Õpet Technical White Paper

Version 3.0
19 June 2018
First Published Version 1.0 on 15 May 2018

General White Paper | Website | Videos

Abstract

The increasing demand for ad-hoc supporting education services combined with the rapid developments in blockchain, artificial intelligence and machine learning technologies call for an amalgamation of the fields manifested in a technologically-responsive, available and cost-effective solution to learning. In a pioneering application of artificial intelligence in the field of education, we present the highly contextualized Õpet companion chatbot - complementing higher secondary/ pre-college National Curriculum worldwide through a 24/7 digital tutor powered by AI and machine learning.

1. Introduction

The existing secondary and high-school education system across the world is fundamentally in need of modernization and refinement, for the most part factoring the individual learning styles and differing educational needs of students from dissimilar financial and familial backgrounds.

As a result, a large number of students require extra-curricular or ad hoc educational assistance commonly sought through means of a professional tutor. While this service adequately addresses the knowledge gap for the majority of students who benefit from it, it is a luxury far removed for the countless students in rural communities, the overwhelming majority of which come from families already hard pressed to afford standard curricular tuition.

Consequently, the aforementioned students who are unable to stay abreast with the pace and depth of the traditional education system are not afforded an opportunity to bridge the knowledge gap, thereby missing significant opportunities for bursaries and/or scholarships, general admission into prestigious universities, as well as having limited options in terms of jobs and career prospects. Inevitably, the cycle continues onwards into subsequent generations, with future students experiencing the same barriers to education and life's opportunities.

In a coordinated effort to address these and similar issues in the existing education system and tutor network, Õpet presents the Õpetbot, an education chatbot powered by artificial intelligence (AI) and machine learning, manifested in the form of a mobile application.

In addition to serving as a reliable and affordable digital tutor companion, the Õpet backend system is designed to *learn* about each student connected to the network through the assimilation of user personality data manifested in the chat history. Invaluable in a number of applications, this personality profiling information will be used to compile research and statistics into the millennial student generation, as well as recommend

appropriate undergraduate courses of study and career paths.

With personal information stored securely on Õpet's own hyperledger[1][14] framework, the capacity for storing additional student information is available, most notably for academic learning material, results and certifications, as well as generalized user content for entertainment and social use.

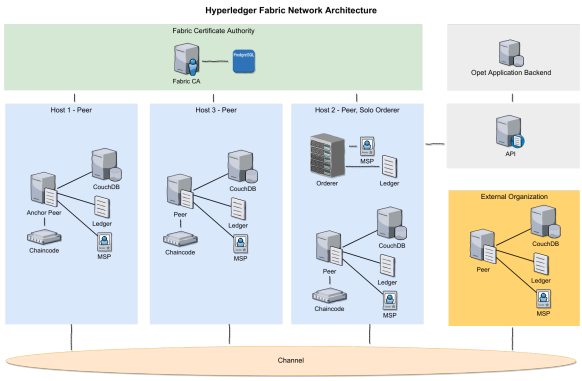
After the introduction of Bitcoin[2], the advent of blockchain, hyperledger and related technologies has brought forward unbounded potential for systems and applications previously considered beyond the reach of the current era. With artificial intelligence evolving more rapidly than experts are able to track[3], it has recently become possible for AI-based applications to replace entirely an increasing variety of outcome-specific human functions.

Õpet is in essence an artificial intelligence continually programmed to learn, understand and ultimately teach high-school grade curriculum to students. While it will not replace the existing education system at this stage, the technology presented in the form of a digital chatbot companion will serve to complement and expand on traditional school-based tuition in an effort to bridge the prevailing knowledge gap between students' understanding and standardized learning outcomes.

2. Network Components

Õpet Hyperledger Fabric Network has two types of participants. Õpet, the main network organization, runs orderer and CA[9] server, has access to network configuration and write operations. Partnered external organizations run one or more peers and have access to a read-only copy of the blockchain data.

Õpet uses blockchain to store the data that can be observed by external organizations who have read-only access to the blockchain.



