DeepSuccess - Predict the Success of Tech Startups

Davide Di Dio

***Abstract*—Many interesting applications emerged with the increasing popularity of deep learning. This project explored natural language processing and visualization techniques as well as two neural network architectures to classify ICOs. The first network focused on the ICOs white-paper with a bi-directional LSTM attention network. The second targeted the ICOs website structure with a Graph neural network as well as page topics with Latent Dirichlet Analysis.**

I. Introduction

CRYPTOCURRENCIES (e.g, Bitcoin, Ripple, Ethereum) popularity has seen an unprecedented increase in the last two years. The market capitalization of cryptocurrencies has increased rapidly. The high trading volume of cryptocurrencies is comparable to the trading volume of the New York Stock Exchange in 2017 [1].

The crowdfunding of digital coins does not need to go through the required standard procedure of Venture Capital investment because of their decentralized nature. Instead they go through initial Coin Offerings (ICOs) [1]. The ease of crowdfunding creates opportunities for unscrupulous businesses to use ICOs to execute "pump and dump" schemes. The ICO founder drive up the value of the crowdfunded cryptocurrency and then quickly "dump" the coins for a profit. Even if ICOs are able to provide fair and lawful investment opportunities, it is imperative to be able to differentiate between rightful and fraudulent projects. Additionally, the decentralized nature of cryptocurrencies poses significant challenges for government regulation, creating the perfect environment to extort investor's money.

Some key statistics from ICORATING report for the second and third quarter of 2018 [2] [3]: 827 projects raised \$8,359,976,282 in total funds. While 55% failed to complete their crowd-funding, the other ICOs had a median return on investment(ROI) of -55.38%. This means that the majority of ICOs are worth less than 50% of their initial sale value. This statistic is a bit better in the third quarter, where the median ROI is at -22%. Figure 1 illustrates the ROI when grouping the ICOs by industry. This is a major problem for investors that seek reliable investment opportunities.

In this project, we aim to predict the success of Initial Coin Offering (ICO). Since by their nature ICO are high risk high reward investments, this project attempts to reduce the risk of the investment with an automated deep learning system that classify ICOs by their potential success.

II. Problem Statement

We want to predict the success of an ICO project using deep learning. In other words, we want to use very little (or no) prior knowledge about ICO projects, and learn the features and their importance directly from the data. The model we use here is a supervised learning model.

Advisors: Nguyen Thanh Tam, Karl Aberer

[1]https://coinmarketcap.com/

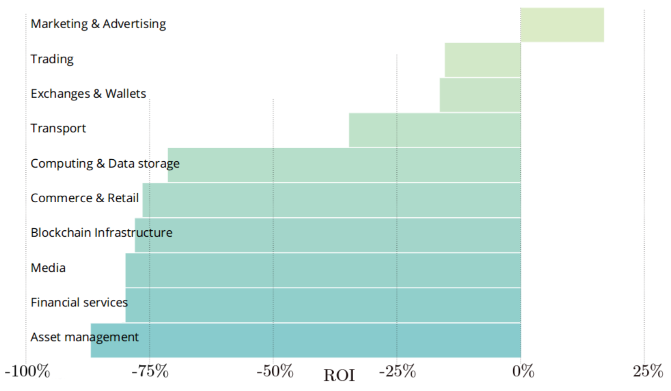


Fig. 1. Average return on investment by industry. 2018 third quarter.[3]

Problem 1 (Success Prediction): Given an ICO project x, which includes different aspects of its publicly available information, we wish to find a model F, that maps the input x to an output y, which indicates the success of the ICO project:

$$y = F(x) \tag{1}$$

To classify the ICOs by their success, we need to define a ground-truth. Since ICOs are very volatile, the notion of success can be a bit fuzzy.

