Web Services and Cloud-Based Systems
Report of Assignment 4b

Ruijia Lei[1,2[13823612]], Han Lin[1,2[13619985]], Zeyang Zhang[1,2[14121875]]

Vrije Universiteit[1] and Universiteit Van Amsterdam[2]

1 Kaggle Project: Natural Language Processing with Disaster Tweets[1]

1.1 Introduction and Motivation

The growth and influence of social networks[1] has not only led to massive data collection, but is also a reliable source of data to gain valuable information. Twitter[2] as one of the major social platforms is now used to collect tweets related to disasters. These tweets are identified and collected by tags, and finally through Natural Language Processing to further analyze and classify whether the disaster tweets are genuine or not.

Sometimes, delayed coverage of disasters in the news media can have serious consequences, especially in poor or inaccessible areas. The importance of this project is that it can drastically reduce the delays. Through social networks, NLP model[3] can quickly identify and analyze disaster information in order to save more lives and property in time. The main aim of our project is to predict or distinguish if a tweet contains a real disaster or not. We aim to complete the basic model construction, generate the visualization output, and then implement the process orchestration using the brane framework.

1.2 Process of implementation

The followings are some of the key steps necessary for us to perform NLP pre-processing:

(a)Text Cleaning(Removal of stop words, HTML tags, emoticons, etc.)

(b)Tokenization(word segmentation) which is a way of separating a piece of text into smaller units called tokens(words, characters, or subwords).

(c)Lemmatization is a process of converting words to their base form. Such as making "running" and "ran" to be back to "run".

(d)Generation of N-grams[4]. To process text into successive sequences of words or symbols or markers.

As for modeling, we used BERT(Bidirectional Encoder Representations from Transformers)[5] to supervise the chemistry department, training a model that predicts disaster tweets. The BERT model[6] has the following characteristics:

(1)Transformer is used, which is more efficient (parallelism) than RNN

[1] `https://www.kaggle.com/competitions/nlp-getting-started/overview`

(2)Capable of capturing longer distance dependencies
(3)Adopt a bidirectional language model for true learning context
As for visualization, we implemented the display of data distribution characteristics, word frequency and n-gram statistics. In addition, we showed top 10 locations in a world map where catastrophic statements are posted and demonstrate the accuracy of the model.

2 Brane Framework for Orchestration[2]

This section elaborates details about the wholoe project that we implemented with Brane. The URL of our project is here [3]. Building packages, making different packages being submodules of the main Github repository for easier import, unit testing against packages, composing pipeline with Brane script, DOI generation, automated builds and tests execution with Github actions. The overview of the process is demonstrated1.

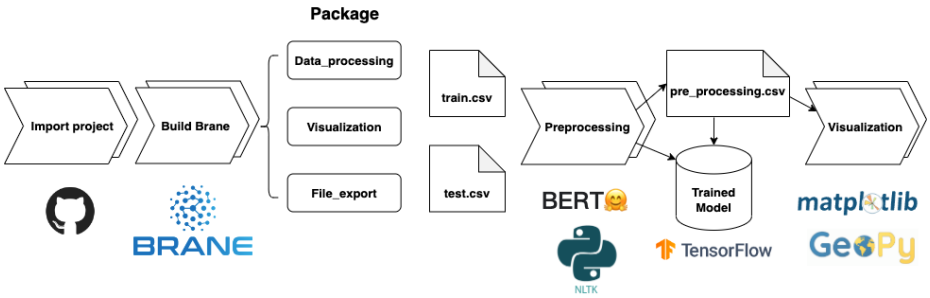

Fig. 1. The execution process based on the Brane framework

2.1 Functional Brane packages

Brane packages which contain different functions and play associated roles through the whole Kaggle challenge project have been implemented separately.
File Export We use python's os package to implement the file copy statement, which we will execute at first in the pipeline. It is able to import the local dataset into the public "Data" folder automatically. So that subsequent packages can interact with the files by giving the parameter csv_path with the value '/data'.
Data processing The data processing pipeline is based on RAHIL PARIKH's[4] solution. The preprocessing package reads the CSV files that were downloaded from the Kaggle competition and turns them into 5 pandas dataframes. It uses the BERT model and classifies tweets into catastrophic and non-catastrophic contents, so the target of its preprocessing is to remove non-text content in

[2] https://wiki.enablingpersonalizedinterventions.nl/specification
[3] https://github.com/Han-Linnn/WSCBS2022
[4] https://www.kaggle.com/code/rprkh15/nlp-eda-bert/notebook

the dataframes that affect the model training. We perform remove punctuation, emojis, URL strings unprocessed HTML code blocks, and other types of useless content by using regular expressions patterns to match them. Finally, the function saves the resulting train and test set as CSV files.

Visualization This package aims to provide an initial overview of the data distribution, and text characteristics in the step of exploratory data analysis[7]. We used bar charts to observe the frequency of occurrence of specific texts and the count of locations in which tweets were sent. More clearly, we used the Geopy library to generate a global map to show the top10 tweeting locations2, and since it is interactive, the file was exported in HTML format(also supported by Brane).