The network consists of the following main components:

Fabric Certificate Authority (CA)[9], which generates and manages the certificates for other network components. This includes a PostgreSQL[10] server for Fabric CA database; one Solo Orderer which orders transactions and combines them into blocks; and three Peers which hold the blockchain data using CouchDB[11] as database.

API communicates with the blockchain to provide simplified interface for the Õpet Application. As the external organization runs one or more peers, they don't need to run Fabric CA server, instead accessing the network via the certificate generated with openssl[12] or cryptogen[13] tool.

3. Proof of Authority Consensus

Õpet works on a proof of authority[8] consensus where each entry made into the hyperledger is validated by permissioned nodes. Where data has been tampered with, the majority nodes will fail, rejecting conflicting altered information.

In addition, Õpet's hyperledger network collaborates with the Ethereum

mainnet for transactional processing. The use case is as follows: In the process of an admission application, the user provides his private hash key to the university at which he is applying. The university requests from Õpet to the user's profile, for which access the university is charged in Opet tokens (OPET).

As the above transaction takes place on the Ethereum mainnet, one of the data fields contains the private user hash given to the university. Using the hash, the API will extract the user information from the Õpet blockchain, where the data is stored in the hyperledger. This process makes up the Õpet proof of authority[8] system, where such a blockchain- hyperledger network combination is unique and a first in the cryptocurrency industry.

As the hyperledger is simply a storage facility validated by external nodes, it does not have a consensus protocol of its own, while the API provides access to the private hyperledger.

The structure of Hyperledger Fabric provides the ability to toggle the visibility of nodes. This tool will be deployed at maximum utility to maintain data privacy and integrity in line with GDPR and global privacy protection legislation. Should an anonymous ledger/database query be required for the purpose of acquiring or curating further analytics, this function can be performed using an app layer which servies the function of a blockchain explorer.

4. Artificial Intelligence

An AI-powered highly deliberate chat application can be a powerful means of guiding or nudging a student through topic by topic reviews.

Identifying unmotivated students using AI is a challenging task that involves undertaking data acquisition schemes to incorporate a broad spectrum of features.

Curating metrics such as the number of questions scanned, quizzes attempted and their respective scores provides insight into the students' sincerity and strength, which assist in addressing this issue.

Through classifying students into distinct levels, strategies can be designed and adopted in line with different students' needs, which contributes towards ensuring accurate, efficient and highly-personalized recommendations.

Problem Statement

The goal is to classify students based on their level of motivation into various classes. The following tasks are involved:

1. Consolidate and collate the data of simulation and test users. Remove sensitive information.
2. Preprocess the data by scaling features and split into training, validation and test sets.
3. Train different models of classifiers.
4. Optimize the different parameters of the different models to improve accuracy on validation set.
5. Test all models on test set and select the model with best performance.

The final model should be able to classify students based on motivation level with high efficiency.

Algorithms and Techniques

There is no single answer about which is the best classification method for a given dataset. Different kinds of classifiers should be always considered for a comparative study over a given dataset. Given the properties of the dataset, we might have

some clues that may give preference to some methods.

Having said that, four classifiers would be appropriate: Random Forest, Decision Trees, Gaussian Naïve Bayes and Support Vector Machines.

Data Preprocessing

As there substantial difference between the mean and standard deviation of the different numerical features, it is important to bring these features on an equivalent scale. It is further necessary practice to perform some type of scaling on numerical features.

Applying a scaling to the data does not change the shape of each feature's distribution; however, normalization ensures that each feature is treated equally when applying supervised learners. A Min Max Scaler transformation is to be used to scale all the features to zero mean and unit variance.

Once scaling is applied, observing the data in its raw form will no longer have the same original meaning, as exampled below.