Real-value ground-truth. Before any description, let us introduce some required notations: We define S as the ICO sale price and V_i as the market open value i days after the ICO sale ends. Now, we can define the first ground-truth as:

$$gt_1 = sigmoid(\frac{V_D}{S}),\ D = 120 \tag{2}$$

This ground-truth was inspired from the paper [4]. Let us now analyze this definition with some plots:

As we can see in Figure 2 and 3, many ICO fluctuate by a large amount from one day to the next. Thus defining a ground-truth with a single value is very unstable and changes drastically with respect to the value D. We thus preferred an alternate more stable definition for the ground-truth:

Discrete ground-truth. Define the series $V = \{V_i : i = [1, D]\}$ and the function $above(V, S) = \sum_{i=1}^{D} \mathbb{1}\{V_i \geq S\}$ that counts the number of days where the market price is bigger than the sale price. Finally, we can put ICO in three distinct categories :

- Failed : The ICO did not reach the required founding or $above(V, S) < 0.1 \times D$.

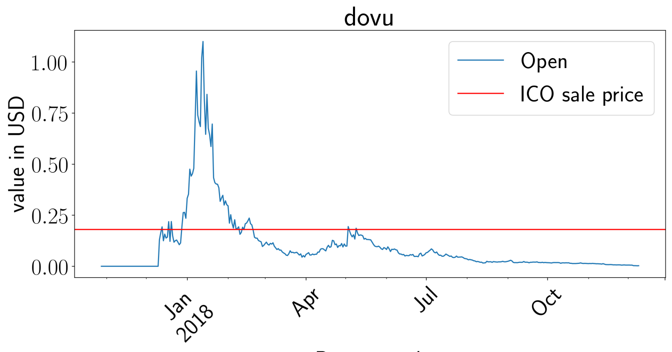


Fig. 2. Market value and sale price of the DOVU ICO.

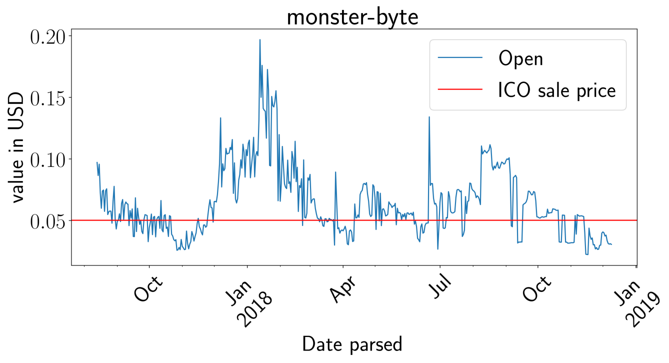


Fig. 3. Market value and sale price of the MONSTER-BYTE ICO.

- Success : $above(V,S) \geq 0.9 \times D$.
- Risky : $above(V,S) \geq 0.1 \times D$ and $above(V,S) < 0.9 \times D$.

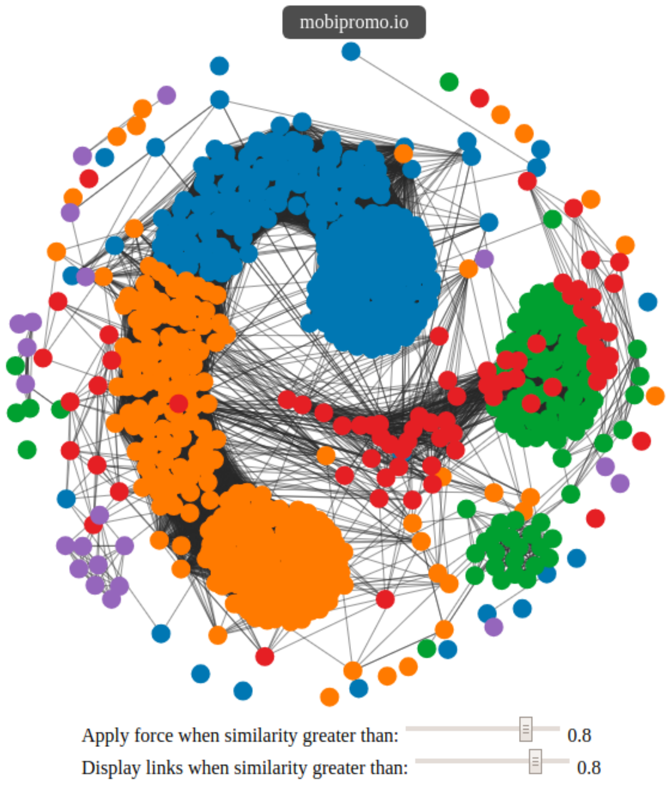
This definition is useful if we want to predict the short time success of an ICO. One can adjust the window on which to consider the market data. However, since most of the ICO are quite young, we cannot grow the window too much yet. We finally decided to take $D = 120$ Days ≈ 4 Month. For the next sections of this report, we will only consider the second definition of the ground-truth. At the end, we collected 1512 ICOs, we refer the ground-truth of the i-th ICO as y_i.

