Solutions to Gig Economy using AI and Blockchain

Harihara Vinayakaram Natarajan[1], Kunal Sunil Kasodekar[2], Annant Vijay Kushwaha[3], Keerthana Jayakumaran[4], Sahana Krishna Karki[5], and Prachi Gupta[6]

[1,2,3,4,5,6]*Wipro Limited, Bangalore*

1 harihara.natarajan@wipro.com ; 2 kunal.kasodekar@wipro.com ; 3 annant.kushwaha@wipro.com ; 4 keerthana.jayakumaran@wipro.com; 5 sahana.karki1@wipro.com ; 6 prachi.gupta4@wipro.com

Abstract

In oil and gas enterprises, oil drilling produces a huge volume of toxic wastewater as a by-product. The process of disposal of this toxic water lacks transparency and suffers from a multitude of problems like environmental damages, longer payment cycles (7-8 weeks), fraud, etc. There is a considerable human cost on the drivers as they are hired for uncertain schedules and working hours. This paper explores the solutions to the above problems using AI and Blockchain. It pro- poses a forecasting model that will predict the time when the wastewater tanks are filled. This will enable the truck drivers to actively participate in Gig Economy and plan their day accordingly. The blockchain-based model will allow the enterprises to verify the payment parameters and enable seamless auditing to detect frauds. Given the ubiquitous distributed nature of sensors and the likelihood of these sensors/networks malfunctioning, the solution includes additional validation from the truck driver using his smartphone. This solution also utilizes the blockchain to provide a reputation and uses this as an input to the AI algorithms. To solve this problem at a large scale we explore the use of various clustering mechanisms along with a continuous learning model.

Keywords: Gig economy, Blockchain, Predictive Analysis, Time Series Forecasting, Reputation Mechanism

1. Introduction

We had completed a POC using Blockchain for solving the Waste Water Management problems [1]. We had automated the payment using blockchain and smart contracts, all the events in the lifecycle of a trip are stored in the blockchain. Analyzing the results and shortcomings of the POC we realized the need for considering the system holistically. We treated the system as a "Digital Transformation Initiative". We documented the following:

(a) The need for a prediction system that would predict the time when tanks would be full and the Scheduler service to schedule the trips based on prediction. We realized an efficient prediction system coupled with the scheduler and a mobile app could empower the young drivers and fuel the dreams in the true Gig economy model [2] allowing them to plan their schedule in advance as per their preferred time and

location. This would also enable the fleet owners to plan their fleet utilization optimally and reduce the unnecessary time slots spent by trucks in oil fields reducing chances for pilferage.

(b) The power of the "nudge economy" to create positive behaviors. The blockchain is an ideal tool to implement the "reputation mechanism" to power this. We also realized that the blockchain can be used as a marketplace to power better algorithms by implementing the same "behavioral economics" for the AI vendors. The auditing feature of the blockchain could be used to validate the claims of the competing AI vendors.

(c) The sensors in the water tanks can be resilient by using the mobile phone of the truck driver coupled with AI-based image recognition and the "nudge economy".

(d) The need for a public chain that provides privacy using mechanisms like "verifiable data structures" along with the standard auditable features.

As they say, you can eat an elephant "bit-by-bit". This paper focuses on various approaches for the prediction of the time when the tank will get full so that truck drivers can work as per their preferred time and location, hence promoting the GIG economy and how the prediction models can be applied at a large scale. It also addresses the Reputation model to trust the reading by the truck drivers as IOTs devices are prone to give the wrong reading. We hope to address the other problems in a different paper.

2. Solution Architecture and Models

In this section, we will explain our solution architecture. We will separately explain our blockchain architecture, data collection based on reputation, Machine Learning models, and methodology used.

2.1. Blockchain

We have performed an event storming session with field experts and stakeholders to understand the problem in a better way. The session helped us in nailing the requirements of the services such as numbers of services needed, defining the functions/ API for the services, type of blockchain and AI model that can be used, etc. We have decided to have a microservice-based architecture which will have the benefit of scaling, accelerate development, reduce the service load, facilitate testing, etc. To communicate between the various services we are using Redis.

