BIO-BLOCKS: a P2P Boilerplate Python Blockchain Application for Data Handling in the Biomedical Domain

BIO-BLOCKS

University of Bologna
Blockchain and Cryptocurrencies

Serban Cristian Tudosie[1]

[1] serban.tudosie@studio.unibo.it

December 2022

Abstract

In many applications today, managing data generation and handling can be challenging when multiple entities are involved in a consortium, network, or project. In the biomedical context this is a pronounced problem. Maintaining a record of all data transactions is crucial. Additionally, data generated for machine learning purposes must adhere to certain criteria and standards. To address these issues, this work proposes using a blockchain to provide a decentralized shared ledger and a deep learning model pretrained on a binary task to filter and verify data before it is added to a new block in the blockchain.

1 INTRODUCTION

This work presents an alternative and novel way of data handling in the biomedical field. In this section, the first part is dedicated to the exposition of the problem, and the second to the implemented solution. The code is available at: github.com/CrisSherban/bio-blocks.

1.1 Data Issues

A substantial drawback in data generation in the medical field is the difficulty of having data from multiple sources and from different days of acquisition that are coherent across all the acquisition. For example, one known problem in the previous context is given by batch effects, which pose a relevant issue.

1.1.1 Batch effects

Batch effects can occur in a variety of molecular biology experiments, including gene expression studies, proteomics, and metabolomics. They can be caused by a variety of factors, including differences in the batch of reagents or samples used, differences in the equipment or protocols used to process samples, and variations in the environmental conditions in which the experiments are conducted.

Batch effects can be difficult to identify and control, as they can be subtle and may not be apparent until after the data has been collected. However, it is important to try to identify and control for batch effects, as they can significantly impact the validity and reproducibility of an experiment.

1.1.2 Some classical solutions

There are several approaches that can be used to identify and control for batch effects in molecular biology experiments. One approach is to include a balanced design, in which samples from each group of interest are evenly distributed across batches. This can help to reduce the impact of batch effects on the results of the experiment. Another approach is to use normalization techniques, such as quantile normalization or median polish, to adjust for differences in the distribution of data between batches. These techniques can help to reduce the impact of batch effects on the results of the experiment.

1.2 Proposed Solution

This work proposes an alternative based on deep learning that can be summarized as follows:

1. A group of experts decides on the research question or problem they want to address (e.g., "How can we improve the coherence and validity of data generated by multiple sources in the biomedical field?") and creates a labeled dataset with binary labels to be used for training a deep learning model. The labels could indicate whether a particular data point meets certain criteria (e.g., free from batch effects) or not.
2. A deep learning model is trained on the curated dataset.
3. The model is used within the blockchain to validate a new transaction.

The approach can be visualized in Figure 1, which is a schematic view of the system and does not include all components.

