Mini Project 6

Group members: Alireza Hodaei(610397164), Sepehr Abbaspour(610398147), Amir Mohammad Ramezan Naderi(610398126), Hossein Nazari(610398179)

1. Introduction:

Author identification in Persian literature is a challenging task with significant implications for understanding the unique writing styles and patterns of individual authors. The primary objective of this mini-project is to harness the power of BERT (Bidirectional Encoder Representations from Transformers) models for accurate author identification in Persian literary works. BERT's bidirectional nature, allowing it to capture intricate relationships within sentences, is key to unraveling the nuances of diverse writing styles.

In this project, BERT is employed to fine-tune on a diverse dataset, where authors are associated with specific thematic focuses, ranging from cars to cinema. BERT's contextual understanding adapts to the nuances of authors' language usage, enabling precise identification within the rich and varied landscape of Persian literature. This report provides an overview of dataset construction techniques, model selection, the fine-tuning process, experimental results, and a comparison with traditional ML approaches, and concludes with insights gained and potential avenues for future work. Through the integration of BERT, this project aims to contribute to a deeper understanding of authorship in the context of Persian literature.

2. Dataset Construction Techniques:

Process of Dataset Creation:

The dataset was thoughtfully compiled to cover a variety of subjects within Persian literature. Each author was assigned a distinct thematic focus to ensure variety and representation across different genres. The following themes were allocated to the authors:

Dehghan: Writing about cars.

Ashtari: Writing about cars.

Farid Matin: Focused on cinema.

Saber Rastikerdar: Specializing in Persian fonts.

Mohammad Dehghani: Engaging with language models.

Elham Hesaraki: Concentrating on machine learning and AI.

Behzad Bahramijoo: Discussing games.

Reza Hajmohammadi: Centered around cinema.

Sheikhi: Delving into technology.

Zahedi: Exploring the world of games.

Challenges and Solutions:

Creating a well-balanced dataset presented certain challenges, especially regarding the availability of content for each specific theme. Some authors may have had more extensive publications on their assigned topics compared to others. To address this, a careful selection of sources was undertaken to ensure a representative dataset for each author.

Rationale Behind Genre Selection:

The chosen genres were selected to reflect the diverse interests of the authors while aligning with the broader themes in Persian literature. This approach aimed to capture distinct writing styles and language nuances related to specific subjects.

Uniform Document Length:

To maintain consistency and comparability across the dataset, each document was set at around 500 words. This decision ensures that the models are trained on a standardized length, preventing biases associated with varying document lengths.

Content Curation:

The thematic focus for each author was carefully assigned based on their known expertise or published works. This process involved an in-depth review of existing literature and online content associated with each author.

By assigning specific themes to each author, the dataset construction process not only represents a diverse range of topics in Persian literature but also addresses challenges related to content availability and consistency. This thematic approach allows for a nuanced exploration of author writing styles within distinct genres.

3. Model Selection and Fine-Tuning:

Choice of Model and Reasoning:

For the author identification task, BERT models from Hugging Face were chosen for their proven effectiveness in capturing contextual information within text. This aligns seamlessly with the bidirectional nature of the task, where understanding both preceding and succeeding words is crucial for accurate identification.

The model chosen here is ParsBERT or "HooshvareLab/bert-base-parsebert-uncased". ParsBERT is a monolingual language model based on Google's BERT architecture with the same configurations as BERT-Base. The model is pre-trained on a large Persian corpus with various writing styles from numerous subjects (e.g., scientific, novels) with more than 2 million documents. As a part of ParsBERT methodology, an extensive pre-processing combining POS tagging and WordPiece segmentation was carried out to bring the corpus into a proper format. This process produces more than 40M true sentences.

In addition to Hugging Face's BERT models, ParsBERT, a transformer-based model designed explicitly for Persian language understanding, was incorporated. ParsBERT is introduced in the paper titled "ParsBERT: Transformer-based Model for Persian Language Understanding" by Mehrdad Farahani, Mohammad Gharachorloo, Marzieh Farahani, and Mohammad Manthouri [1]. This model specifically addresses the nuances of the Persian language, enhancing the project's capacity for author identification in Persian literature.

Additionally, hazm, a Python library tailored for Persian language processing, was utilized. Similar to NLTK but specialized for Persian, hazm enhances the project's capabilities in text analysis, tokenization, and other language processing tasks [2].