0	0.000000	0.047619	0.028302	0.0	0.000000	C	A	A
1	0.310827	0.238095	0.150943	0.5	0.448696	B	B	A
2	0.816609	0.523810	0.452830	0.5	0.853296	B	B	B
3	0.230314	0.190476	0.174528	0.5	0.384283	C	A	C
4	0.705661	0.452381	0.339623	0.5	0.764550	B	A	C

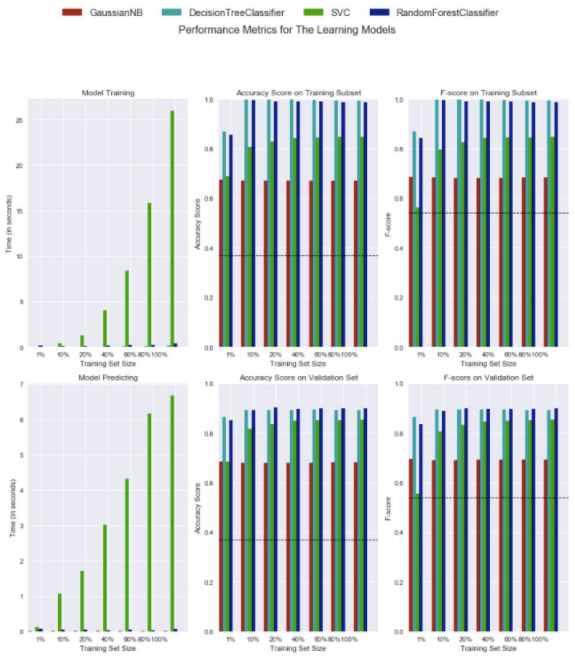
The final data set, as illustrated in the table above, was to be split into training, validation and test set to be used for optimization of the models. To maintain a fair splitting, the data was shuffled first, and then out of the 100 examples, 20 percent or 200 examples stored away separately as test data. This subset is not to be used until the very end when the final models are tested. Further the remaining 800 examples, were split into the training and validation sets in the ratio 600:200 respectively, using the train_test_split function.

In order to properly evaluate the performance of each chosen model, it's important to create a training and predicting pipeline that allows quick and effective training of models using various sizes of training data and performing predictions on the validation data.

The training pipeline begins by fitting the data to the various models and then obtaining predictions on the validation set and a subset of the training data for comparison.

Also to observe the effect of increase in examples for training, all the models are trained on 1 percent, 10 percent, 20 percent, 40 percent, 60 percent, 80 percent, and 100 percent of training data respectively. The models were then evaluated on the validation set. These predictions were evaluated using accuracy and F1 score which were stored as separate variables for analysis. Both F1_score and accuracy are evaluated to keep to maintain a solid understanding of the performance.

These metrics are then plotted to get an even better understanding of the performances of the various models. The graphs below illustrate these plots.



A thorough analysis of these graphs makes it abundantly clear that decision trees and random forests have a high F1 score and accuracy score on the validation sets. It can be seen graphically that as the training set size increases, the performance improves in general, although there is no improvement in the performance of the Naïve Bayes classifier.

As can be seen from the graphs, the training and prediction time for SVC increased exponentially with increase in training set size.

The next step was a complication as employing Grid Search which is an exhaustive loop to optimize the parameters of SVC took a lot of time.

Because decision trees and Random Forest were better at performance and training time, the SVC and Naïve Bayes algorithm were dropped, whereas the former two classifiers were taken ahead for optimization and refinement.

Refinement

In order to improve performance of a model, fine-tuning the parameters is important. Grid search technique is to perform an exhaustive search of various combinations of the hyper-parameters and find the best, optimal model. This way the learning algorithm is optimized because the optimal parameters are used to better fit the specific dataset at hand.

The table below illustrates the parameters to be iterated to find the best combination.

Classifier	Best Parameters
Random Forest	'min_samples_split': 10 'bootstrap': False 'criterion': 'gini' 'max_depth': 10 'min_samples_leaf': 3
Decision Tree	'presort': True 'splitter': 'best' 'min_samples_leaf': 2 'criterion': 'entropy' 'min_samples_split': 10 'max_depth': 10 'class_weight': None

Using grid search the parameters were optimized and the before and after results were reported.

