Results

Aritz Bercher

December 19, 2020

Abstract

Contains results of different models for the task of predicting tags (multi-label classification) on stack overflow posts.

- For the pipeline

  1. gensim removal html tags
  2. gensim removal punctuation,
  3. gensim removal multiple whitespaces
  4. gensim removal numerics
  5. gensim removal stop words
  6. gensim removal words shorter than a threshold (len ≤ 4)
  7. gensim text stemming
  8. **Doc2Vec** embedding
  9. Sklearn **OneVSRest** with **Logistic Regression** (max iter = 4000) as base classifier

  I obtained the following results:

  ```
  accuracy =  0.0
  F1 score macro =  9.253042371415962e-06
  F1 score micro =  0.0006574351810001233
  F1 score weighted =  0.0006466435361030257
  Recall score =  4.708375022070508e-06
  Hamming loss = 0.002187148941956245
  Jaccard score = 4.654667322198981e-06
  ```

  The vectorization of the training set took 460 seconds and the training of the classifier took 680 seconds.

- For the pipeline

  1. gensim removal html tags

2. gensim removal punctuation,
3. gensim removal multiple whitespaces
4. gensim removal numerics
5. gensim removal stop words
6. gensim removal words shorter than a threshold (len ≤ 4)
7. gensim text stemming
8. sklearn **bag of word**

   ```
   CountVectorizer(analyzer=’word’, ngram_range=(1, 1), binary=True, token_pattern=
   ```

9. Sklearn **OneVSRest** with **Logistic Regression** (max iter = 4000) as base classifier

I obtained the following results:

```
accuracy =  0.041255289139633285
F1 score macro =  0.11847502879063147
F1 score micro =  0.3807525753785739
F1 score weighted =  0.34937882588009894
Recall score =  0.09317980831186831
Hamming loss = 0.002070421861381466
Jaccard score = 0.07776028480455001
```

- With the following pipeline and training only on the first 1’000 samples of the dataset (because of computational limitations):

  1. gensim removal html tags
  2. gensim removal punctuation,
  3. gensim removal multiple whitespaces
  4. gensim removal numerics
  5. gensim removal stop words
  6. gensim removal words shorter than a threshold (len ≤ 4)
  7. gensim text stemming
  8. sklearn **bag of word**

     ```
     CountVectorizer(analyzer=’word’, ngram_range=(1, 1), binary=True, token_pattern=
     ```

  9. **scikit multi-learn MLARAM**(threshold=0.05, vigilance=0.95)

  I obtain the following scores:

```
accuracy =  0.006346967559943582
F1 score macro =  0.00017462757006054923
F1 score micro =  0.0667865086377624
F1 score weighted =  0.01028869759140275
Recall score =  0.0007651109410864575
Hamming loss = 0.0027544173735939243
Jaccard score = 9.856154107084596e-05
```

- For the following pipeline:

    1. gensim removal html tags
    2. gensim removal punctuation,
    3. gensim removal multiple whitespaces
    4. gensim removal numerics
    5. gensim removal stop words
    6. gensim removal words shorter than a threshold (len ≤ 4)
    7. gensim text stemming
    8. sklearn **bag of word**

       `CountVectorizer(analyzer='word', ngram_range=(1, 1), binary=True, token_pattern=`
    9. Sklearn **OneVSRest** with **Logistic Regression** (max iter = 4000) as base classifier trained **only on the labels having more than 100 samples associated to them**:

```
accuracy =  0.03984485190409027
F1 score macro =  0.09149156071331765
F1 score micro =  0.3757963510179738
F1 score weighted =  0.34071957714760054
Recall score =  0.07337756322745101
Hamming loss = 0.0020580049016716093
Jaccard score = 0.05946561565904122
```

with a training time of 1344 seconds.

- For the following pipeline:

    1. gensim removal html tags
    2. gensim removal punctuation,
    3. gensim removal multiple whitespaces
    4. gensim removal numerics
    5. gensim removal stop words

6. gensim removal words shorter than a threshold (len ≤ 4)
7. gensim text stemming
8. **Doc2Vec** embeddings

   ```
   CountVectorizer(analyzer='word', ngram_range=(1, 1), binary=True, token_pattern=
   ```

9. Sklearn **OneVSRest** with **Logistic Regression** (max iter = 4000) as base classifier trained **only on the labels having more than 100 samples associated to them**:

```
accuracy =  0.0
F1 score macro =  9.196750098433964e-06
F1 score micro =  0.0006572731380684385
F1 score weighted =  0.0006464839957758486
Recall score =  4.6797309154723605e-06
Hamming loss = 0.0021743793808019648
Jaccard score = 4.626349954459367e-06
```

with a training time of 255 seconds.

- For the following pipeline (Keeping only tags with more than 100 samples associated to them):

  1. gensim removal html tags
  2. gensim removal punctuation,
  3. gensim removal multiple whitespaces
  4. gensim removal numerics
  5. gensim removal stop words
  6. gensim removal words shorter than a threshold (len ≤ 4)
  7. gensim text stemming
  8. keep only 5'000 most common words
  9. **Trainable embedding** layer (size = 100) initialized with GloVe embeddings
  10. drop out layer ($p = 0.1$)
  11. **1D convolutional** layer with 225 filters of size 3, relu activation, and stride 1.
  12. Max pooling on time dimension
  13. Dense layer
  14. Sigmoid activation
  15. Keras callback functions were
      - ReduceLROnPlateau

