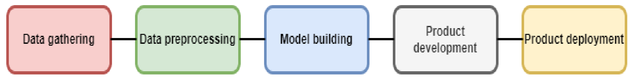
Text classifier - a road from obtaining data to product deployment

Stanislav Zámečník† and Vojtěch Šindlář†

†Department of Mathematics and Statistics, Faculty of Science, Masaryk University

1. Text classifier

Our project is a web-based app - text classifier, based on Neural Network, which enables a user to categorize his text into three separate categories, namely: sport, travel and science category. The app is built in Python programming language, with various tools, which we would like to present also, as so the whole process of obtaining data, i.e. which tools we used, till deployment model into "production".



2. Data gathering

Sometimes, when modelling, we might miss data. This problem can be due to many reasons: lack of finance, time to obtain them, no available data sets. To solve this problem, we can use the process of web scraping technique. One of the top-rated scraping tools in Python is the package Beautifulsoup.

Beautiful Soup

It is a very easy-to-use package: with few lines of code, we are able to gather text, pictures, weblinks and possibly everything we could find on the Html page. We developed a scraper to get a text for model training from the web page: `https://www.dailymail.co.uk/home/index.html`.

3. Data preprocessing

In reality, this is the longest period in product development. Since we train NN in our model, we cleaned text from URL links, removed blank spaces from text, unknown symbols, and put the whole text into lower letters, so it's easier for the model to learn. In this process, we used packages NumPy and pandas, which are popular within the data science community.

NumPy

pandas

4. Model building

Our text-classifier is built upon the trendy concept of Neural networks. Since our data are pure text, we used specifically Recurrent neural networks (RNN). The winning model architecture comprises an embedding layer, two LSTM (long-short term memory) layers, and one fully connected dense layer. We obtained 92% accuracy on the testing set with the loss 0.195.

```
def create_model():

    model = tf.keras.Sequential()

    model.add(Embedding(vocab_size,240, input_length = max_length))
    model.add(Bidirectional(keras.layers.LSTM(150, return_sequences=True)))
    model.add(Bidirectional(keras.layers.LSTM(64, return_sequences=True)))
    model.add(Bidirectional(keras.layers.LSTM(32, return_sequences=True)))
    model.add(Flatten())
    model.add(Dense(3,activation = 'softmax'))


    model.compile(loss = 'categorical_crossentropy',optimizer='adam',metrics = ['accuracy'])
    model.summary()

    return model
```

As you can see from the previous picture, the model is built in TensorFlow and Keras packages.

TensorFlow

Keras

Both of these tools are among the most popular for building NN models. They are straightforward to use, and from the newest version, they incorporate the GPU (graphical processing units) of a computer.

colab

Our model was originally developed in Google Colab - another cloud tool for collaboration. We decided for this platform because Colab let users use GPU and TPU (tensor processing units developed by Google) for performance improvement (although time per project is limited).

5. Product development I

We built our web app with the python package called Streamlit.

Streamlit

It is a tool with a fast-growing base that allows users to quickly build highly interactive web applications. Also, a strong advantage is that it is a much more "Pythonic" friendly way of building apps. Other popular python packages are Django, Flask and R tool Shiny. Primarily the first tool is widely used in the commercial sphere, and all of them are much more customizable.

6. Product development II

The whole development process was made via Git and GitHub, a provider of Internet hosting for software development and version control using Git. If you don't want to remember all the Git commands, we strongly recommend you to use one of the Git clients - it was Fork in our case.

7. Product deployment

To finally put our model into "production" so other users can explore all features of our product, we need to deploy it into some server (so far, we could only use it locally on our machine). Here comes Heroku into the game.

HEROKU

Heroku is a cloud platform that now supports many programming languages. Currently, it can host up to five web-app projects for free (there are some limitations about the size of the app, etc.). We also deployed our model to this platform, although the restriction about size prevented us from the final stage of our project.

8. Features of our web-app

Among features of our web-app we include:

- Categorizing copy of any text article.
- Possibility to add .txt file for categorization.
- It's also possible to use a combination of a copy of text and text from a .txt file.
- Use of web-scraper to obtain own data from page `https://www.dailymail.co.uk/home/index.html`.
- Possibility to train your own model.

9. Source code

Our repository with complete source code and presentation can be found here: `https://github.com/stazam/Datamining2-project`.