We decided to build our solution on ethereum. We are making an entry of every event in the blockchain for a complete life cycle of the trip i.e from the event of wastewater disposal request raised to wastewater dumped at the disposal site by the truck driver. Since we are storing all the events of a trip on the blockchain, the invoice generation will be instantaneous as there is no need for any manual input and verification to prove

the occurrence of the event. After the final event, the invoice will automatically be generated as part of contract logic. This will solve our problem of delayed payment and help us to find the culprit for the inappropriate disposal of wastewater. We have used ethereum as our blockchain platform because-

1. Ethereum blockchain provides a platform for private network setup.

2. Ethereum allows the use of smart contracts to model any complex business logic.

We have deployed a network using ganache where the oil field operator, the fleet owner, and the services represent the nodes of the network. We also have a 12 node setup on the cloud platform using ethereum private network. The different business logic is coded by means of a smart contract using solidity language. We have also build ERC-20 tokens on top of ethereum and called it as "AQUA COINS". Every onboarded user like the truck driver is assigned with AQUA COINS. These coins are also used for rewarding the truck drivers according to their reputation which will increase by providing the correct meter reading.

2.2. Datacollection through IoT devices

For our time-series forecasting model, data originating from the IoT device/meter needs to be correct. However, IoT devices are sparsely connected as the oil fields are spread across a large area. Hence it is very difficult to get correct data every time.

So, we decided that a mobile application would be used by the drivers so that we can verify the correctness of readings that we are getting from the IoT device thus augmenting humans in the system. We decided that the driver should click a picture of the meter reading using his mobile camera. The application automatically captured the date/time and the location of the picture. We used the Amazon Rekognition service to extract text from an image uploaded by the driver. Amazon Rekognition [3] is highly scalable and uses deep learning technology to analyze the image. It provides a simple API that can analyze images and extract text, human faces, known shapes, etc. Interestingly the accuracy of detection significantly increases when there are additional objects in the frame. Thus, we mandated the use of a selfie (human face). The user will enter the reading manually which will be used as a mechanism to assign a reputation to the user according to the correctness of the reading provided by him. If the reputation of the user is high and there is a mismatch in the reading, we will consider the user input to train our AI Model as IoT devices sometimes give wrong readings. We have a reputation model that will assign some reputation to the truck drivers according to the meter reading entered by him. This will be matched with the IoT device and image rekognition reading. Using these readings our AI models will be trained.

2.3. AI

For forecasting when the storage tank will get full, we required genuine time-series data, but it was difficult to get this data from oil and gas enterprises due to an NDA. We under- stood the various features that affect the flow rate of toxic water generated like pressure in the oilfield,

chemical concentration, drill depth, etc, which in turn further affected the time for filling the wastewater tanks. Based on our understanding of these various features that affect the filling time we accordingly created our dataset. Various machine learning algorithms were used so that it eventually learns based on the data provided to predict future events and uses a feedback loop to handle/correct the prediction of wrong events in case of flow rate or physical parameters changes.

2.3.1. Forecasting

We have considered time series forecasting technique to predict future events. The models we have used for forecasting are as follows:

1. Arima [4]
2. FB Prophet [5]
3. LSTM [6, 7]
4. Rule-based Machine Learning using COREL [8]

2.3.2 Clustering

The dataset we are using has data for around 35,000 oilfields with 3 to 5 storage tanks (wastewater tanks) each which sums up to approximately 1.4 lakhs storage tanks. Using one model to predict events of all 1.4 lakh storage tanks is infeasible as each of them belongs to different geographical areas and have different features like initial parameters which result in a varied flow rate. So we will have a large number of input and target variables to train the model. Also having 1.4 lakh models to train for each storage tank is also cumbersome. So we need a scalable solution to train the models. The solution is Clustering (Unsupervised learning) of the wastewater tanks.