III. Methodology Overview

A. Data Collection

We used Python libraries collect and extract the data. Selenium is needed to load JavaScript generated websites.

- ICO description: Contains the name, status, sale price and the links for their websites/white-papers. More information is collected such as social media links and usage frequency but those features were not used for this project. This information is scraped from https://ico.coincheckup.com.
- Market data: The daily value of the coin is used to build the ground-truth. Collected from https://coinmarketcap.com/



Apply force when similarity greater than: 0.8
Display links when similarity greater than: 0.8

Fig. 4. Latent Dirichlet Analysis visualization

B. Exploratory data analysis

With the data collected, we want to explore the data before jumping to the classification task. The first visualization displays the major topics of the white-papers. Each white-papers goes trough a Latent Dirichlet Analysis (LDA) pipeline. Then a similarity metric is computed between of each documents.

$$sim = 1 - \frac{\|v_i - v_j\|_1}{2}, \; v_i = \text{ LDA vector of white-paper } i \tag{3}$$

Where $\|\|_1$ is the L_1 norm. Note that the similarity goes from 0 to 1. We then design a visualization with the d3.js library using force graphs (Fig. 4). The graph represents each white-paper as a node and the each link represents their topic similarity. A force proportional to the similarity is then applied to each links grouping similar documents together. Finally each nodes have a color that stands for their respective topic. We selected a total of 5 topics by manual inspection and with the perplexity measure.

C. Data Features

One of the key parts in 1 is how to represent the input x for each ICO project. Due to the time limitation of the project and the difficulties of data collection, we can only craw two aspects:

- *Whitepaper:* is a document summarizing the operation of a start-up for potential investors. It is often public on the website of the ICO project
- *Website files:* is a collection of static files (e.g. HTML, JS, etc.) we crawl from the website of the ICO projects.

In the following sections, we will detail how we transform each of these aspects into a machine-readable vector in our deep learning model.

IV. White-paper model

A. Simple Models

This section explains the models used as baseline. Define $\vec{x}$ as the vector

1) Max-Class: The simplest model, always returns the most frequent class from training set.

2) Random: Randomly assign value from according to the distribution of the training classes.

3) Naive Bayes: We want to find the best class C_k according to: $\underset{C_k}{argmax}\ p(C_k|x_1,...,x_n)$. With Bayes rule the expression becomes: $\underset{C_k}{argmax}\quad p(C_k)\cdot p(x_1,...,x_n|C_k)$. Then with the following naive assumption : $p(x_i|C_k,x_1,...,x_{i-1},x_{i+1},...,x_n = p(x_i|C_k)$, the expression can be written as $\underset{C_k}{argmax}\ p(C_k)\cdot\prod_{i=1}^{n}p(x_i|C_k)$, where the denominator is removed since it has no impact in the maximization. Finally the distribution of the features is assumed to be gaussian: $p(x_i|C_k)=\frac{1}{\sqrt{2\pi\sigma^2_{C_k}}}exp(-\frac{(x_i-\mu^2_{C_k})}{2\sigma^2_{C_k}})$

4) Random Forest: We use 25 decision tree classifiers with maximum depth of 10. The number of tree and the max depth have been selected in order to minimize the variance.

5) Logistic Regression: Simple logistic regression model, the features are input to the model. No normalization nor constant is added.

6) MLP Logistic: Perceptron with two layer of size 40 and 20 and logistic (sigmoid) activation function.

7) MLP Relu: Perceptron with two layer of size 40 and 20 and Relu (rectified linear unit) activation function.

8) SVC: Support-vector machine with kernel : $\exp(-\gamma\|x-x'\|^2)$ where $\gamma=\frac{1}{\#features}$

B. Data Processing

Three methods have been compared to process the raw text:

1) Bag of word - Counts: With this approach, we assume that distribution of word in the document is enough to capture the success. We first tokenize each white-paper and lemmatize with the Wordnet Lemmatizer in order to reduce the dimensionality of the data. We also remove English stop words and count the number of occurrences in each word. Then we only keep words that appear in more than 5 and less than 80% of the documents.

