Predict Chain

Matthew Pisano, William Hawkins

Spring 2023

1 Introduction

Throughout our project, we have built upon, or have related to, the ideas of many other projects and papers. Some of these relations were explicit, such as our work with, and utilization of, various types of neural network. Others were implicit, like the relation that our project has to the many other implementations of blockchain-based predictive projects. It is important to acknowledge the influences that other works have had on our project, and it is equally important to analyze the similarities between our project and the other papers that we have referenced here. Through this, we hope to better understand the perspective of others on similar topics to our project and hopefully improve upon our work and techniques in the future.

2 Related Work

2.1 Sharing Updatable Models

Decentralized & Collaborative AI on Blockchain [2] is a great example of a project that combines both artificial intelligence and blockchain technologies in a similar manner to our own. In this paper, Microsoft outlines a blockchain-based predictive marketplace that utilizes AI for predictive analysis. In this system, there exists an initial model that is trained on the data collaboratively provided by multiple users. To encourage the submission of high-quality data, they propose an incentive mechanism that rewards honest users, and punishes dishonest ones. This model is trained off-chain on the client's machine. All the logic involving the data submission and incentives are done within an Ethereum smart contract.

The Model This system leaves the exact definition of the model flexible between different clients running the system. They suggest that the specific class of model would be some type of neural network, giving examples of CRF-RNNs and Support Vector Machines. Notable, both of these models are geared towards regression and series analysis as the focus of this system is on predictive analysis. Depending on the model that is used, the authors mention that the corresponding mechanism would change from model to model.

Incentives Similarly to the models, the authors leave the incentive mechanism of the project also without an exact definition. This is to give end users flexibility in their individual implementations. They do give several examples and specifications of this mechanism, however. These examples fall into two categories: gamification and monetary incentives. In the gamification strategy, they suggest that there should be no monetary incentive for contributing data. This would create a system that encourages a collaborative effort in building a model for the common good. In this strategy, instead of money, they would have tokens or badges that are awarded to users for good data. The contract would evaluate contributed data as 'good' if it increases the accuracy of the model on a testing set. For the monetary route, they suggest that users who choose to upload data may have to also submit some stake into the contract. Using this stake as collateral, the contract uses a similar evaluation of 'goodness' to distribute awards or punishments.

Relation to Our Work This paper shares a very large similarity with our project. However, while the core ideas are quite similar, there are some notable differences. Both of the projects involve creating a model, open to public query, and trained on user-provided data. Both of these projects have the goal of making machine learning predictions more accessible to more people. In comparison, our project is more centralized and with a larger focus on incentives. For example, in our project, models are trained on a special, central node instead of on a node managed by a client. Each model in our project has only one user-uploaded dataset as its source instead of each model being a collaborative effort from many users. In PredictChain, there is a greater focus on costs and incentives than in the sharing models project. Specifically, our project is built around encouraging usage through the possibility of monetary reward, but in the author's system, the usage is achieved by either gamification or a monetary incentive. Additionally, queries are not free in PredictChain as they are in Decentralized & Collaborative AI on Blockchain. In both systems, individual users make contributions in the form of data, one initial user controls the model hyperparameters and which dataset the model will be trained on, and all users can query the publicly available models.

2.2 The Price You Pay For Trust

Blockchain Enabled AI Marketplace: The Price You Pay For Trust [5] is another example of a marketplace that uses both AI and blockchain technologies effectively. One of the main focuses of this project is on privacy. It explicitly gives examples of health or insurance entities needing models trained on confidential data. The principles that this project are *trust*, *fairness*, and *auditability*.

Datasets The datasets in this system are unique since the bulk of the data is not tied directly to any one training request. In this project, a user uploads only a small chunk of data. This data will then be used as the validation set for the training. The majority of the training data for this query is pulled from private sources. This serves the purpose of only requiring users to upload a small portion of their data. However, this has the disadvantage of requiring external data to train models. To acomodate this, the authors implement several layers of security to protect private data sources.

Models The models that this paper targets are much less integrated than those of other papers. Also unlike other implementations, these models are trained in a decentralized nature. This training strategy is reflective of federated learning. No one model is given all the data in bulk, it is only given sections with the confidential information stripped out. To add to their security, they also use hashing algorithms to further secure their data. Additionally, the models themselves are eventually shared with the requesting user along with the model results.

Blockchain Communication Like other project, this implementation uses a significant off-chain component. This component communicates with the private, permissioned blockchain that this system uses. This is to allow for more complexities in the training of the models ad on-chain training has significant speed and memory constraints. Additionally, the authors utilize Hyperledger Fabric to aid in the distribution of the data between the nodes and the aggregator. By using this bridge, data can be selectively distributed to only the nodes that need the data for training will have access to it.

