Challenges of Trustable AI and Added-Value on Health
B. Séroussi et al. (Eds.)

doi:10.3233/SHTI220417

Implementing a Microservices Architecture for Predicting the Opinion of Twitter Users on COVID Vaccines

Guillaume GUERDOUX[a], Bissan AUDEH[c], Théophile TIFFET[b] and Cédric BOUSQUET[b,c,1]
[a] *Geegz, Paris, France*
[b] *Unit of Public health, University hospital of Saint-Etienne, France*
[c] *Sorbonne Université, INSERM, Univ Paris 13, Laboratoire d'Informatique Médicale et d'Ingénierie des Connaissances pour la e-Santé, LIMICS, 75006, Paris, France*

Abstract. A strong trend in the software industry is to merge the activities of deployment and operationalization through the DevOps approach, which in the case of artificial intelligence is called Machine Learning Operations (MLOps). We present here a microservices architecture containing the whole pipeline (frontend, backend, data predictions) hosted in Docker containers which exposes a model implemented for opinion prediction in Twitter on the COVID vaccines. This is the first description in the literature of implementing a microservice architecture using TorchServe, a library for serving Pytorch models.

Keywords. Artificial Intelligence, MLOps, COVID-19, Social Media, Vaccines

1. Introduction

The remarkable performance of deep learning and its ongoing improvements raises the question of its usability in real life in the medical context. In this paper, we evaluate the feasibility of implementing a microservices architecture for the deployment of a deep learning model to classify Twitter users' opinion about COVID-19 vaccination that was implemented in a previous work [1]. In a nutshell, a deep learning model was implemented with PyTorch and CamemBERT [2], a French variant of Bidirectional Encoder Representations from Transformers (BERT) [3].

2. Method

We implemented an architecture based on three components within Docker containers, NGINX and two microservices: a backend implemented with Django-uWSGI and a prediction application programming interface deployed with TorchServe. We customized an inference handler and started TorchServe to serve the model, listening for clients' requests, and processing these requests. Docker-compose was used to define,

[1] Corresponding Author, Dr Cedric Bousquet, SSPIM, Bâtiment CIM42, chemin de la Marandière, Hôpital Nord, 42055 Saint Etienne, France; E-mail: cedric.bousquet@chu-st-etienne.fr.

build and manage the containers providing the services. Communications between clients and the server are managed through a RESTful API without maintaining the session state, hence the use of a JSON web token each time the client requests a service requiring authentication. The architecture is shown in figure 1.

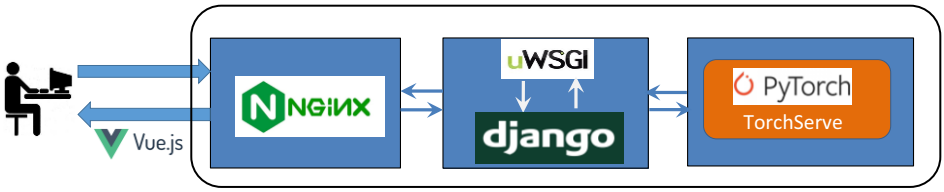

Figure 1. The microservices architecture.

3. Results and Discussion

As a proof of concept, we have implemented a simple web interface that allows the user to request the analysis of a sentence using our model. This request is then forwarded to Django via NGINX and uWSGI, then validated (format, authorization, etc.) and finally sent to our TorchServe service for prediction. Finally, the result is presented to the user in the same web interface. This interface was implemented using Vue.js, an open source JavaScript framework for building user interface and single-page applications. The F-Score of the classifier was 0.75 (precision: 0.74; recall: 0.75) [1].

This work demonstrates the feasibility of integrating a deep learning model with other applications once it is served using the proposed architecture. Furthermore, it is quite flexible, and it can be modified to meet various requirements. To our knowledge, this is the first description of a microservices architecture using TorchServe in the medical literature. This library presents two benefits: First, TorchServe keeps the deep learning model in memory and doesn't necessitate to reload it every time a new request arrives. It can also handle requests in parallels. Second, TorchServe can manage a pre-processing and a post-processing function that are defined in handlers. The main limitation of this work is that it does not take into account all the constraints related to production such as scaling, management of versions of the model, and verification of the stability of predictions over time.

This preliminary work is a first step for a research program on best practices related to the deployment of deep learning algorithms using Machine Learning Operations (MLOps), their advantages and disadvantages. It can serve as a basis for future comparisons with other types of architectures.

References

[1] Dupuy-Zini A, Audeh B, Gagneux-Brunon A, Bousquet C. Users' Reactions on Announced Vaccines against COVID-19 Before Marketing in France: Analysis of Twitter posts. medRxiv 2022.02.14.22268832; doi: https://doi.org/10.1101/2022.02.14.22268832

[2] Martin L, Muller B, Suárez PJO, Dupont Y, Romary L, de La Clergerie ÉV, et al. CamemBERT: a tasty French language model. arXiv preprint. 2019. arXiv:1911.03894. 2019.

[3] Devlin J, Chang MW, Lee K, Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint. 2018. arXiv:1810.04805. 2018.