2) Bag of word - TF-IDF: Using the vector generated in the previous subsection, we compute a new vector which also reduces the impact of term present in most white-papers. The formula is as follow:

$$\text{tf-idf}(d,t)=tf(d,t)*idf(d,t) \tag{4}$$

$$tf(d,t)=\frac{f_{t,d}}{\sum_{t'\in d}f_{t',d}} \tag{5}$$

$$idf(d,t)=log\frac{|D|}{|\{d\in D:t\in d\}|} \tag{6}$$

Where $f_{t,d}$ is the number of times term t appears in document d. D is the full collection of white-papers.

3) Word sequence: In this method, we consider the order of the words as well as their frequency. We first build a vocabulary with a unique index for every words. The text is then encoded using these indices in a sequence. This sequence is padded in order to have the same length for each documents.

C. Attention network

To prepare the data for this model, we tried two methods which produce similar results:

1) Sent2Vec [5]: We split each documents by sentences using the PunktSentenceTokenizer from NLTK. It splits text by sentences using the punctuation with a unsupervised machine learning algorithm. Then the sent2vec model with Wikipedia's uni-gram is used to encode the sentences with a dense vector.

2) Word2Vec [6]: We first download the word2vec model from Google. We can use it to generate an embedding of each words in the white-papers. Finally we can simple feed the sequence of vectors to the network for the classification. Note that any word that is not present in the model vocabulary is deleted.

Let us now introduce the attention network model. First we define the attention layer:

$$\vec{\alpha}=softmax((\vec{input}\cdot W)+\vec{bias}) \tag{7}$$

Note that α can be interpreted as a probability distribution since:

$$\sum_{i=1}^{n}\alpha_i=1 \tag{8}$$

$$\vec{output}=\sum_{i=1}^{n}\alpha_i*input_i \tag{9}$$

Since we are working with word sequences, we decided to work with a recurrent layer, we choose LSTM cells defined with the diagram in 5. The input x is fed to an array of 15 bi-directional LSTM cells and it outputs the full sequence of predictions $h_t,\ t\in[0,1000]$. Then the attention layer is applied before a second layer of 15 bi-directional LSTM cells that output the last evaluation of the sequence h_{1000}. Finally a fully connected layers transforms the output to the required format for classification. We will minimize the categorical cross-entropy loss function to train this model.

V. Website model

A. Graph neural network

In this project, we try to use the website's structure as a success predictor. For each website, we create a adjacency matrix of the websites pages. Next, we extract the topic of the pages using Latent Dirichlet Analysis with 20 topics. We end up with an adjacency matrix $A_i\in\{0,1\}^{N\times N}$ and a feature matrix $X_i\in\mathbf{R}^{N\times F}$ for website i where N is the number of pages in the website and F is the number of topics (20). The model uses multiple Graph Neural Networks defined as follow:

$$\hat{A}=A+I \tag{10}$$

$$f(H^{(l+1)},A)=Relu(\hat{D}^{\frac{1}{2}}\hat{A}\hat{D}^{\frac{1}{2}}H^{(l)}W^{(l)}) \tag{11}$$

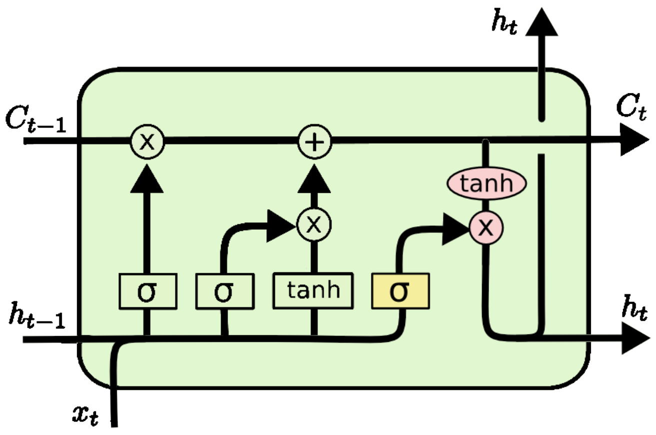


Fig. 5. LSTM cell.