Relation to Our Work This paper also has a notable similarity to PredictChain. Both of these projects utilize blockchain as an important means of communication and record-keeping in an AI marketplace environment. Similar to the updatable models project, this project implements this idea in a more decentralized manner. Instead of only keeping models on the client nodes, like PredictChain, this project allows users to query the content of the models directly, provided that the users is the one who asked for the training or if the user is a training node. Also unlike our project, the training of these models is a collaborative effort. Instead of one, monolithic node, this project used a federated style of learning. Finally, the authors have implemented a data system that is more decentralized than monolithic. Instead of users uploading data directly, it is pulled form a variety of sources, including the original user. In this system, the private data owners must submit bids to the aggregator for the chance that their data gets included in a trained model.

2.3 Blockchain Prediction Markets for Renewable Energy

Towards the Use of Blockchain Prediction Markets on Forecasting Wind Power [6] is a paper that lays the groundwork for a use-case of blockchain empowered prediction markets. Like blockchain technology, renewable energy has ballooned in use recently. Many people are drawn to both of these innovations for some of the same reasons: A promise of a new way of doing things that breaks from the old system. Like renewables replacing fossil fuels many see blockchain replacing centralization as a necessary step to a better, freer, more sustainable future.

Motives A key problem with many forms of renewable energy sources are their unpredictable supply. Wind turbines need wind, solar panels need sunshine. Being able to better forecast the output of these energy production methods at a various time or place can help alleviate the pitfalls that prevent them from growing in market share. At the time of publication the authors, Shamsi and Cuffe, note that the use of blockchain based prediction markets in renewable energy forecasting had not yet been explored.

The Oracle Problem Shamsi and Cuffe discuss various prior proposals that seek to take on this issue. One crowdsourcing proposal, they say, lacks proper incentives. While another prediction market with decent incentives is centralized. The authors discuss the *Augur* Platform, an Ethereum based prediction market. The Augur system seeks to deal with a key issue with decentralized prediction markets: The oracle problem. This is the problem of getting new outside data into the platform. Augur gives a basis for "Independent participants [to] report, dispute and validate the data through a decentralized set of smart contracts". For their paper, the authors have based their framework off of Augur's. They hope to expand upon its proven reliability with their own improvements.

Relation To Our Work Both PredictChain and this paper enable user to gain access to better forecasting of events. In PredictChain users are able to query pre-existing models as well as contribute their own data to train these models. This collaboration improves performance and accessibility to predictive capabilities. In the predictive marketplace for renewable energy, users can observe the spread prices on trades and calculate what the market clearing probabilities are for any such event. The more contracts that are sold and bought on the marketplace the closer the spread price will be to the true event probability or value. In other ways, this paper differs from our project. In PredictChain, the oracle gathers new data to determine the accuracy of model queries. More specifically, in stock market prediction for example, a user may specify a model to query. That user can also provide the data to call the model on e.g. stock data for $AAPL from 4/17/2023-4/19/2023 if we assume this recurrent model is using a three-day window. The oracle would then gather the true stock market price for the target date, 4/20/2023, and record the model's accuracy.

2.4 Table Comparison

As a summary, this chart illustrates the commonalities and differences between our project and these related works:

Table 1: Comparison of Each Paper

Paper	**PredictChain's Specification**	**Paper's Specification**	**Commonalities**
Harris et al. 2019 [2]	• Model run and trained on special node • Datasests from one user only • Always has incentives • Paid queries	• Model run and trained on client node • Datasets collaborative from many users • Has optional gamification • Free queries	• Community dataset contributions • User-specified models and hyperparams • All users can query public models
Sarpatwar et al. 2019 [5]	• Model run and trained on special node • Models kept only on one node • Datasests from one user only	• Models run and trained on multiple nodes in a federated manner • Models can be shared between nodes for training • Datasets are submitted by users with additional data submitted by private soucres	• Users can submit custom datasets • Data from different users can be used for a common purpose
Shamsi and Cuffe 2020 [6]	• Central Node reports new data to the oracle. • Predictions from models hosted on central node.	• Users report new data to the oracle for contract-settling. • Users manually submit their predictions via buying contracts.	• Both systems allow greater access to predictive capabilities through pooling data and predictions from different parties.