Parameters:
The reason behind some of the parameters is explained.
maxLength is equal to 512 which is the maximum number of sequential tokens in the documents.
num_labels is set to 10 as a result of having 10 authors which represent the documents.
the learning rate is set to 2e-5 which is what is most commonly used for BERT models.
epoch_num or number of epochs is set to 5 at first which is in the range of (3, 5) for the beginning, in relation to our dataset size and complexity. however, changing this number could affect the result of the validation.

Fine-Tuning Process:

The fine-tuning process involved careful adjustments to hyperparameters, including learning rate and batch size. It was essential to strike a balance between adapting the model to the specifics of the author identification task and preserving the integrity of the pre-trained BERT models.

ParsBERT and hazm underwent fine-tuning on our diverse dataset to tailor their pre-trained knowledge to the nuances of Persian literature. Similar considerations were applied, ensuring effective adaptation without compromising the inherent strengths of the original models.

Architecture and Parameters:

The base model selected for fine-tuning was BERT-base-uncased, chosen for its balance between performance and computational efficiency. Key parameters such as learning rate, batch size, and the number of training epochs were fine-tuned to optimize model performance in the context of author identification.

In summary, the combined use of Hugging Face's BERT models, ParsBERT, and hazm, fine-tuned with thoughtful adjustments to hyperparameters, ensures an effective and contextually aware approach to author identification in Persian literature.

Explaining code and results:

Text preprocessing:

Here we define several text preprocessing functions:
- rm_link: The function removes hyperlinks from the text.

- rm_punct2: This function removes punctuation marks from the text.
- rm_html: This function removes HTML tags from the text.
- space_bt_punct: The function adds spaces between punctuation marks and words.
- rm_number: The function removes numbers from the text.
- rm_whitespaces: The function removes extra whitespaces from the text.
- rm_nonascii: Here the function removes non-ASCII characters from the text.
- spell_correction: This function is used for checking the correct spelling given a text
- clean_pipeline: This function combines all the above preprocessing steps into a single pipeline and produces the final processed result given an input text.

Next, the code reads stopwords from the 'stopwords.txt' file and stores them in a list.
Following that, the 'preprocess2' function tokenizes the text using word_tokenize from hazm and removes stopwords. It then stems the words using a stemmer from the hazm library, the modified NLTK library for persian language.
The, the 'remove_point' function removes specific characters such as '\u200c' and '•' from the text and normalize it.

Training:

After preprocessing the texts and embedding them like the BERT model, we set configs for the model. In the following, we set some parameters for the training part. See the first parameter below:

MAX_LEN = 512

TRAIN_BATCH_SIZE = 8

VALID_BATCH_SIZE = 4

EPOCHS = 5

LEARNING_RATE = 2e-5

Now see how we build and compile the model:

TFBertForSequenceClassification.from_pretrained: It initializes a BERT model for sequence classification using the specified pre-trained model and configuration (config). This function is part of the Hugging Face transformers library for TensorFlow.

optimizer = tf.keras.optimizers.Adam(learning_rate=learning_rate): This line creates an Adam optimizer with the specified learning rate. Adam is a popular optimization algorithm commonly used for training neural networks.

loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True): This line defines the loss function for the model.

SparseCategoricalCrossentropy is suitable for classification tasks with integer labels. The argument from_logits=True indicates that the model's output is logits, and the softmax activation will be applied during training.

metric = tf.keras.metrics.SparseCategoricalAccuracy('accuracy'): This line defines the evaluation metric for the model. In this case, it uses the sparse categorical accuracy, which is suitable for integer class labels.

Evaluation:

At the end of this part, we did 5-fold cross-validation on the dataset with parameters. See the result:

1- First result:

MAX_LEN = 512

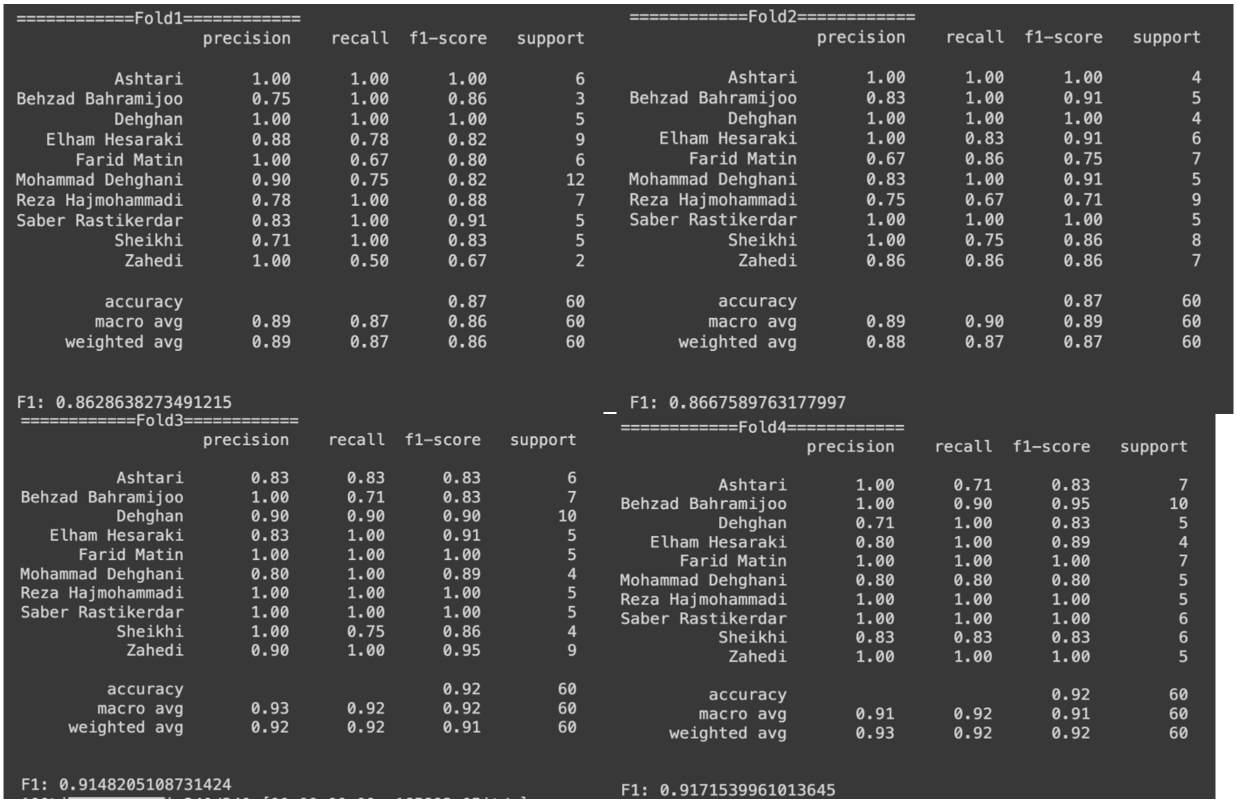
TRAIN_BATCH_SIZE = 8

VALID_BATCH_SIZE = 4

EPOCHS = 5

LEARNING_RATE = 2e-5

We set the parameters like this and see the result:

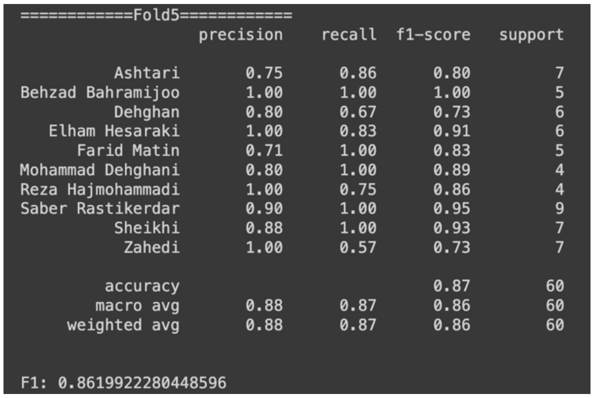


```
============Fold1============
                   precision    recall  f1-score   support

           Ashtari      1.00      1.00      1.00         6
 Behzad Bahramijoo      0.75      1.00      0.86         3
           Dehghan      1.00      1.00      1.00         5
    Elham Hesaraki      0.88      0.78      0.82         9
       Farid Matin      1.00      0.67      0.80         6
 Mohammad Dehghani      0.90      0.75      0.82        12
 Reza Hajmohammadi      0.78      1.00      0.88         7
 Saber Rastikerdar      0.83      1.00      0.91         5
           Sheikhi      0.71      1.00      0.83         5
            Zahedi      1.00      0.50      0.67         2

          accuracy                          0.87        60
         macro avg      0.89      0.87      0.86        60
      weighted avg      0.89      0.87      0.86        60


F1: 0.8628638273491215
```

```
============Fold2============
                   precision    recall  f1-score   support

           Ashtari      1.00      1.00      1.00         4
 Behzad Bahramijoo      0.83      1.00      0.91         5
           Dehghan      1.00      1.00      1.00         4
    Elham Hesaraki      1.00      0.83      0.91         6
       Farid Matin      0.67      0.86      0.75         7
 Mohammad Dehghani      0.83      1.00      0.91         5
 Reza Hajmohammadi      0.75      0.67      0.71         9
 Saber Rastikerdar      1.00      1.00      1.00         5
           Sheikhi      1.00      0.75      0.86         8
            Zahedi      0.86      0.86      0.86         7

          accuracy                          0.87        60
         macro avg      0.89      0.90      0.89        60
      weighted avg      0.88      0.87      0.87        60


F1: 0.8667589763177997
```

```
============Fold3============
                   precision    recall  f1-score   support

           Ashtari      0.83      0.83      0.83         6
 Behzad Bahramijoo      1.00      0.71      0.83         7
           Dehghan      0.90      0.90      0.90        10
    Elham Hesaraki      0.83      1.00      0.91         5
       Farid Matin      1.00      1.00      1.00         5
 Mohammad Dehghani      0.80      1.00      0.89         4
 Reza Hajmohammadi      1.00      1.00      1.00         5
 Saber Rastikerdar      1.00      1.00      1.00         5
           Sheikhi      1.00      0.75      0.86         4
            Zahedi      0.90      1.00      0.95         9

          accuracy                          0.92        60
         macro avg      0.93      0.92      0.92        60
      weighted avg      0.92      0.92      0.91        60


F1: 0.9148205108731424
```

```
============Fold4============
                   precision    recall  f1-score   support

           Ashtari      1.00      0.71      0.83         7
 Behzad Bahramijoo      1.00      0.90      0.95        10
           Dehghan      0.71      1.00      0.83         5
    Elham Hesaraki      0.80      1.00      0.89         4
       Farid Matin      1.00      1.00      1.00         7
 Mohammad Dehghani      0.80      0.80      0.80         5
 Reza Hajmohammadi      1.00      1.00      1.00         5
 Saber Rastikerdar      1.00      1.00      1.00         6
           Sheikhi      0.83      0.83      0.83         6
            Zahedi      1.00      1.00      1.00         5

          accuracy                          0.92        60
         macro avg      0.91      0.92      0.91        60
      weighted avg      0.93      0.92      0.92        60


F1: 0.9171539961013645
```



```
============Fold5============
                   precision    recall  f1-score   support

           Ashtari       0.75      0.86      0.80         7
 Behzad Bahramijoo       1.00      1.00      1.00         5
           Dehghan       0.80      0.67      0.73         6
    Elham Hesaraki       1.00      0.83      0.91         6
       Farid Matin       0.71      1.00      0.83         5
 Mohammad Dehghani       0.80      1.00      0.89         4
Reza Hajmohammadi       1.00      0.75      0.86         4
 Saber Rastikerdar       0.90      1.00      0.95         9
           Sheikhi       0.88      1.00      0.93         7
            Zahedi       1.00      0.57      0.73         7

          accuracy                           0.87        60
         macro avg       0.88      0.87      0.86        60
      weighted avg       0.88      0.87      0.86        60


F1: 0.8619922280448596
```

As you can see the mean of f1-score for this model with those parameters is **88.4** which is good. You can see the precision, recall, accuracy, and the f1-score for each fold.

2- second result:

MAX_LEN = 128

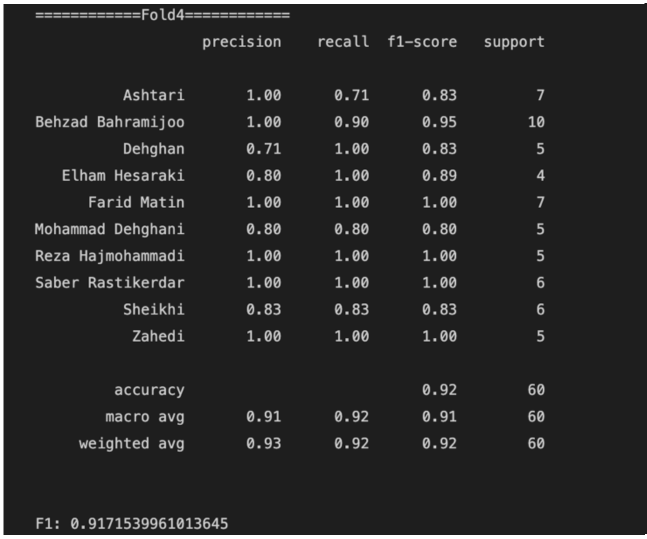
TRAIN_BATCH_SIZE = 8

VALID_BATCH_SIZE = 4

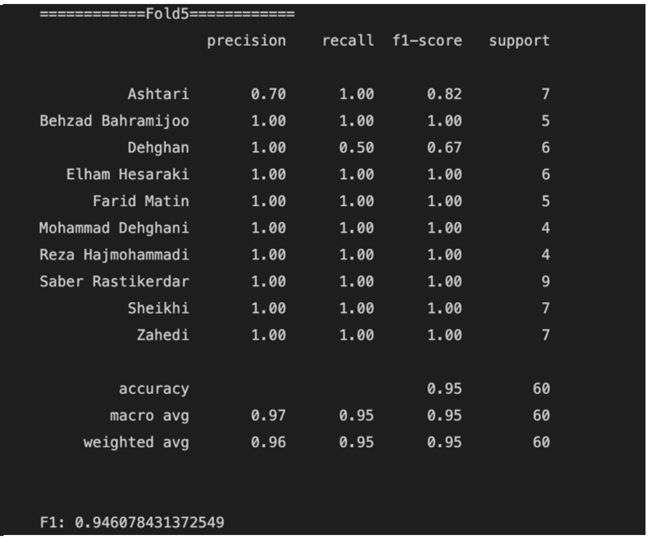
EPOCHS = 5

LEARNING_RATE = 2e-5

For the simplicity, we put the last two folds:



```
============Fold4============
                   precision    recall  f1-score   support

           Ashtari       1.00      0.71      0.83         7
 Behzad Bahramijoo       1.00      0.90      0.95        10
           Dehghan       0.71      1.00      0.83         5
    Elham Hesaraki       0.80      1.00      0.89         4
       Farid Matin       1.00      1.00      1.00         7
 Mohammad Dehghani       0.80      0.80      0.80         5
 Reza Hajmohammadi       1.00      1.00      1.00         5
 Saber Rastikerdar       1.00      1.00      1.00         6
           Sheikhi       0.83      0.83      0.83         6
            Zahedi       1.00      1.00      1.00         5

          accuracy                           0.92        60
         macro avg       0.91      0.92      0.91        60
      weighted avg       0.93      0.92      0.92        60


F1: 0.9171539961013645
```



```
============Fold5============
                   precision    recall  f1-score   support

           Ashtari       0.70      1.00      0.82         7
 Behzad Bahramijoo       1.00      1.00      1.00         5
           Dehghan       1.00      0.50      0.67         6
    Elham Hesaraki       1.00      1.00      1.00         6
       Farid Matin       1.00      1.00      1.00         5
 Mohammad Dehghani       1.00      1.00      1.00         4
 Reza Hajmohammadi       1.00      1.00      1.00         4
 Saber Rastikerdar       1.00      1.00      1.00         9
           Sheikhi       1.00      1.00      1.00         7
            Zahedi       1.00      1.00      1.00         7

          accuracy                           0.95        60
         macro avg       0.97      0.95      0.95        60
      weighted avg       0.96      0.95      0.95        60


F1: 0.946078431372549
```

The mean of f1-score for this model is **91.5.** so we can realize that the smaller number of **max-len** is better for the accuracy.

3- Third result:

MAX_LEN = 128

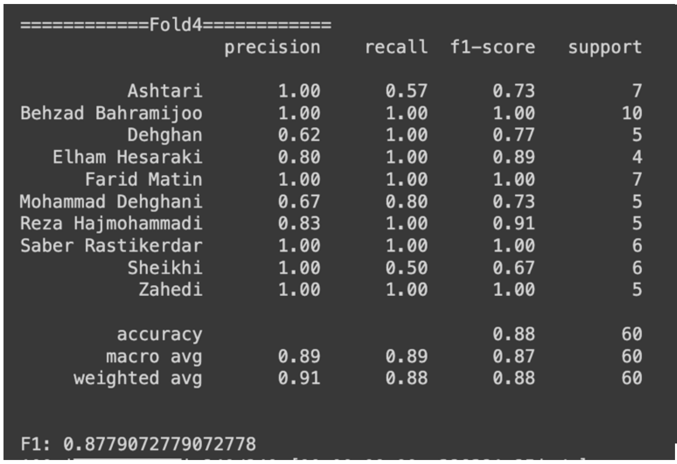
TRAIN_BATCH_SIZE = 16

VALID_BATCH_SIZE = 8

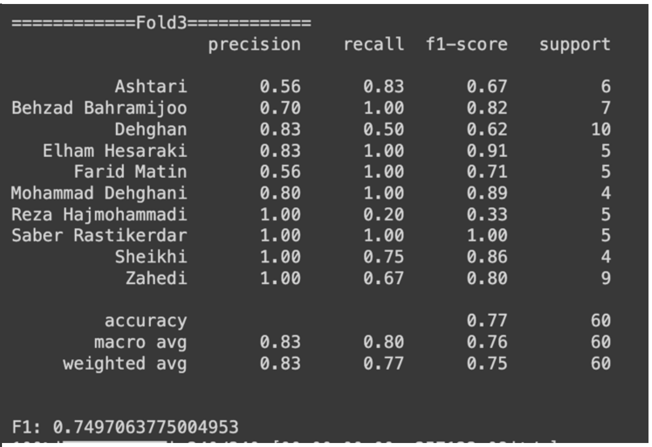
EPOCHS = 5

LEARNING_RATE = 2e-5

In this part, we change the batch size of the train and validation batch size to 16 and 8 respectively. Now let’s see the result:



```
============Fold4============
                    precision    recall  f1-score   support

           Ashtari       1.00      0.57      0.73         7
 Behzad Bahramijoo       1.00      1.00      1.00        10
           Dehghan       0.62      1.00      0.77         5
     Elham Hesaraki      0.80      1.00      0.89         4
        Farid Matin      1.00      1.00      1.00         7
 Mohammad Dehghani       0.67      0.80      0.73         5
 Reza Hajmohammadi       0.83      1.00      0.91         5
 Saber Rastikerdar       1.00      1.00      1.00         6
           Sheikhi       1.00      0.50      0.67         6
            Zahedi       1.00      1.00      1.00         5

          accuracy                           0.88        60
         macro avg       0.89      0.89      0.87        60
      weighted avg       0.91      0.88      0.88        60


F1:  0.8779072779072778
```



```
============Fold3============
                    precision    recall  f1-score   support

           Ashtari       0.56      0.83      0.67         6
 Behzad Bahramijoo       0.70      1.00      0.82         7
           Dehghan       0.83      0.50      0.62        10
     Elham Hesaraki      0.83      1.00      0.91         5
        Farid Matin      0.56      1.00      0.71         5
 Mohammad Dehghani       0.80      1.00      0.89         4
 Reza Hajmohammadi       1.00      0.20      0.33         5
 Saber Rastikerdar       1.00      1.00      1.00         5
           Sheikhi       1.00      0.75      0.86         4
            Zahedi       1.00      0.67      0.80         9

          accuracy                           0.77        60
         macro avg       0.83      0.80      0.76        60
      weighted avg       0.83      0.77      0.75        60


F1:  0.7497063775004953
```

The mean of f1-score for these parameters of this model is **84.6** which is lower than the previous part. So we realize that the number of batches should be relevant to the number of the dataset records. The smaller the batch sizes are, the higher the accuracy.

4- Fourth result(No stopwords):