Where $\hat{D}$ is the normalizing matrix defined as:

$$\hat{D}_{ii} = \sum_j \hat{A}_{ij} \tag{12}$$

The full model stack is given in the equations below:

$$H^{(0)} = X_i \tag{13}$$

$$W^{(0)} \in \mathbf{R}^{20\times 10},\ W^{(1)} \in \mathbf{R}^{10\times 20} \tag{14}$$

$$W^{(2)} \in \mathbf{R}^{20\times 8},\ W^{(3)} \in \mathbf{R}^{8\times 1} \tag{15}$$

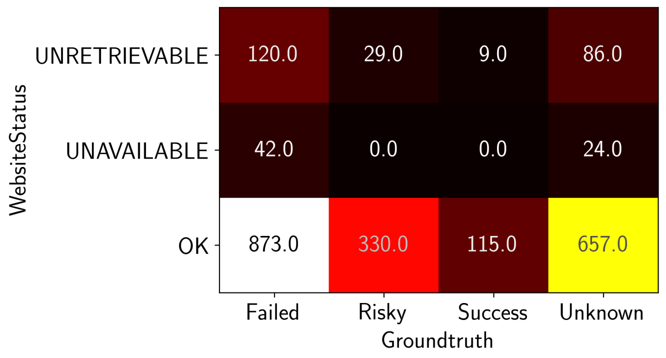
The output of the fourth layer is then flattened and the maximum value of order 6 are kept. Those 6 values are feed in a fully connected layer with softmax activation function to predict the class. The network is then trained with PyTorch and minimize the categorical cross-entropy loss function.

VI. Experimental Results

A. Setup

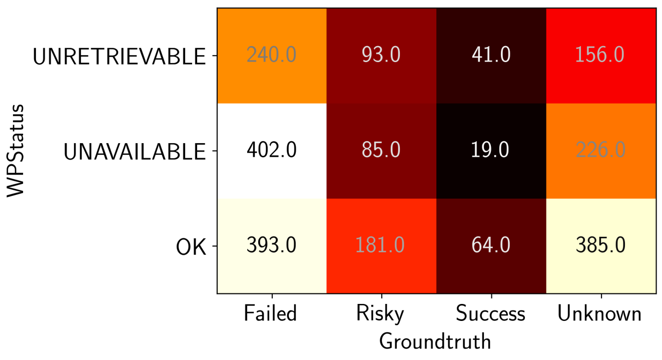
1) Dataset: We collect data from 2285 cryptocurrencies where 1035 failed, 359 are very volatile, 124 succeeded and the remaining 767 are too young to be classified.

From these ICOs, we were able to retrieve 1975 website 6 and 1023 white-papers 7. The ICOs span from the 28Th April 2013 (Bitcoin) to 18Th December 2018 (genaro-network).



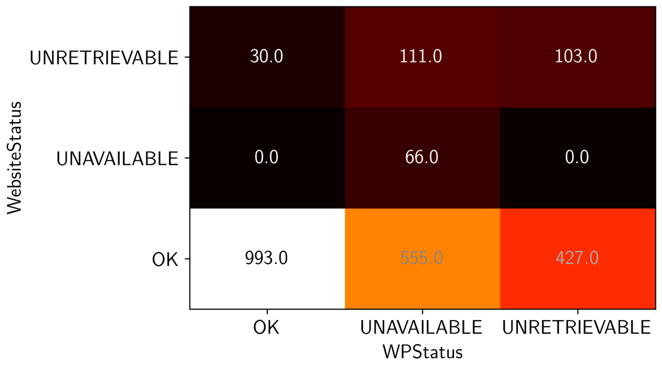
WebsiteStatus \ Groundtruth	Failed	Risky	Success	Unknown
UNRETRIEVABLE	120.0	29.0	9.0	86.0
UNAVAILABLE	42.0	0.0	0.0	24.0
OK	873.0	330.0	115.0	657.0

Fig. 6. Histogram of ground-truth to website status.



WPStatus \ Groundtruth	Failed	Risky	Success	Unknown
UNRETRIEVABLE	240.0	93.0	41.0	156.0
UNAVAILABLE	402.0	85.0	19.0	226.0
OK	393.0	181.0	64.0	385.0

Fig. 7. Histogram of the ground-truth to white-paper status.