3 Background Knowledge

3.1 Long Short-Term Memory Networks

Long Short-Term Memory Networks[4] is one of the most influential, and useful, types of network that we work with in our project. These networks introduce several important improvements on classical backpropagation-through-time recurrent neural networks. One of the most notable contributions of LSTMs is their resolution of the 'exploding and vanishing gradients' problem. In RNNs, it is common, and often disruptive, to observe error signals either grow to extreme values or disappear entirely when updating the weights of the model. This either leads to extreme unpredictability in the next value that weights will take on, or it leads to the weights not being updated at all. In both cases, the effectiveness of the model is greatly impacted. To help remedy this issue, the authors propose an architecture that enforces a 'constant . . . error flow through internal states' inside the model.

Before this paper, it was well known that traditional RNNs performed very poorly when given long time lagged datasets. In this data, the time window that the input data is set in is several time steps ahead of the window that contains the target prediction. This was so pronounced that, in another paper[3], the authors showed that other contemporary RNN algorithms can be outperformed by random guessing. To show that their new architecture is superior in areas that RNNs fall short in, the authors devise a series of six experiments. Through these experiments, they show that LSTMs can outperform RNNs on long, minimal time lag data. Through their papers, the authors show that their new LSTMs can perform well in both short and long time lagged situations.

Relation to Our Work The work of Hochreiter and Schmidhuber has been incredibly useful in our development of PredictChain. As part of our goal, our project is designed to accept a wide variety of datasets and is also designed to train models on a wide range of parameters. Part of this promise of variety is our models' abilities to work with both short and long time lagged datasets. While some of our models fall short in some areas, such as RNNs in a long lagged context, others are able to excell in those areas, such as our LSTM and GRU type models. Through our incorporation of the authors' work, we are able to provide a large range of models for a wide variety of use cases. This helps the project to achieve its goal of greater accessibility in the field of predictive modeling.

3.2 Recurrent Sequence Modeling

One of the core aspects of our project is our usage of various types of neural network for predictive modeling. Most of our neural networks have some recurrent capability, specifically our RNN, LSTM, and GRU model types. Additionally, the data that we suggest that these models be trained on is sequential data, exactly what these types of model were designed for. One paper that aligns especially closely with this component of our project is *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*[1].

Within this paper, the authors aim to evaluate the relative performance of RNNs, LSTM networks, and GRU networks. Similar to our use case, they perform this evaluation using sequential signal data, similar in nature to the stock market dataset that we use in our example. Primarily, they focus on the LSTM and GRU variants of recurrent neural networks, rather than vanilla RNNs themselves. As part of their evaluation, they compare the performance of LSTM unit, GRU unit, and tanh unit RNNs on modeling sequences of polyphonic music and speech signals.

As part of their evaluation, they utilize three publicly available music datasets along with two internal speech datasets from Ubisoft. The design of their experiment is to compare the negative log probabilities of the models after they are given 20 consecutive samples of audio data and are then tasked to predict the next 10 consecutive samples. In the results that they gathered, they concluded that, overall, the GRU unit outperformed the other two evaluated models. These results are amplified in the Ubisoft datasets, where GRU unis significantly outperform the others in one of the test cases and have similar performance to LSTMs in the other. Additionally, in the Ubisoft Dataset B, the negative log-likelihood of the GRU model continues to shrink in a super-linear manner, whereas the other models become asymptotic.

Relation to Our Work This paper both confirmed some of the initial assumptions that we had about the models and changed some others. In the design of the project, we had initially marked GRU models to be better than all the other models. We denoted this perceived difference in output quality through the constant multiplier we applied to each of their *model_complexity* values. After our analysis of this paper, we ran several tests to see if this paper's results also applied to our project. In our tests, we noticed something very similar to the paper. GRUs and LSTMs has more similar performance than we initially thought. Meanwhile, vanilla RNNs lagged behind. Based off of this evidence, we decided to update these values to better reflect the experimental performance of these models.

Table 2: *model_complexity* Values of Our Models

Model	**Initial Value**	**Final value**
MLP	1	1
RNN	1.5	1.7
LSTM	2	2.2
GRU	2.5	2.4

In our changes, we amplified the values related to all RNN-based networks. We also brought the complexities of the LSTM and GRU networks closer together. This change was made to reflect their relative performances to each other and to more traditional RNNs. Overall, this paper has been important for our project as it has helped us to reflect upon our models and how to best evaluate them.