Unoptimized model of
RandomForestClassifier

Accuracy score on training data: 0.9014
F-score on training data: 0.9003

Optimized Model of RandomForestClassifier

Final accuracy score on the validation data: 0.9137
Final F-score on the validation data: 0.9132

Unoptimized model of DecisionTreeClassifier

Accuracy score on training data: 0.8921
F-score on training data: 0.8926

Optimized Model of DecisionTreeClassifier

Final accuracy score on the validation data: 0.9117
Final F-score on the validation data: 0.9137

Total Simulation Time= 227.439269066

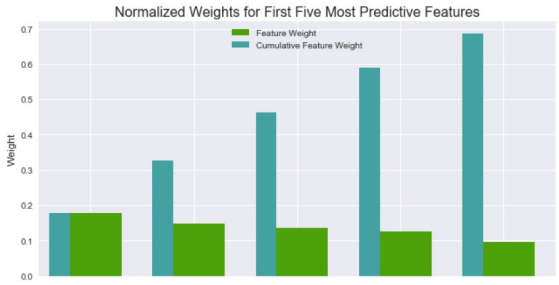
F-score is a fair metric to test in the domain of skewed classes, because it takes into account both precision and recall. The optimized model's accuracy and F-score are better than that of the un-optimized model. This is because some important parameters like max_depth of the tree and criterion for the top splits were fine tuned to get the best F-score.

The min_samples_split is high and min_samples_leaf is low which is appropriate in this scenario of motivation levels.

A type of sensitivity analysis has already been done previously by taking smaller subsets of data for training. The graphs below illustrate clearly that the F_score on the validation data, for these two models is fairly constant. This indicates that the model is robust to variations in input data and is a true predictor of motivation/sincerity level.

Another important task when performing supervised learning, is determining which features provide the most predictive power. By focusing on the relationship between only a few crucial features and the target label, understanding of the phenomenon can be simplified, which is most always a useful thing to do.

In the case of this project, identification of a small number of features that most strongly predict whether a student has high motivation or not, can be done.



Following are the results of the model trained on these five features only.

RandomForestClassifier Final Model trained on full data

Accuracy on validation data: 0.9137
F-score on validation data: 0.9132

RandomForestClassifier Final Model trained on reduced data

Accuracy on validation data: 0.8193
F-score on validation data: 0.7996

DecisionTreeClassifier Final Model trained on full data

Accuracy on validation data: 0.9117
F-score on validation data: 0.9137

DecisionTreeClassifier Final Model trained on reduced data
Accuracy on validation data: 0.8171
F-score on validation data: 0.7985

With reduced features, the F_score is still high. But there is still a significant drop in the F_score of both the models which indicates there are other features that are also important for this prediction.

As a final measure, the models were evaluated using the test data set separated from the original data set in the beginning. A final score for the performance of both the models was obtained. Following are the results.

Test Set Scores of RandomForestClassifier

Final accuracy score on the test data: 0.9141
Final F-score on the test data: 0.9133

Test Set Scores of DecisionTreeClassifier

Final accuracy score on the test data: 0.9116
Final F-score on the test data: 0.9133

Both the models perform equally well in terms of F_score. The result is not affected much even when the models are given unseen data. This implies a high level of robustness and generalization to new data. These results make the models very reliable and they can be trusted to be good classifiers of motivated vs unmotivated students.

5. Entities in the Network

The Õpet network will comprise two primary user types. These include but are not limited to students who utilize the platform for educational advancement and social networking with peers, and educational institutions [nodes] who validate and access student information in the form of academic records and unique personality profiles.

6. Identity Verification (KYC)

All users of the Õpet platform will go through an identification process in accordance with KYC. As such, it is impossible for users to create duplicate user profiles or to use the platform with false or fraudulent personal identification information.

7. System Architecture

The Õpet framework comprises a blend of two distributed ledger technologies, specifically hyperledger and blockchain. Where crypto-assets relating to the token and value transfer within the platform are concerned, management will be facilitated by the Ethereum blockchain[4].

When expressivity is examined, the programming language used will fall between the spectrum of Bitcoin script[5], positioned on the end of simplicity, and Ethereum[6], a complex ledger designed as a generalized compute resource with a myriad of applications.