WebsiteStatus \ WPStatus	OK	UNAVAILABLE	UNRETRIEVABLE
UNRETRIEVABLE	30.0	111.0	103.0
UNAVAILABLE	0.0	66.0	0.0
OK	993.0	555.0	427.0

Fig. 8. Histogram of website to white-paper status.

2) Metrics: For the first ground-truth, the metric used is the F_1-score computed for each class as follow:

$$F_1^C = \frac{2 \cdot precision^C \cdot recall^C}{precision^C + recall^C},\ \text{C} \in \{\text{Failed, Risky, Success}\} \tag{16}$$

$$F_1 = \sum_{\text{C}\in\{\text{Failed, Risky, Success}\}} Support(C) \cdot F_1^C \tag{17}$$

$$Support(C) = \frac{1}{|y|} \sum_{i=1}^{|y|} \mathbb{I}\{y_i = C\} \tag{18}$$

The precision and recall averages are computed in the same way as the F_1 averages. All values have been computed on 4-fold train-test cross-validation set. Where the data set is split in 4 parts. Three parts are used for training and the other one for testing. This technique increase the robustness of the metrics, however it takes 4 times as much training/testing time.

B. Evaluations on White-paper Models

The scores for the white-papers models have been reported in the following tables: I, II III IV.

Let us analyze the results of our Attention network models. First, take a look at the training/validation loss (Figure 9) and accuracy (Figure 10) of the Attention network model. We notice that the training loss is decreasing as expected. However, the model does not seem to be adequate for the data

TABLE I
RESULTS OF BAG OF WORD - COUNTS

Model	precision	recall	F_1-score
Max-Class	0.47	0.68	0.55
Random	0.54	0.53	0.54
Naive Bayes	0.48	0.25	0.29
Random Forest	0.49	0.68	0.55
Logistic Regression	0.57	0.66	0.59
MLP Logistic	0.54	0.66	0.57
MLP Relu	0.55	0.65	0.58
SVC	0.46	0.68	0.55

TABLE II
RESULTS OF BAG OF WORD - TF-IDF

Model	precision	recall	F_1-score
Max-Class	0.46	0.68	0.55
Random	0.52	0.52	0.52
Naive Bayes	0.47	0.24	0.28
Random Forest	0.53	0.68	0.56
Logistic Regression	0.46	0.68	0.55
MLP Logistic	0.55	0.66	0.57
MLP Relu	0.56	0.67	0.58
SVC	0.47	0.68	0.55

TABLE III
WORD SEQUENCE

Model	precision	recall	F_1-score
Max-Class	0.46	0.68	0.55
Random	0.53	0.53	0.53
Naive Bayes	0.57	0.57	0.56
Random Forest	0.55	0.68	0.56
Logistic Regression	0.55	0.63	0.58
MLP Logistic	0.47	0.68	0.55
MLP Relu	0.55	0.62	0.57
SVC	0.46	0.68	0.55

TABLE IV
COMPLEX WHITE-PAPER MODELS

Model	precision	recall	F_1-score
Attention network (sent2vec)	0.54	0.53	0.54
Attention network (word2vec)	0.51	0.61	0.55

since the validation loss is increasing after the first few epochs. Even with a very low number of parameters, regularization techniques and dropouts, we were unable to create a model expressive enough that does fit the data adequately. Either the model always outputs the most prominent class (in our case Failed) or it would perform well on the training set but very poorly on the validation set.

C. *Evaluations on Website Models*

The website model result is reported on the table V. Note that we use the same data-set on both white-paper and website model. Their result are comparable.

TABLE V
COMPLEX WEBSITE MODEL

Model	precision	recall	F_1-score
GNN network	0.61	0.69	0.59

Unfortunately, this representation of the data is not distinct enough. 46% of the websites in our data-set contain a single page, the topics distribution in those cases is not enough to classify an ICO. This model does not seem to pick up on any sufficiently significant correlation between success and topic in the Websites pages. However if we look at the details of the classification, it is able to precisely classify Risky ICOs, albeit with low recall. We could use the full text on the website instead of the topic to be more expressive. This would require to a much larger number of parameter and most likely lead to over-fitting.