Fig. 2. Top 10 locations of Twitter posting places

2.2 Automate builds and tests by Github actions[5]

Github actions helps us Continuous Integration and deployment of the project. It splits the import tests of the Brane package and unit tests. In the test of Brane, we compile the Brane framework and use RPEL[6] to test the package. To solve this efficiency issue, we upload the locally compiled brane binary file to the Github repository, so that the virtual machine running by workflow can directly use the binary file without executing Brane compilation process, which just takes 5 minutes to finish the test. For unit testing, we used flake8 to assist in executing the unit test file successfully to run our unit test.

2.3 Pipeline and DOI generation

Pipeline is written in Brane Script, it is supposed to be executed through Jupyter Lab. Digital Object Identifier(DOI) is generated by data archiving tool Zenodo[7] which It ensures that every Github update is synchronized and that the project is trusted and secure.

[5] Github actions helps create custom software development lifecycle workflows directly in the Github repository.

[6] A read–eval–print loop (REPL) is an interactive language shell, where a program is executed line by line.

[7] A general-purpose open repository, `https://zenodo.org/`

3 Trouble Shooting

3.1 Installation

We encountered some errors with cargo or rust, we needed to install the dependencies available for the cargo command, or rustup. There is one package missed, to modify the Dockerfile.rls and add a command line for installing skopeo[8].

3.2 Building project for brane package

1. When went through brane_let debugging, when it came to using environment variables - 'Can't run nested package call Exec format error(os error 8)'
   Solution: We set a shebang line before the beginning of the file '`#!` /usr/bin/env python3' to define the location of the interpreter. Then modify the permission of our file 'chmod +x brane_visualization.py'.
2. When debugging, it came to using environment variables it shows 'Could not run nested package call Permission denied (os error 13) & couldn't find an app called 'python3\r'.
   Solution: When writing the code in Windows, it has Windows line endings (CRLF which stands for 'Carriage-Return, Line Feed', which means every newline is both the \r and \n character). However, the container runs in Linux; thus, it expects only "LF" line endings. That means that the `#!`/usr/bin/env python3 line also ended with a weird \r character as far as Linux is concerned, and thus couldn't find an app called 'python3\r'.

4 Discussion

We marvel at the design of Brane and its effectiveness. However, Brane is still in developing stage, we experienced numerous problems that hindered our development process. From an implementor's perspective, the possibility to use JupyterLab notebooks and an interactive REPL[9] is very comfortable and promising, which can integrate functions in different Brane packages for resource sharing. For now, almost all systems are able to compile the Brane framework normally and use its basic test command and REPL tools, however, some systems like the Mac M1 system still cannot compile and build a local Brane instance, and every build will show the lack of support for the m1 chip configuration, thus restricting mac m1 users to use the remote invocation function of the Brane instance. In addition, in the current stage of Brane, starting a Brane instance and using an instance will occupy a lot of system memory, and sometimes it is strange that the space can't be released although we pruned suspended docker images and volumes. In the process of repeatedly compiling the instance or Brane package, the disk space will not be freed after compilation failure, which will affect the system execution efficiency and even cause the system to crash.

[8] Skopeo performs operations on container images and image repositories.`https://github.com/containers/skopeo`

[9] REPLs provide an interactive environment to explore tools available in specific environments or programming languages

References

1. Peng S, Zhou Y, Cao L, et al. Influence analysis in social networks: A survey[J]. Journal of Network and Computer Applications, 2018, 106: 17-32.
2. Weller K, Bruns A, Burgess J, et al. Twitter and society: An introduction[J]. Twitter and society [Digital Formations, Volume 89], 2014: xxix-xxxviii.
3. Khan W, Daud A, Nasir J A, et al. A survey on the state-of-the-art machine learning models in the context of NLP[J]. Kuwait journal of Science, 2016, 43(4).
4. Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., Chanona-Hernández, L. (2014). Syntactic n-grams as machine learning features for natural language processing. Expert Systems with Applications, 41(3), 853-860.
5. Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).
6. Chanda, Ashis Kumar. "Efficacy of BERT embeddings on predicting disaster from Twitter data." arXiv preprint arXiv:2108.10698 (2021).
7. Good, I. J. (1983). The philosophy of exploratory data analysis. Philosophy of science, 50(2), 283-295.

Contribution Table

Han Lin: I configured and deployed Brane. Then I implemented the automatic test by utilizing Github actions, and deployed the Github library. In addition, I finished DOI creation and data processing package.

Ruijia Lei: I helped the pre-processing task and model selection for project, showing the visualization part of exploratory data analysis and the result. At the same time, I deployed unit test for individual package and completed visualization package.

Zeyang Zhang: First, I successfully configured the pipeline by brane script, then I assisted to implement a supervised learning model - BERT which predicts disaster tweets. Finally, I finished the file_export package.