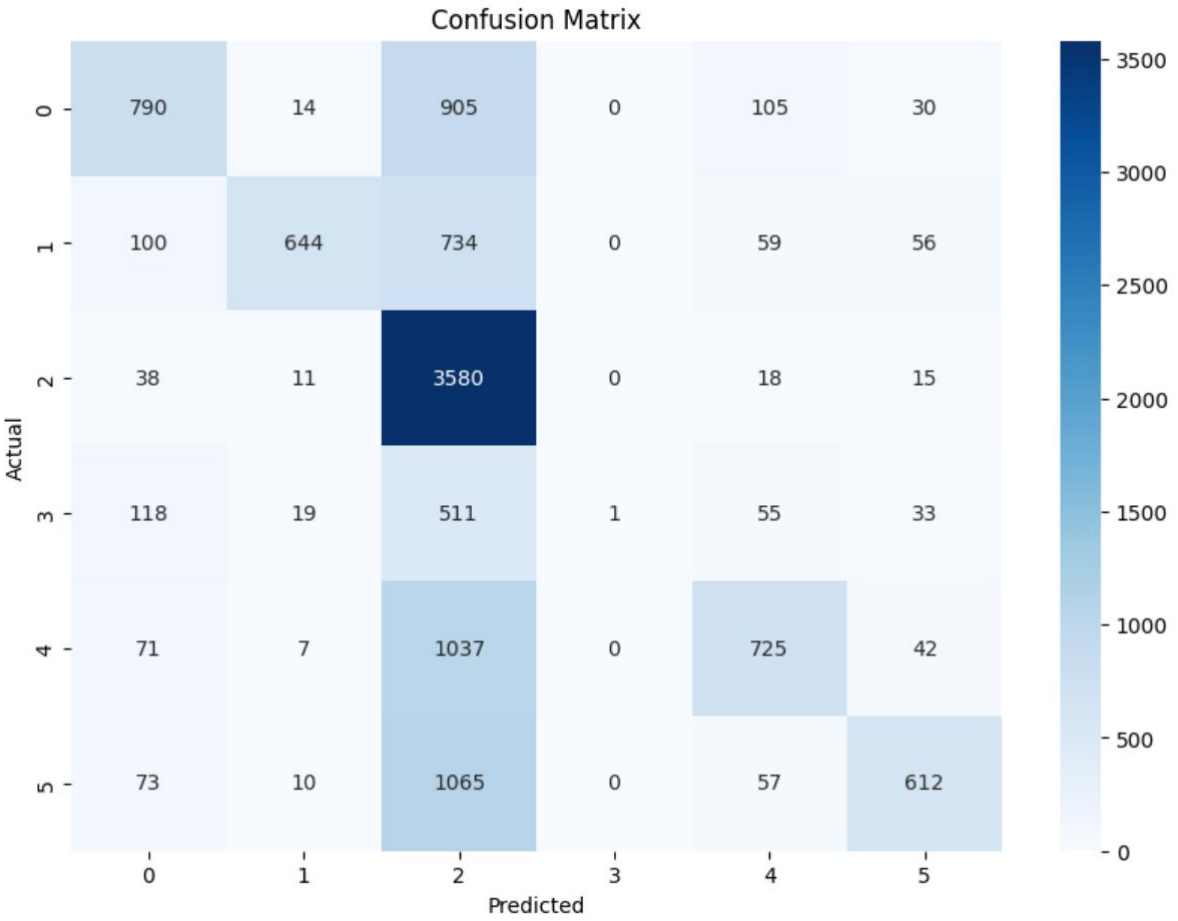
2) <u>Метрики качества модели на тестовой выборке</u>
   Модель достигла точности 0.5507 и F1-score 0.4956. Видно, что некоторые классы определяются моделью лучше других.

```
Accuracy: 0.5507
F1-score: 0.5142

Classification Report:
              precision    recall  f1-score   support

           0       0.66      0.43      0.52      1844
           1       0.91      0.40      0.56      1593
           2       0.46      0.98      0.62      3662
           3       1.00      0.00      0.00       737
           4       0.71      0.39      0.50      1882
           5       0.78      0.34      0.47      1817

    accuracy                           0.55     11535
   macro avg       0.75      0.42      0.45     11535
weighted avg       0.68      0.55      0.51     11535
```


Confusion Matrix
Actual
Predicted
0
1
2
3
4
5
790
14
905
0
105
30
100
644
734
0
59
56
38
11
3580
0
18
15
118
19
511
1
55
33
71
7
1037
0
725
42
73
10
1065
0
57
612
3500
3000
2500
2000
1500
1000
500
0