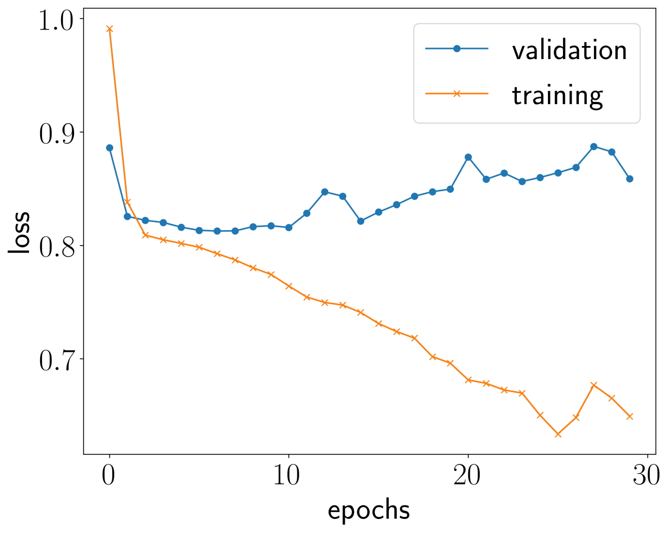


Fig. 9. Training/validation loss of the attention network

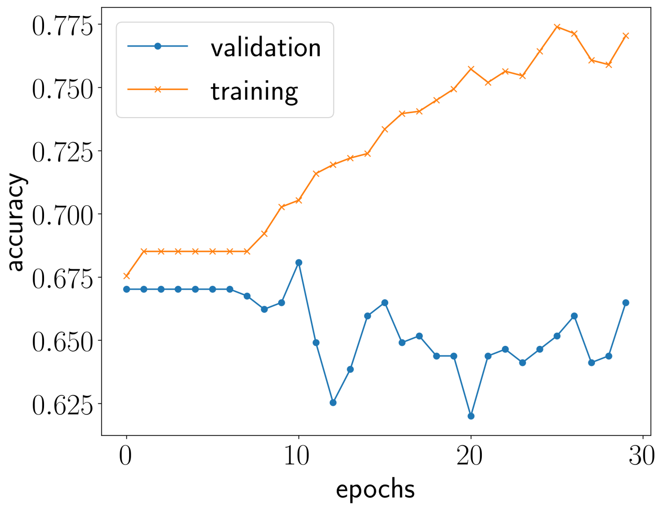


Fig. 10. Training/validation accuracy of the attention network

VII. RELATED WORK

A. *Success Prediction*

Predicting the Success of Kickstarter Campaigns [7]: Their paper proposes models with time series. This approach allows them to refine the prediction over time. This technique could be used for a new model, we already have collected multiple social media information (such as Facebook, Twitter, Telegram and YouTube accounts). The drawback of this technique is that the majority of these feature require multiple samples from the campaign launch period. Hence, building such a Data set takes a lot of time. Note that as their paper states, they predict the success of Kickstarter campaign. A campaign that raises the predefined amount of money is considered successful. This definition differs from ours has it does not detect potential scam.

IcoRating: A Deep-Learning System for Scam ICO Identification[4]: We already mentioned this work in Ground-truth section. At first we attempted to recreate the same Data set that they used, but we were unable to reproduce their Ground-truth distribution. They use a skip-thought model to generate sentence embedding on their white-papers and website text. We tested the skip-thought model but it did not perform better than the sent2vec model in our classification.

B. Embedding Models

While our current website's model uses the text displayed on the browser, we could also incorporate the information coming from the JavaScript code. The code could have some correlation with success as it could predict the potential of the programming team. However, I believe this would be marginal since multiple websites obfuscate their code. Moreover, the team working on the actual project could be unrelated to the team working on the website. Instead of working with the website code, I think that a better predictor of success would be the project's code available on git-hub. To generate such embedding of the project code, we could use the model described on the code2vec paper [8]. The model structures to code with an Abstract Syntax Tree (AST) and learns the embedding by predicting method's name from the method's body. We could use this technique to investigate other predictor and use bagging to improve our model.

C. Further Complex Models

We could improve the current results by applying the ideas of joint fusion and joint learning models from various domains such as data integration [9], [10], [11], [12], [13], human computation [14], [14], [15], [16], [17], [18], streaming query processing [19], recommendation systems [20], web credibility [21], data exploration [22], information retrieval [23], sensor data [24], and financial time series [25].