2.3.2.1 DBSCAN

We make use of DBSCAN [9] an algorithm that will cluster data points based on their density and will minimize geodetic distance as compared to k-means which though being a popular algorithm will only minimize variance, not the great circle distance creating improper clusters. Therefore Dbscan works better for spatial latitude-longitude data. It is not affected by outliers as it treats them as noise and it automatically decides the number of clusters required depending on epsilon, minimum points and the number of core points.

- We can cluster the storage tanks that are in the same geographical area as they will have approximately the same features like pressure, chemical concentration, drill depth, etc.
- After creating clusters based on the proximity, in each cluster, we will create sub-clusters based on the capacity of the tanks so that all operations get

synchronized. This will reduce the number of models to be trained.

2.3.3 Prediction Mechanism

Storage tanks are clustered together and a cluster centroid is assigned to it. The Prediction will be done for a cluster by training the models of a sub-cluster rather than training models for each storage tank in the sub-cluster. Each oilfield has storage tanks of 3 capacities 15000,21000 and 35000 so we create 3 sub-clusters for every cluster. Every cluster can have up to 3 models irrespective of the number of storage tanks it has based on the capacity of the tanks each cluster has. This reduces the computation and storage overhead.

2.3.4 Feedback loop

A machine learning model is trained on some data and its hyperparameters are tuned to create a highly accurate model. When this model is put into prediction its accuracy will decrease exponentially with time as the parameters considered for training will dynamically change. Also, it may suffer from concept drift: the model may try to capture a relationship between the target and independent variable but sometimes the hidden relation will not be captured by the model as the data may be changing dynamically along with other hyperparameters (like pressure in the oilfield, chemical concentration, drill depth in our case) resulting in resulting in reducing accuracy. Hence continuous models are necessary for deployment in real-life scenarios where reduced performance is not preferred.

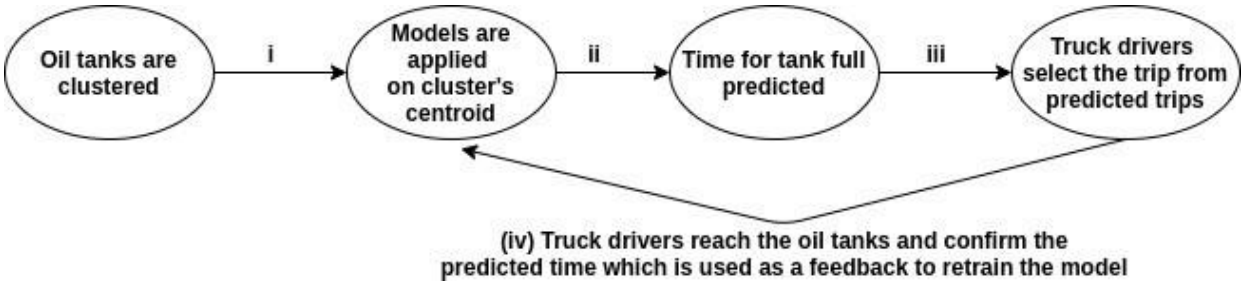

Figure 1: Solution Workflow

3 Results and Discussion

Our POC solves the problem of time-series forecasting along with the clustering of wastewater tanks to reduce the number of models for training and prediction. It also conducts large scale forecasting with a feedback loop to maintain and improve accuracy over a large time interval. We started with statistical models like ARIMA for forecasting and then moved to forecasting specialized model FBprophet and eventually moved onto powerful deep learning model LSTM. We have used DBSCAN to cluster closeby wastewater tanks based on the principle that nearby tanks have similar physical features like chemical concentration, pressure, flow rate, etc. We have trained one model for the cluster centroid and use this model for prediction. We also propose clustering mechanisms based on other features which will be explained

below. Finally, we achieve a workflow where we train models for each cluster centroid and apply continuous learning on all these models to deploy a highly accurate forecasting model on a large scale.