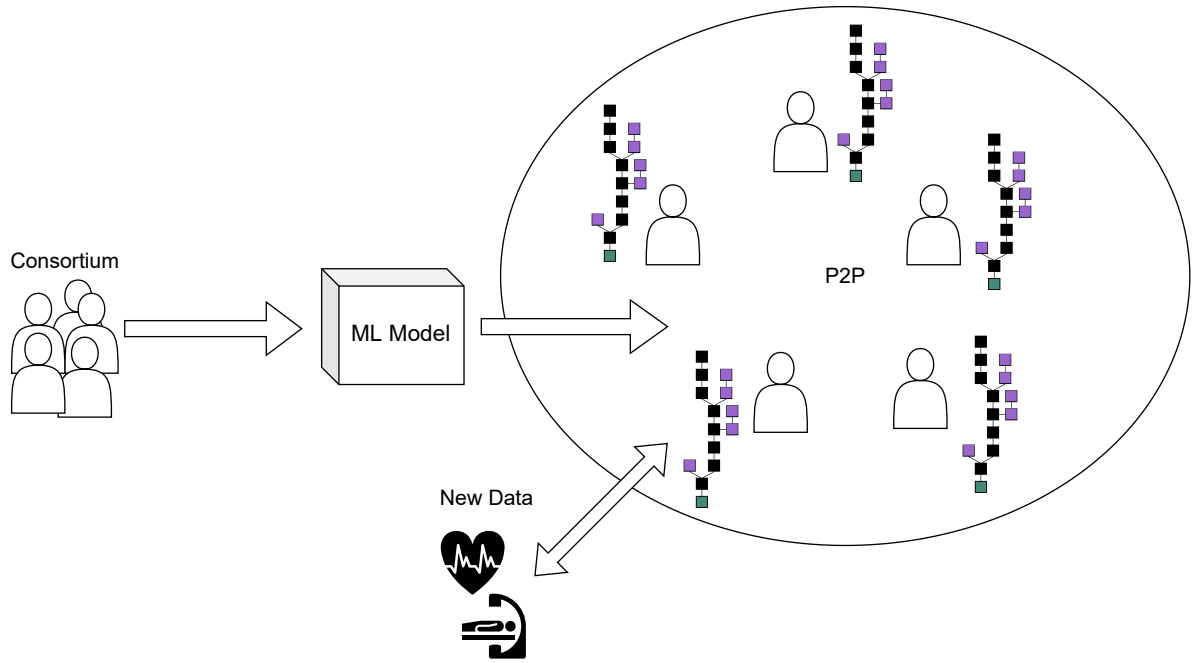

Figure 1: Graphical summary of the overall approach. On the left, a consortium initializes a wanted machine learning/deep learning model that is responsible to filter unwanted entries that do not respect certain criteria. The consortium then becomes part of the P2P network and each node contains a copy of the blockchain. On the bottom, a node makes a new acquisition, which is reported to the blockchain in the form of a transaction. When the transaction is validated the model is called to check for the data quality and a new block is added to the system. The blockchain is represented by the series of blocks for each node.

2 METHODS

2.1 Blockchain

A blockchain [1] is a decentralized, distributed database that maintains a continuously growing list of records called blocks.In general, each block contains a timestamp and a link to the previous block, which allows blockchains to be used as a ledger that cannot be altered retroactively. This work makes use of a simple blockchain implemented in python. To summarize the used implementation, each block is made of: an id, the hash of the previous block in the blockchain, the timestamp of the block creation, nonce, and a list of transactions. We treat each data entry as a transaction that for simplicity is comprised of: a the sender's id, the data itself and

a given name of the data. An UML summary of the most important classes is given in Figure 2.

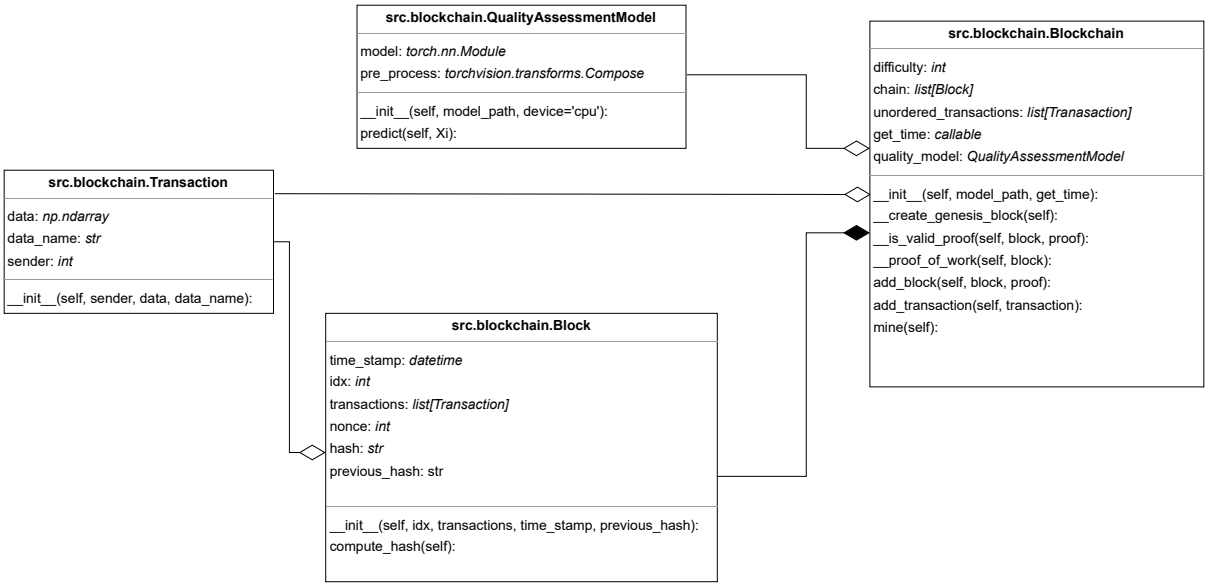

Figure 2: The UML class diagram of the blockchain.py file. The four classes are represented by blocks and the relationships of aggregation and compositions are represented by arrows with standard white rhombus and black rhombus respectively.

2.1.1 Advantages and Disadvantages

The blockchain provides multiple advantages for this setting. One is the availability of a shared ledger, where anyone can add a block. A fingerprint is then left in the data by the end-point that generated the acquisition. If the goal in the consortium is to have as much data as possible and everyone wants to contribute equally, by construction the system pushes the consortium towards a faster generation of data. Introducing a reward strategy in the system, each node could be rewarded in the form of credits for each addition which further increases the necessity of producing new data by all the nodes.

By default, there is a disadvantage. This type of system requires additions to it in order to easily scale up. Specific protocols should be used e.g. to update the blockchain from one node to another. More on the matter in the future works section.