The full expressivity of the Õpet framework remains to be determined as the platform progresses through its developmental stages. It is however clear at present that a Turing-complete[7] language is unnecessary in supporting the platform and its smart contract ecosystem, predominantly due to the varied yet limited use cases of the Õpet system and its underlying AI technology.

8. Internal Organization Structure

The Õpet organization structure in terms of certificate authority / membership service provider (MSP)[9] is set up as follows:

Fabric.opetbot.com

1. ca.fabric.opetbot.com - certificate authority
2. Orderers
    a. Orderer.fabric.opetbot.com
3. Peers
    a. Peer0.fabric.opetbot.com
    b. Peer1.fabric.opetbot.com
    c. Peer2.fabric.opetbot.com
4. Users
    a. Admin@fabric.opetbot.com
    b. User1@fabric.opetbot.com

    The user will be used by API application

Transaction endorsement policy: OpetMSP.peer. Any Õpet organization can endorse the transaction.

9. API

The Õpet API currently allows to:

1. Create / retrieve user records
2. Store user json documents
3. Retrieve the list of user documents
4. Retrieve specific documents
5. Store binary files such as certificates
6. Retrieve the list of user files
7. Retrieve specific files

Create User

The *Application* main database serves as a primary source of user data, where *Application* refers to the Õpet application code that uses Blockchain through API to store user data.

The *Blockchain* needs to have a user record in order to be able to bind other user data (documents and files) to it, where *Blockchain* refers to the Hyperledger Fabric based data storage.

The *Application* is free to store some additional data for the user, although this data does not participate in any business logic.

POST /users
Create user record in the Blockchain, generates and returns user UID.

Input:

- user data, POST body, optional

POST Body: json dictionary with additional user data to store (can be empty).
POST Body example: `{"property1": "value", "property2": "value2"}`

Output: 200 OK, Body: json object with user UID

```
{
    "data": {
        "UID": "xxx"
    }
}
```

Retrieve User

GET /users/{xxx}
Retrieve the user by UID.

Input:

- user UID, URL path parameter, required

Output: 200 OK, Body: json object with user data

```
{
    "data": {
          "UID": "xxx",
          "type": "user",
          "data": {
              "property1":
     "value",
              "property2":
     "value2"
          }
    }
}
```

Create Document

POST /documents/{user uid}
Create the json document (psychometric test result, quiz result, etc) for the specified user, generates and returns document UID.
Note: for documents we can use document Hash as UID.
Input:

- user UID, URL path parameter, required
- document, POST body, required

POST Body: json dictionary with the document to store.
POST Body example:

```
{
    "type": "psychometric",
    "document": {
```

```
        "predictions": [ ....],
        "contributions": [...],
        "interpretations":
[...]
    }
}
```

Note: the above is the recommended format for the Application to store the data - the dictionary with two top-level fields, "type" (the document type - psychometric, quiz, etc) and "document" (the actual document content). Although API does not enforce this structure.

Output: 200 OK, Body: json object with document UID (Hash)

```
{
    "data": {
        "UID": "xxx"
    }
}
```

List Documents

GET /documents/{user uid}
Get the list of documents for the specified user.

Input:

- user UID, URL path parameter, required

Output: 200 OK, json object with the list of document UIDs for the user

```
{
    "data": {
        "documents": [
"doc_uid_1", "doc_uid_2", … ]
    }
}
```

Retrieve Document

GET /documents/{user uid}/{document uid}
Get the specific document for the user. The API verifies that the requested document (document uid) belongs to the specified user (user uid).

Input:

- user UID, URL path parameter, required
- document UID, URL path parameter, required

Output: 200 OK, json object with the document content

```
{
    "data": {
       "type": "psychometric",
       "document": {
            "predictions":
[....],
            "contributions":
[...],
            "interpretations":
[...]
        }
    }
}
```

Note: the content under "data" key is exactly the same as was sent to the "Create Document" API method.

File API

The file API allows to store binary files uploaded by the user, such as certificates. Actual files are stored on AWS S3 private bucket, where the API creates the hash of the file and stores the hash into blockchain.