3.1 Data and Design For Prediction Algorithm

On account of business data confidentiality, it is not possible to get the parameters that directly affect the flow-rate of the produced water during fracking from the enterprises. These parameters (chemical concentration, water pressure, etc) directly affect the flow but it is not possible to successfully get the values of these parameters for all the oilfields for training. Also training with so many parameters on a large-scale is not feasible. Hence we will be making use of other indirect parameters to predict the water level of the wastewater tank. We have thought about using a time series forecasting algorithm that will continuously predict the outcome (tank level) for the next time-period (t+1) using data of the previous time sequence. This process can be recursively repeated to predict water level for t+n time-periods. As mentioned above in research methodology it is not possible to get genuine time-series data from oil and gas enterprises hence we have generated a dataset on our own after observing various dependent variables and consulting people within our organization. Our main goal is to apply a feedback loop and to deploy and enable the model in a real-life scenario. For the initial training, we have considered predicting the volume only for one tank with 21000-gallon capacity and a time period of one hour.

3.1.1 Parameters for Prediction of wastewater level

- Capacity of the tank
- Water level of the tank during previous timestamp: V(t-1)
- Timestamp: t
- **Time-period:** 1 hour
- **Target Variable:** capacity of the tank during the next or t+n timepriod: C(t+1) or C(t+n)

3.2 Results of forecasting models

We will discuss the results and inferences of the AI models used along with the feedback loop applied below:

3.2.1 ARIMA

We made use of the Arima model as a level 1 model for rolling forecasting. We normalize the data during preprocessing. The parameters used for fitting are p=6 as the autocorrelation graph shows maximum correlation at lag=6. D=0 as no trend is observed hence differencing is not used. We plot an autocorrelation graph to get the p-value and plot time series graph to pick

the d value. Q is kept as 0 (default value in most scenarios). We get an RMSE error of 6 and observe that Arima is not able to forecast the high and low values but forecasts the intermediate values with less error. The model is unable to find the relation between the current data point and future time lag as there seems to be a hidden relationship between the dependent and target variables not captured by the model or due to some concept-drift.

3.2.2 FbProphet

We found a drastic increase in accuracy after using Facebook's Prophet Algorithm. We normalize and regularize the data and after doing some hyperparameter tuning the model gives a low RMSE of 3.49 along with low uncertainty and seems to closely mimic the actual time series. We define prophet object with daily seasonality as true and increase changepointpriorscale to 0.5 from 0.05 along with increasing Fourier order to increase flexibility in the model. We add hourly seasonality with a time-period of 9. We predict for 40 days in the future. We plot a predicted vs actual output graph on the test data set and see the predictions to be very accurate, following the time series. The library has an inbuilt feature to add seasonality based on holidays and special days. Hence we can include this feature to take into account all the holidays and special days. Hence we can include this feature to take into account all the holidays and can specify the maintenance days to fully automate the system. We included this feature in the time series to keep water tank level zero during maintenance days (all days already specified) and public holidays (USA specified).