MAX_LEN = 128

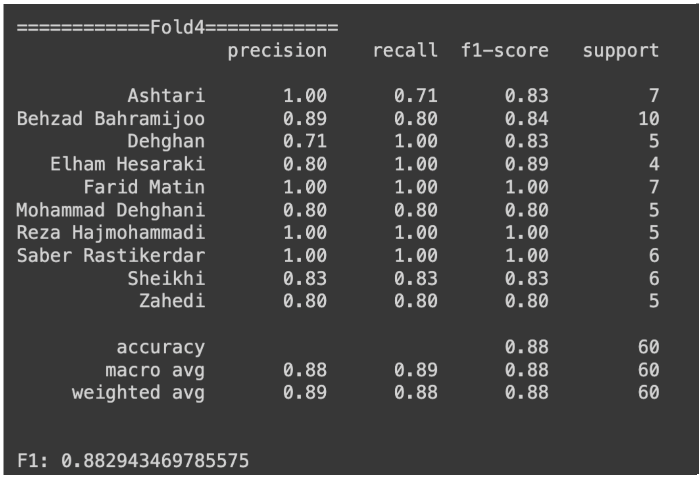
TRAIN_BATCH_SIZE = 16

VALID_BATCH_SIZE = 8

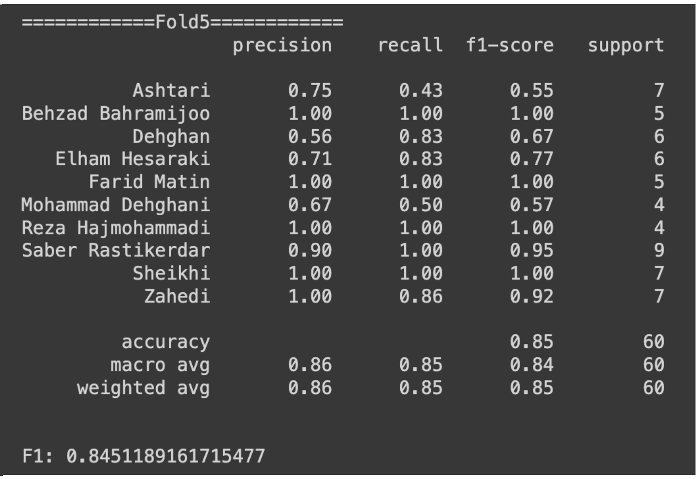
EPOCHS = 5

LEARNING_RATE = 2e-5

In this part, we remove applying the stopwords. Let’s see the result:



```
============Fold4============
                   precision    recall  f1-score   support

          Ashtari       1.00      0.71      0.83         7
Behzad Bahramijoo       0.89      0.80      0.84        10
          Dehghan       0.71      1.00      0.83         5
   Elham Hesaraki       0.80      1.00      0.89         4
      Farid Matin       1.00      1.00      1.00         7
 Mohammad Dehghani      0.80      0.80      0.80         5
Reza Hajmohammadi       1.00      1.00      1.00         5
 Saber Rastikerdar      1.00      1.00      1.00         6
           Sheikhi      0.83      0.83      0.83         6
            Zahedi      0.80      0.80      0.80         5

         accuracy                           0.88        60
        macro avg       0.88      0.89      0.88        60
     weighted avg       0.89      0.88      0.88        60


F1:  0.882943469785575
```



```
============Fold5============
                   precision    recall  f1-score   support

          Ashtari       0.75      0.43      0.55         7
Behzad Bahramijoo       1.00      1.00      1.00         5
          Dehghan       0.56      0.83      0.67         6
   Elham Hesaraki       0.71      0.83      0.77         6
      Farid Matin       1.00      1.00      1.00         5
 Mohammad Dehghani      0.67      0.50      0.57         4
Reza Hajmohammadi       1.00      1.00      1.00         4
 Saber Rastikerdar      0.90      1.00      0.95         9
           Sheikhi      1.00      1.00      1.00         7
            Zahedi      1.00      0.86      0.92         7

         accuracy                           0.85        60
        macro avg       0.86      0.85      0.84        60
     weighted avg       0.86      0.85      0.85        60


F1:  0.8451189161715477
```

The mean of f1-score for this part is 85.2 which is good. The effect of stopwords for this dataset and this BERT model is not noticeable.

5- Fourth result(Learning rate):