3.3 Stock Predictions

This next paper is closely aligned with the example dataset that we are using for PredictChain. They focus on using big data to train predictive machine learning models to forecast the short-term future behavior of stocks within the Chinese stock market [7]. Within their paper, they utilize several techniques designed to cut down on the noise that is inherent to the stock market to allow for better predictions. In their paper, they focus mainly on three methods that they use to accomplish this goal:

- Creating a new dataset, gathered primarily from the *Tushare* API
- Performing feature engineering on their dataset to ensure that it can be better used by their model
- Customizing a Long Short-Term Memory model for the actual predictions

For their dataset, they had gathered the past performance of 3558 stocks from the Chinese stock market over a period of two years. Each entry in their dataset contains many attributes including the daily price of a given stock and the ID of that stock. In order to gather this data, they utilized the *Tushare* API, along with web-scraping from *Sina Finance* and the *SWS Research* website.

As part of their feature engineering, they utilized three primary techniques. First, they applied feature extension to their dataset. Through the use of this technique, they added additional meta-attributes to the entries, such as polarizing or calculating the fluctuation percentage. Adding this meta-data to the dataset helps to give the model a more complex picture of the stock and how it performs over time. Next, they eliminated some features by using the Recursive Feature Elimination (RFE) algorithm. This algorithm evaluated each feature based on the degree to which that feature influenced the stock's performance. By eliminating unnecessary features, they allow their model to pay more attention to the most influential features, improving its predictive capabilities.

Finally, they developed their models. They used a two layered LSTM model, with the model only having an input layer and an output layer. As for their output, they kept the output as simple as possible to make sure the model was focussed only on predicting the movement of the stock. The model either outputted a one for the stock going up at a given time step, or a zero if the model believed the stock would go down. Their hope is that this simplicity will also simplify the complexity of the information that investors will have to interact with.

Relation to Our Work As mentioned previously, the general purpose of this work mirrors part of the functionality of our project. The main difference here, being that it is much more focused on the predictive aspect. Along with their LSTM model, they test several other models on their dataset as a comparison. Out of these models, the LSTM performed the best, with the multi-layered perceptron and the other, non-neural network, algorithms falling behind. Their observed performance matches our initial assumption and the modifications we had made after reviewing [1].

One notable feature that this paper has that our project implicitly lacks is their complex preprocessing of the dataset. At present, PredictChain leaves the majority of dataset preprocessing up to the uploading user. That is, our project does not perform any feature engineering before submitting the dataset to models. A future improvement that we could make to our project is to give users the option of having feature extension or elimination automatically applied to their dataset after submission. This change may result in yet better performance of our models after training on user-submitted datasets.

3.4 Table Comparison

As a summary, this chart illustrates the impact that each of the papers had on our work and our knowledge:

Table 3: Impact of Each Paper

Paper	Notable Impact
Hochreiter et al. 1997[4]	Encouraged us to expand our support for forcing time lag in datasets and expanding the degree to which users could customize the lookback windows
Chung et al. 2014[1]	Helping us to better understand the relative performances ofRNNs, LSTMs, and GRU networks
Shen et al. 2020[7]	Inspiring possible future improvements to our dataset preprocessing and model training

4 Closing Thoughts

After researching and reading these relevant papers, we were able to gain valuable insight into projects similar to our own, implement relevant features, and take note of some features that could be added in future revisions of PredictChain. The knowledge that we have gained from these papers falls broadly into two categories: improvements that we could make to our usage of artificial intelligence and improvements that we could make to our usage of blockchain.

4.1 AI Papers

The papers that had focused on the AI component helped us to re-revaluate how we rate our models and how we can best improve the data processing and training aspects of the project. [4] and [1] helped us to better understand how we should rank the potential effectiveness of our models through our usage of their *model_complexity* values. Specifically, [4] encouraged us to expand the degree to which users could customize the time lag that is present in the dataset upon model training. [7] helped us to better understand some of the future improvements that we could make to both our dataset preprocessing and model training.

4.2 Blockchain Papers

The papers that had focused on blockchain instead

References

[1] Chung, Gulcehre, Cho, and Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *Neural and Evolutionary Computing*, 2014.

[2] Harris and Waggoner. Decentralized & collaborative ai on blockchain. *IEEE*, 2019.

[3] Hochreiter and Schmidhuber. Bridging long time lags by weight guessing and ”long short term memory”. 1996.

[4] Hochreiter and Schmidhuber. Long short-term memory. *Neural Computation*, 1997.

[5] Sarpatwar, Ganapavarapu, Shanmugam, Rahman, and Vaculin. Blockchain enabled ai marketplace: The price you pay for trust. 2019.

[6] Shamsi and Cuffe. Towards the use of blockchain prediction markets for forecasting wind power. *IEEE*, 2020.

[7] Shen and Shafiq. Short-term stock market price trend prediction using a comprehensive deep learning system. *Journal of Big Data*, 2020.