Through this practice, we guarantee that the file cannot be modified without invalidating the hash stored in the blockchain. Additionally, the blockchain database is not overloaded with binary data.

The API will return path of the file on S3 to the *Application*. *Application* backend can the query the file directly from S3 and download or display it for the user or generate the temporary signed URL for download.

Files can also be marked as valid or invalid in instances where the user uploads the incorrect file through the Application. By default, files are marked as valid.

Create File

POST /files/{user uid}
Upload the file for the specified user, generates and returns file UID.

Note: for files we can use file Hash as UID. This way we can also guarantee that the same file was not uploaded twice.

Implementation note: The API saves file to the AWS S3, hashes the file and stores the file hash to blockchain.

Implementation note:
By default, the uploaded file is marked as valid.

Input:

- user UID, URL path parameter, required
- file, POST body, required

POST Body: file to store.

Output: 200 OK, Body: json object with file UID (Hash)

```
{
    "data": {
        "UID": "xxx"
    }
}
```

List Files

GET /files/{user uid}
Get the list of files for the specified user.

Input:

- user UID, URL path parameter, required

Output: 200 OK, json object with the list of file UIDs along with "valid" flag values

```
{
    "data": {
        "files": {
          "file_uid_1": {"valid": true}
          "file_uid_2": {"valid": false},
           …
        }
    }
}
```

Retrieve File

GET /files/{user uid}/{file uid}
Get the specific file for the user.

Input:

- user UID, URL path parameter, required
- file UID, URL path parameter, required

Output: 200 OK, file content or temporary signed file URL.

Mark File As Invalid or Valid

Mark file as invalid:
POST /files/{user uid}/{file uid}/mark?invalid=1

Mark file as valid:
POST /files/{user uid}/{file uid}/mark?invalid=0

Input:

- user UID, URL path parameter, required
- file UID, URL path parameter, required
- invalid, URL query parameter, required, boolean (1 or 0)

POST body: empty.

Output: 200 OK

Delete File

DELETE /files/{user uid}/{file uid}
Delete the specific file for the user.
File is deleted from S3 and removed from current state in blockchain (but the file hash is kept in the blockchain history).

Input:

- user UID, URL path parameter, required
- file UID, URL path parameter, required

Output: 200 OK.

10. External Organizational Structure

The external organization structure in terms of certificate authority and membership service provider (MSP) is set up as follows:

Fabric.external.org

1. ca.fabric.external.org (static certificate, no server)
2. Peers
    a. Peer0.fabric.external.org
3. Users
    a. Admin@fabric.external.org

11. Token Economics

The native currency of the Õpet ecosystem is the Õpet's utility token (OPET), with 100,000,000 tokens minted in preparation for the token sale.

The primary use case of the Opet Token (OPET) is for students to purchase educational services on the Õpet platform on an ad hoc or subscription basis. In lieu of traditional scholarship or bursary funds disbursement methods, sponsors and philanthropists can also transfer funding directly to respective students by means of Opet Tokens (OPET).

For interested parties such as universities, academic institutions and prospective employers to view a specific students' data, access can be purchased using Opet tokens (OPET). To maintain anonymity of all other data stored, and protect each individual's privacy in line with GDPR, access to the blockchain does not imply an ability to view the data belonging to other students. Interested parties will require the unique hash code belonging to the specific student for which they are inquiring, which will match and confirm his or her identity respective to the data accessed.

Tokens will also be given as a reward to students who actively engage with Õpet and in doing so promote platform growth. Primary methods of doing this include writing and sharing blog articles, social media posts. The above represents standard token interaction with traditional cryptocurrency projects.

Opet Tokens (OPET) are not restricted to the platform, but will also be listed on as many exchanges as possible for purchase or sale in the open market. In an effort to mitigate standard token value volatility, Õpet will purchase a predetermined amount of tokens back from the open market in regular intervals.

12. Hyperledger Integration

Õpet's own instance of Hyperledger[1] is designed for the primary function of storing platform data as opposed to utilizing a traditional database such as PostgreSQL.