MAX_LEN = 128

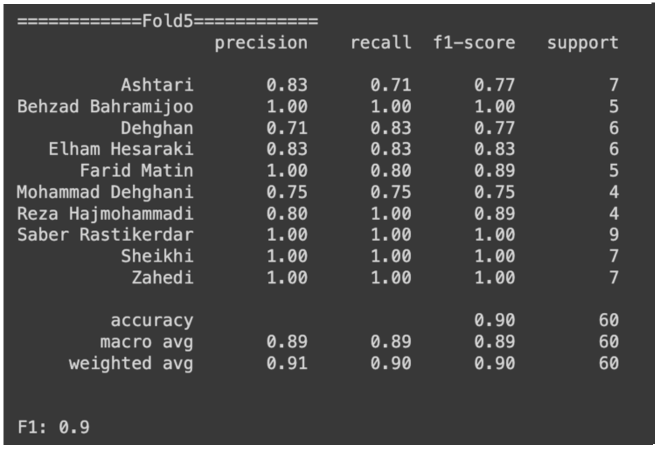
TRAIN_BATCH_SIZE = 16

VALID_BATCH_SIZE = 8

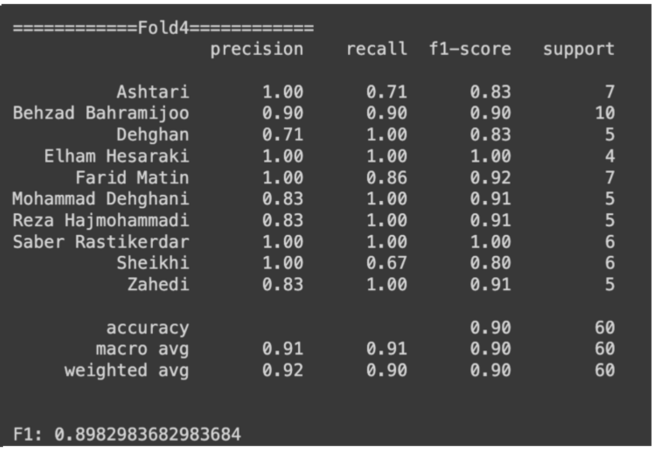
EPOCHS = 5

LEARNING_RATE = 2e-5 * 1.8

n this part, we highr the learning rate. Let’s see the result:



```
============Fold5============
                   precision    recall  f1-score   support

          Ashtari       0.83      0.71      0.77         7
Behzad Bahramijoo       1.00      1.00      1.00         5
          Dehghan       0.71      0.83      0.77         6
   Elham Hesaraki       0.83      0.83      0.83         6
      Farid Matin       1.00      0.80      0.89         5
 Mohammad Dehghani      0.75      0.75      0.75         4
Reza Hajmohammadi       0.80      1.00      0.89         4
 Saber Rastikerdar      1.00      1.00      1.00         9
           Sheikhi      1.00      1.00      1.00         7
            Zahedi      1.00      1.00      1.00         7

         accuracy                           0.90        60
        macro avg       0.89      0.89      0.89        60
     weighted avg       0.91      0.90      0.90        60


F1:  0.9
```



```
============Fold4============
                   precision    recall  f1-score   support

          Ashtari       1.00      0.71      0.83         7
Behzad Bahramijoo       0.90      0.90      0.90        10
          Dehghan       0.71      1.00      0.83         5
   Elham Hesaraki       1.00      1.00      1.00         4
      Farid Matin       1.00      0.86      0.92         7
 Mohammad Dehghani      0.83      1.00      0.91         5
Reza Hajmohammadi       0.83      1.00      0.91         5
 Saber Rastikerdar      1.00      1.00      1.00         6
           Sheikhi      1.00      0.67      0.80         6
            Zahedi      0.83      1.00      0.91         5

         accuracy                           0.90        60
        macro avg       0.91      0.91      0.90        60
     weighted avg       0.92      0.90      0.90        60


F1:  0.8982983682983684
```

the mean of f1-score for this part is **89.8** which is good. The point is in this training, the model tends to have an overfitting problem. Because we got 100% accuracy in the training dataset and the same accuracy as in the previous parts.

At the end the best result was obtained by these parameters:

```
MAX_LEN = 128

TRAIN_BATCH_SIZE = 8

VALID_BATCH_SIZE = 4

EPOCHS = 5

LEARNING_RATE = 2e-5
```

Where the mean of f1-score in 5-fold cross-validation was 91.5.

In summary, while traditional machine learning methods rely on handcrafted features and may struggle with complex tasks like natural language understanding, the BERT approach leverages deep learning techniques to learn representations directly from raw data, leading to improved performance on various NLP tasks. However, this comes at the cost of increased computational resources and reduced interpretability. The choice between traditional ML methods and deep learning approaches depends on factors such as task complexity, available resources, and the need for interpretability.

References:

[1] Mehrdad Farahani, Mohammad Gharachorloo, Marzieh Farahani, Mohammad Manthouri. "ParsBERT: Transformer-based Model for Persian Language Understanding." ArXiv, 2020. Link

[2] "hazm: A Python library for digesting Persian text." PyPI. Link