2.2 Dataset

This work relies on a dataset to show a specific use case, the one of x-ray images. In particular, the dataset is made of images of x-ray acquisitions[1] from people with SARS-CoV-2 and from people without the virus infection. The labels can be retrieved from the metadata.
However, in general, a specific and curated dataset can be used by experts to filter out unwanted effects in the upcoming acquisitions. In this specific case to test the overall system we decide that there is the interest in having only x-ray images of people infected by SARS-CoV-2. Although, this choice does not address directly batch effects, which would require manual labeling, it shows how the overall system can be implemented with the binary classification task.

[1] https://www.kaggle.com/datasets/bachrr/covid-chest-xray?resource=download

2.3 Model

To tackle this binary classification task this work employs a small Resnet [2]: a resnet18 which is further adjusted for the task at hand.

2.3.1 Pre-processing

The pre-processing is an important step. The choice of the methods are drawn from classical computer vision approaches. In the train pre-processing, data augmentation tecniques are used to balance the lack of data entries in the current dataset. The augmentation techniques are: random flips and random rotations. Sequentially, the transormations used are: resizing, histogram equalization and smoothing.

2.3.2 Architecture

The model is adjusted to accepting one channel images as input by the usage of a new convolutional layer instead of the classical 3 channel one. Additionally, a new classification head is added to support the binary task.

2.3.3 Training

This work employs a pre-trained version of the resnet18 which was trained on imagenet given the importance of weight initialization. Then, a transfer learning step is employed to adapt the newly added classification head to the task. Subsequently and finally, models' parameters are unfreezed and the whole model is finetuned on the task. We do not wish to polish the model, thus the training stops when 80% of accuracy is reached on the validation set.

2.4 P2P and Web Interface

To address the decentralized nature of the system this work threats the users as nodes. In addition this work provides a web interface to easily add new blocks and visualize the current blockchain. With the current implementation the responsability of updating the blockchain falls under each node which provides a send() method.

3 RESULTS & CONCLUSIONS

To emulate the working scenario of the system several nodes are instantiated on multiple ports and are tested with respect to the various proposed methods such as: adding transactions, updating nodes on the network, verifying the trained model when adding new acquisitions.

3.1 A working example

Focusing on the web interface, we can show a working example of loading and updating the blockchain. Figure 3 and Figure 5 show how the web app provides the two main pages to interact with. The choice of the nodes ports can be selected in the CLI as well as uptading other nodes.

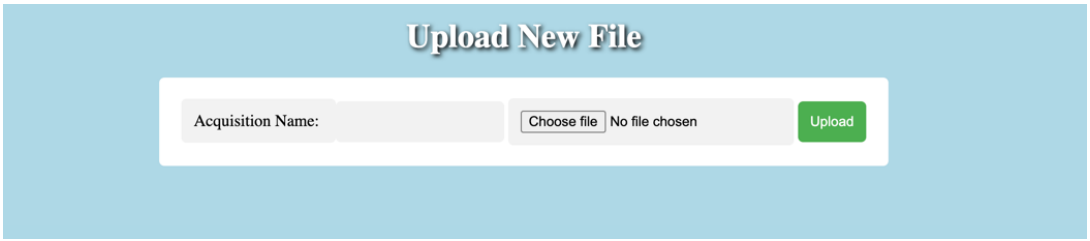

Figure 3: Upload web interface. The left box allows to enter the name field for the new acquisition. The right form instead allows the uploading of an image.

Figure 4: An example image that can be passed through the web app. In this case an x-ray chest image of a person with covid.

Figure 5: A visualization page for the blockchain that shows the genesis block along with an added block. The second block has a single transaction in this case.

3.2 Future works

In order to efficiently deploy such system, the suggestion is to include in future works a revised implementation based on technologies such as Kademlia [3] to allow scaling up the overall

system. A popular python implementation for Kademlia can be found on github by bmuller[2]. Another addition is the usage of non-fungible tokens (NFT)s[3] to allow the authenticity and ownership to the uploaded data.

3.3 Conclusions

This work expands the alternative to data incongruity in biomedical applications that are destined for machine learning purposes. It provides the user with a web interface and nodes that can interact with each other. Transactions are stored within a blockchain and are further verified by a deep learning model. The work shows that the approach might be useful in the exposed setting, implementable in the real world scenario and easily extendable to be scaled up.

[2] https://github.com/bmuller/kademlia
[3] https://www.merriam-webster.com/dictionary/NFT

4 REFERENCES

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Cryptography Mailing list at https://metzdowd.com*, 03 2009.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[3] P. Maymounkov and D. Mazières, *Kademlia: A Peer-to-Peer Information System Based on the XOR Metric*, vol. 2429, p. 53–65. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002.