Our Hyperledger solution prevents corruption of data otherwise vulnerable to or unforeseen system failure or fraudulent alteration by students or other individuals, and offers tamper free data storage stored in blockchain.

Moreover, as a cryptocurrency is not required for the data storage element of the Õpet service offering, Hyperledger was the natural solution.

13. Scalability

When scalability is examined, hyperledger presents greater scalability with reduced cost when compared to the Ethereum network, which is comparatively costly per transaction for the expected volume potentially taking place on the network.

Õpet has successfully created a smaller scale instance of the platform which will be expanded upon once funding from the token sale is attained, at which point thousands of master nodes will be hosted.

As a blockchain technology, each server will contain a full copy of the hyperledger, wherein student records will be securely stored and confirmed by each master node.

Thereafter, third party partners such as universities, National Registrars, or Education Ministries of different countries will be invited to run their own validation nodes on the Õpet network without the ability to edit or write data. The independent external master nodes complete the decentralization and ultimate security of the platform, preventing Õpet from being able to edit user data.

14. Future Directives

Future improvements for the network setup comprise the following non-exhaustive aspects:

1. Setup the external organization (generate crypto material and setup the peer for the external organization)
2. Automatic backup procedure
3. Centralized logs collection
4. Centralized monitoring system
5. Centralized deployment system (docker swarm / kubernetes etc)
6. Redundant orderer setup (two or more orderers with Kafka / Zookeeper cluster)

In addition to the above developmental enhancements, Õpet has several potential objectives and avenues for growth as a business and further applications of our AI technology. At present, these include the offering of Õpet educational events as online courses, as well as the use of artificial intelligence for:

1. AI-based student learning
2. AI-based testing and assessment
3. AI-bot to help buyers navigate choices and created education programs
4. AI to AI smart contract creation and management to streamline requests among
5. buyers and sellers
6. Use of augmented reality (AR) in the classroom and online education experiences

While progress on the above is likely only to commence after the successful token sale, it can be tracked and referenced on Github on the following page: https://github.com/opetfoundation/eco_blockchain.

REFERENCES

[1] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith A. Sorniotti, C. Stathakopoulou, M. Vukolić, S. W. Cocco, J. Yellick, “Hyperledger Fabric: A Distributed Operating System for

Permissioned Blockchains," https://arxiv.org/pdf/1801.10228.pdf, 2017.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," http://bitcoin.org/bitcoin.pdf, 2008.

[3] Y. Shoham, R. Perrault, E. Brynjolfsson, J. Clark, "Artificial Intelligence Index, 2017 Annual Report," http://aiindex.org/2017-report.pdf, 2017.

[4] E. Foundation, "Ethereum's white paper," https://github.com/ethereum/wiki/wiki/White-Paper, 2014.

[5] "Bitcoin Script," https://en.bitcoin.it/wiki/Script, accessed on May 18, 2018.

[6] Dr. G. Wood, "Ethereum, a Secure Decentralised Transaction Ledger, EIP-150 Revision, " http://gavwood.com/paper.pdf, 2014.

[7] A. M. Turing, "On Computable Numbers, With an Application to the Entscheidungsproblem," http://www.cs.ox.ac.uk/activities/ieg/e-library/sources/tp2-ie.pdf, 1936.

[8] POA Network, "Proof of Authority: consensus model with Identity at Stake," https://medium.com/poa-network/proof-of-authority-consensus-model-with-identity-at-stake-d5bd15463256, 2017.

[9] Hyperledger," Welcome to Hyperledger Fabric CA (Certificate Authority)," http://hyperledger-fabric-ca.readthedocs.io/en/latest/index.html, 2017.

[10] PostgreSQL, https://www.postgresql.org

[11] Apache CouchDB, http://couchdb.apache.org.

[12] Openssl, https://www.openssl.org.

[13] Hyperledger, "Building Your First Network," http://hyperledger-fabric.readthedocs.io/en/release-1.1/build_network.html, 2017.

[14] Hyperledger, "A Blockchain Platform for the Enterprise," http://hyperledger-fabric.readthedocs.io/en/latest/index.html, 2017.