Языковые представления

Классификация на эмбеддингах

In
```
# ds — датасет
# model — модель классификации

import numpy as np
import pandas as pd
import torch
import transformers
from tqdm import notebook
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split

tokenizer = transformers.BertTokenizer(vocab_file='vocab.txt')
tokenized = df['text'].apply(lambda x: tokenizer.encode(x, add_special_tokens=True))

max_len = 0
for i in tokenized.values:
    if len(i) > max_len:
        max_len = len(i)

padded = np.array([i + [0]*(max_len - len(i)) for i in tokenized.values])
attention_mask = np.where(padded != 0, 1, 0)
```

Загрузка модели Bert

In
```
config = transformers.BertConfig.from_json_file('bert_config.json')
model = transformers.BertModel.from_pretrained('rubert_model.bin', config=config)

batch_size = 100
embeddings = []
for i in notebook.tqdm(range(padded.shape[0] // batch_size)):
    batch = torch.LongTensor(padded[batch_size*i:batch_size*(i+1)])
    attention_mask_batch = torch.LongTensor(
                                        attention_mask[batch_size*i:batch_size*(i+1)])
    with torch.no_grad():
        batch_embeddings = model(batch, attention_mask=attention_mask_batch)
    embeddings.append(batch_embeddings[0][:,0,:].numpy())

features = np.concatenate(embeddings)
target = df['target']
train_features, test_features, train_target, test_target = train_test_split(
                                                    features, target, test_size=200)

model.fit(train_features, train_target)
print(model.score(test_features, test_target))
```

Словарь

Word embeddings
способы представления текстовых данных в векторной форме