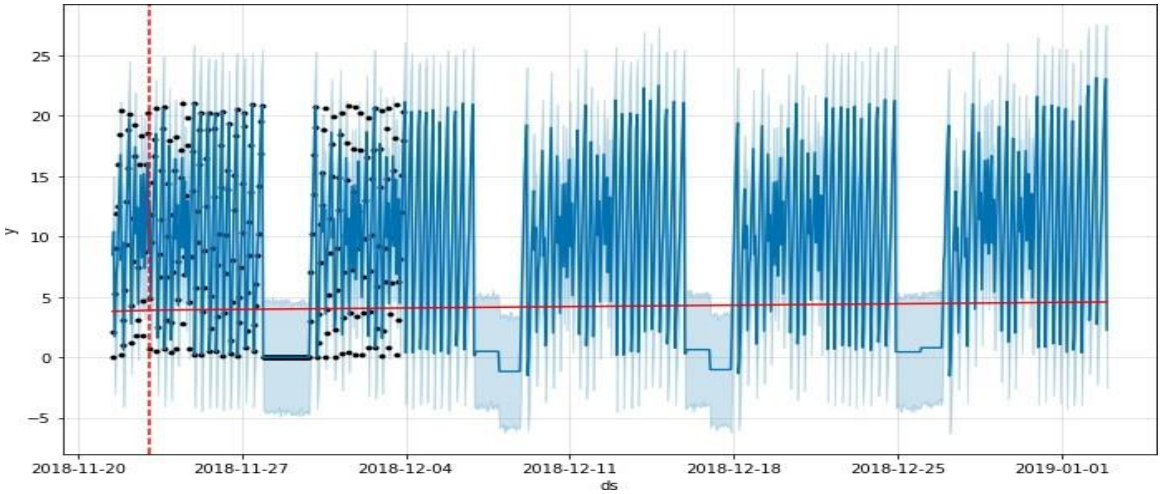

Figure 2: FbProphet

3.2.3 LSTM

This model gives us the highest accuracy as expected from a deep learning model and thus is one of the best models to use for time-series forecasting. We do preprocessing on the data, removing null values, regularizing and normalizing the data.

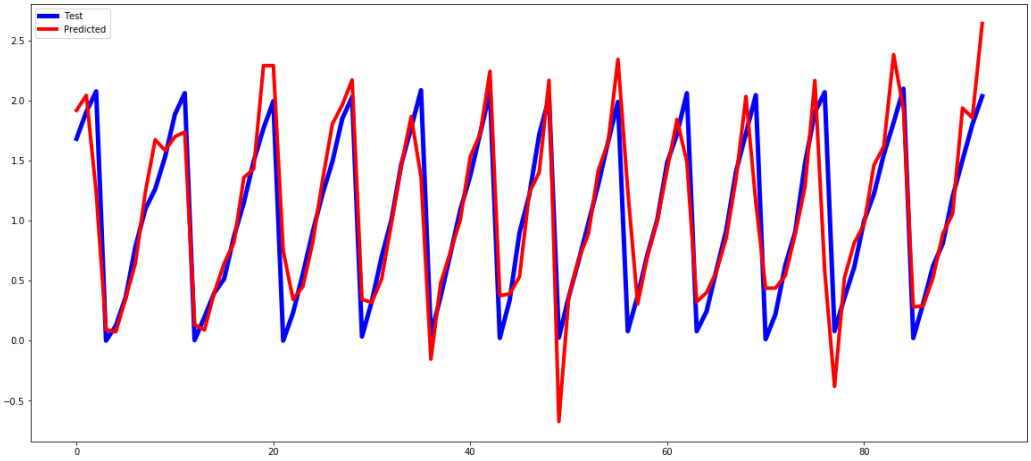

Figure 3: LSTM

We convert the time-series to a supervised dataset with the dependent variables being the previous 6 time-steps and the target variable to predict being the future timestep:t+1. We define a vanilla LSTM with a single hidden layer of 100 LSTM units and an output layer to make the predictions. We train the datapoints for 10,000 epochs and batch size of 4 with the validation loss reaching around a minimum of 0.187. This model gives the best results among all the models we tested. However, we can increase the accuracy even more and also reduce overfitting by using a lot of normalized, regularized data points (1,00,000) for proper training, validation, and testing. However, this was not possible as we have less data. We get a RMSE error of 0.340 and the predicted time-series curve closely mimics the test data however there are random aberrations on low and high values.

3.2.4 Feedback loop with Arima

We tried using the ARIMA model for continuous learning using a point-based learning approach but we did not get expected results with the amplitude of the time series decreasing with increasing time period. It seems that the q value i.e the lagged error value had a dampening effect on the predicted variable.