  - EarlyStopping with patience= 10

The results are

```
accuracy =  0.00011753643629525152
F1 score macro =  4.392006764721729e-05
  average, "true nor predicted", 'F-score is', len(true_sum)
F1 score micro =  0.00247080859203762
F1 score weighted =  0.002342487316193282
Recall score =  2.3693588415782534e-05
Hamming loss = 0.0022373038303471035
Jaccard score = 2.213801663434974e-05
```

- For the following pipeline (Keeping only tags with more than 100 samples associated to them):

  1. gensim removal html tags
  2. gensim removal punctuation,
  3. gensim removal multiple whitespaces
  4. gensim removal numerics
  5. gensim removal stop words
  6. gensim removal words shorter than a threshold (len ≤ 4)
  7. keep only 5'000 most common words
  8. **Trainable embedding** layer (size = 100) initialized with GloVe embeddings
  9. drop out layer ($p = 0.1$)
  10. **1D convolutional** layer with 225 filters of size 3, relu activation, and stride 1.
  11. Max pooling on time dimension
  12. Dense layer
  13. Sigmoid activation
  14. Keras callback functions were
      - ReduceLROnPlateau
      - EarlyStopping with patience= 10

  The results are

```
accuracy =  0.0
F1 score macro =  7.766140659917165e-05
F1 score micro =  0.0031293881644934803
F1 score weighted =  0.0029473517746327962
Recall score =  4.191321106691312e-05
Hamming loss = 0.002220857667397806
Jaccard score = 3.921709209300353e-05
```

- For the following pipeline (Keeping only tags with more than 100 samples associated to them):

  1. gensim removal html tags
  2. gensim removal punctuation,
  3. gensim removal multiple whitespaces
  4. gensim removal numerics
  5. gensim removal stop words
  6. gensim removal words shorter than a threshold (len ≤ 4)
  7. keep only 5'000 most common words
  8. **Trainable embedding** layer (size = 100) initialized randomly
  9. drop out layer ($p = 0.1$)
  10. **1D convolutional** layer with 225 filters of size 3, relu activation, and stride 1.
  11. Max pooling on time dimension
  12. Dense layer
  13. Sigmoid activation
  14. Keras callback functions were
      - ReduceLROnPlateau
      - EarlyStopping with patience= 10

  The results are

  ```
  accuracy =  0.0
  F1 score macro =  0.0
  F1 score micro =  0.0
  F1 score weighted =  0.0
  Recall score =  0.0
  Hamming loss = 0.002173664330238952
  Jaccard score = 0.0
  ```

- For the following pipeline (Keeping only tags with more than 100 samples associated to them):

  1. gensim removal html tags
  2. gensim removal punctuation,
  3. gensim removal multiple whitespaces
  4. gensim removal numerics
  5. gensim removal stop words

6. gensim removal words shorter than a threshold (len ≤ 4)
7. keep only 5'000 most common words
8. **Trainable embedding** layer (size = 100) initialized with GloVe embeddings
9. drop out layer ($p = 0.1$)
10. **1D convolutional layer** with 300 filters of size 3, relu activation, and stride 1.
11. Max pooling on time dimension
12. Dense layer
13. Sigmoid activation
14. Keras callback functions were
    - ReduceLROnPlateau
    - EarlyStopping with patience= 4

The results are

```
accuracy =  0.0
F1 score macro =  0.0
F1 score micro =  0.0
F1 score weighted =  0.0
Recall score =  0.0
Hamming loss = 0.002173664330238952
Jaccard score = 0.0
```

and the training time was 390.80 seconds.

- For the following pipeline (Keeping only tags with more than 100 samples associated to them):

1. gensim removal html tags
2. gensim removal punctuation,
3. gensim removal multiple whitespaces
4. gensim removal numerics
5. gensim removal stop words
6. gensim removal words shorter than a threshold (len ≤ 4)
7. keep only 5'000 most common words
8. **Trainable embedding layer** (size = 100) initialized with GloVe embeddings
9. drop out layer ($p = 0.1$)
10. **1D convolutional** layer with 300 filters of size 3, relu activation, and stride 1.

11. Max pooling on time dimension
12. Dense layer
13. Sigmoid activation
14. 30 epochs
15. Keras callback functions were
    - ReduceLROnPlateau

The results are

```
accuracy =  0.00047014574518l0061
F1 score macro =  0.00022281674111435695
F1 score micro =  0.009042209642490787
F1 score weighted =  0.007862811128499894
Recall score =  0.000130356731382182
Hamming loss = 0.0023313329793833047
Jaccard score = 0.00011396048553095816
```

and the training time was 1606.99 seconds.

- For the following pipeline (keeping only tags with more than 100 samples associated to them):

  1. gensim removal html tags
  2. gensim removal punctuation,
  3. gensim removal multiple whitespaces
  4. gensim removal numerics
  5. BERT tokenizer from hugging face:

     `BertTokenizer.from_pretrained('bert-base-uncased')`
  6. **Frozen** layer of **BERT** base uncased:

     `transformers.BertModel.from_pretrained('bert-base-uncased')`
  7. Drop out layer ($p = 0.3$)
  8. Feed forward layer with (# labels) output neurons

  The results for 1 epoch (it gets worse for more epochs) are

```
accuracy =  0.0
F1 score macro =  0.002916586699736247
F1 score micro =  0.012690280475422063
F1 score weighted =  0.03091273006062874
Recall score =  0.12495208830760628
Hamming loss = 0.12567103026272744
Jaccard score = 0.0014879615753613255
```

and the training time was 127.92 seconds and the prediction time was 3227.07 seconds