3.2.5 Feedback loop with LSTM

For employing the feedback loop we load the LSTM model we had previously trained. We use mini-batching based continuous learning. We collect a batch of 200 data points from the data stream. This data is converted to the time-series data frame as required by the LSTM model along with other preprocessing processes (removing null values/aberrations, normalization, regulariza- tion). This time-series is then converted to a supervised problem with dependent variables being the 6 previous time lags and target variable been future (t+1) time-period to predict. We fit this new data on the old model using model.fit function and the model continues training on the new data. Weights are not reset and rapidly changed

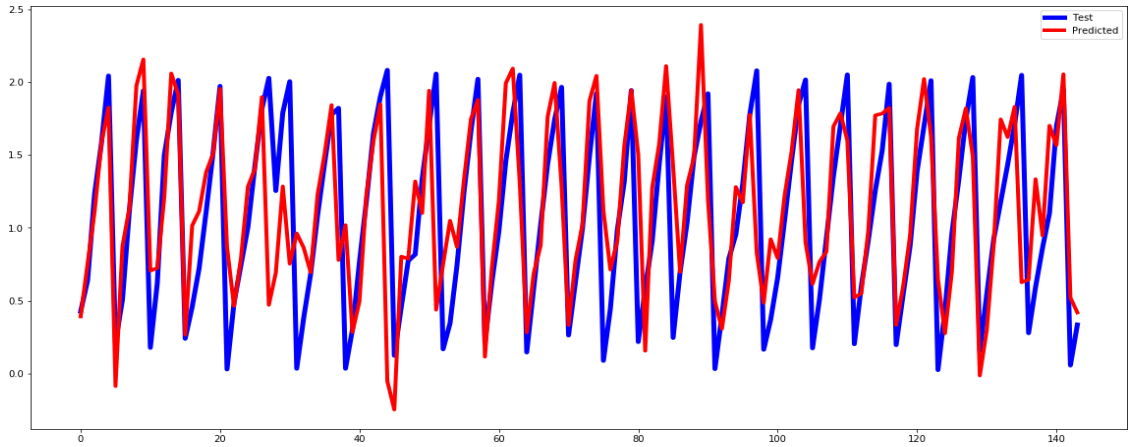

Figure 4: LSTM with feedback loop

while retraining. After retraining we see an increase in the RMSE error from 0.340 to 0.500 but if we keep retraining on new data from the feedback loop the accuracy will keep on improving.

3.3 Clustering for large scale forecasting

Premise: Storage tanks that are located in the same oilfield, as well as tanks that are located in the same geographical area nearby to each other, will have almost the same wastewater filling rate assuming same fracturing process with same chemical concentration is carried out due to similar geographical structure. Storage tanks with similar capacity in a similar geographical area will fill in a similar time.

Prediction Mechanism: Hence we will cluster the storage tanks and designate a cluster centroid. We will train a model for the cluster centroid and use this model to predict the capacity of any tank at the time (t+1) belonging to the same cluster. This will reduce the number of models to train to predict the capacity of all the tanks thus reducing computation and storage overhead on the server. Tanks belonging to the same locality and having the same capacity are clustered together and hence will use the same model for prediction.

Process: After data cleaning, datapoints are clustered using DBSCAN with metrics as follows: Epsilon: 1.5 km, minpts (Minimum points with eps radius): 1. Then the Centroid of all the clusters is found and data is plotted with original vs cluster centroids. The distance between farthest centroids is 410.383 km. Dataframe with latitude, longitude of all the oilfields and also their respective centroids, apinumber and storage tanks with their capacities is created which will be later used for deploying the models individually for each tank.

3.3.1 Analysis of other possible Clustering Mechanisms

We have used DBSCAN to cluster the data points and train the model on the centroids

data. We will then use the same model on all the data points (wastewater tanks)in the vicinity. However, the oilfields may start at different times and may have different capacities. We have considered other clustering mechanisms like:

1. **Cluster on the basis of similar flow rate/similar time-series graph:** Datapoints which get filled at the same rate may be clustered together even if they are far apart as they have similar flow rates.

    - ***Problems and solutions***: When the points are not located close to each other there may be a change in the flow rate with time leading to decreased accuracy. To counter this problem daily clustering can be done and accordingly, the model can be retrained but it will cause computation overhead.

2. **Clusters on basis of same initial parameters**: Another way to cluster the data points is based on the initial parameters like the pressure of water during fracturing, depth of oilfield, chemical concentration, etc.

    - ***Problems and solutions***: If any of the parameters are changed the model will fail however like the above solution clustering can be re-done periodically and models can be retrained however there will be computation overhead.

3.3.1 Problems with RBML

Initially, we had considered applying a rule-based model for forecasting. To apply it we need to devise the rule space from which optimum rules may be selected to reduce the rule space or more rules may be added using a genetic algorithm to create better rules. However, as the data is a time-series any data point is dependent mainly on the lagged (previous) observations. Any rules created manually/by the model will not be able to see the interdependence between subsequent data points. Hence even if we can capture some relations like weekly seasonality etc via some rules, overtime on validation data the model will suffer from concept drift. Hence due to these problems, we could not apply RBML.

4 Conclusion

In the current trend of the Gig Economy, the nature of the work system is changing where an employee wants to do work according to his time convenience so he can get more profit out of his work. We have incorporated this model of employment in our POC. Now the truck driver can use the app to select from a list of trips from various locations and time according to his convenience enabling gig economy. We have implemented various models that are best suited for our use case to predict the time of the wastewater tank getting full. We have tried to solve the problem of applying the time-series forecasting model on a large scale for every oilfield by clustering wastewater tanks. This allows us to train the model for a cluster centroid not individually for all wastewater tanks. Hence we can use the centroids model for all the tanks in the cluster enabling large scale forecasting. The data we are

getting for the prediction depends on various physical factors so the flow rate over a period of time. Hence we have implemented a feedback loop that will re-train the model with new data and give better prediction with time. Sometimes the data from the IoT devices can be faulty, hence we will promote positive human behavior by incentivizing the driver for manually entering the correct reading.

In our future works, we will use the blockchain/market place data to evaluate the accuracy of the AI algorithms and deploy advanced deep learning models with a better feedback loop for prediction. We will also implement and compare different clustering mechanisms and create an ensemble of AI models which will give the highest cumulative accuracy for the whole system. In a nutshell, the solution tries to overcome the problem of time delay and process halting that was not solved using our blockchain solution. Hence blockchain with AI and reputation mechanism can be used to solve real-world problems efficiently, enabling large scale forecasting and promoting the gig economy.

References

[1] AV Kushwaha, HV Natarajan, K Jayakumaran, P Gupta, Raksha, SK Karki. (2019). "IoT Based Blockchain Solution To Endorse Positive Human Behaviour", ISRDC@IITB, Third workshop on blockchain technologies and its applications.

[2] DR. EMILIA ISTRATE, JONATHAN HARRIS. (2017). The Future of Work. Retrieved from https://www.naco.org/featured-resources/future-work-rise-gig-economy

[3] Detecting text. (2021). Retrieved from https://docs.aws.amazon.com/rekognition/latest/dg/text- detection.html

[4] Sangarshanan. (2018). Time series Forecasting - ARIMA models. Retrieved from https://towardsdatascience.com/time-series-forecasting-arima-models-7f221e9eee06

[5] Quick Start. (2021). Retrieved from https://facebook.github.io/prophet/docs/

[6] Recurrent layers. Retrieved from https://keras.io/layers/recurrent

[7] Christopher-Olah. (2015). Understanding LSTM Networks. Retrieved from https://colah.github.io/posts/2015-08-Understanding-LSTMs

[8] CORELS Overview. Retrieved from https://corels.eecs.harvard.edu/corels/

[9] DBSCAN. Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html