VIII. Conclusions and Future Work

ICOs classification is an hard task. The data collected on multiple sources is not reliable. Even the ICOs sale prices have a large variability between two sources. This leads to an unstable system where the definition of success cannot be asserted with confidence. In this project, we kept these uncertainty in mind while designing our definitions but cannot circumvent the core of the problem. On top of this problem, the ICO market is not time invariant, and the deep learning approach we took, does not consider the publication order of ICOs.

Using deep learning on ICO classification is an interesting approach. However currently the unreliability of the data coupled with the it's low volume are not the best ecosystem to train and test deep learning models. In fact, their usually high number of parameter is currently unable to capture the underlying function without over-fitting it. As reported in the simple models results, a well designed logistic regression looks more promising. In any case, I believe that our data set requires additional manual data cleaning to properly train deep learning architectures.

Multiple improvements can be made throughout the project. The top priority would be to expand the current data-set, the precision can be increased with more prepossessing. Translating the white-papers that are not in English, using a part-of-speech tagger to have more insight on the words and grouping together composite words such as ”real estate” would greatly increase the overall performance of every models.

The tool we made to scrape the ICOs also collects social media information. This data could be used to create new models. One idea would be to perform sentiment analysis on the messages exchanged on Twitter, Facebook or Telegram. Since a large part of the success is due to advertising, considering how much discussion and the relative positivity of the coin would likely help the classification. Since the current models don't perform very well, we did not attempt to create a final model using a combination of the two. However, where they to perform well enough, the models could be bagged together. This could be achieved by either voting (i.e. selecting the class that the majority of models would pick) or averaging the prediction of every models to select the best prediction (this takes into account the discriminating power of neural networks).

References

[1] U. Chohan, “Initial coin offerings (icos): Risks, regulation, and accountability,” *Discussion Paper Series: Notes on the 21st Century*, 2017.

[2] ICORATING, “Ico market research q2 2018,” Tech. Rep., 2018. [Online]. Available: https://icorating.com/pdf/44/1/TQFiiZ7n2pBdJeAy49lj9z5zBFSCUZWgM8h18riB.pdf

[3] ICORATING, “Ico market research q3 2018,” Tech. Rep., 2018. [Online]. Available: https://icorating.com/pdf/74/1/8TuF3swJWq8J82o4CnGI6dyEn3GQYhw9qdKTAE7U.pdf

[4] S. Bian, Z. Deng, F. Li, W. Monroe, P. Shi, Z. Sun, W. Wu, S. Wang, W. Y. Wang, A. Yuan *et al.*, “Icorating: A deep-learning system for scam ico identification,” *arXiv preprint arXiv:1803.03670*, 2018.

[5] M. J. Matteo Pagliardini, Prakhar Gupta, “Unsupervised learning of sentence embeddings using compositional n-gram features,” *arXiv preprint arXiv:1703.02507*, 2017.

[6] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[7] V. Etter, M. Grossglauser, and P. Thiran, “Launch hard or go home!: predicting the success of kickstarter campaigns,” in *Proceedings of the first ACM conference on Online social networks*. ACM, 2013, pp. 177–182.

[8] U. Alon, M. Zilberstein, O. Levy, and E. Yahav, “code2vec: Learning distributed representations of code,” *arXiv preprint arXiv:1803.09473*, 2018.

[9] N. Q. V. Hung, N. T. Tam, Z. Miklós, K. Aberer, A. Gal, and M. Weidlich, “Pay-as-you-go reconciliation in schema matching networks,” in *ICDE*, 2014, pp. 220–231.

[10] N. T. Tam, N. Q. V. Hung, M. Weidlich, and K. Aberer, “Result selection and summarization for web table search,” in *ICDE*, 2015, pp. 231–242.

[11] N. Q. V. Hung, X. H. Luong, Z. Miklós, T. T. Quan, and K. Aberer, “Collaborative schema matching reconciliation,” in *CoopIS*, 2013, pp. 222–240.

[12] N. Q. V. Hung, T. K. Wijaya, Z. Miklos, K. Aberer, E. Levy, V. Shafran, A. Gal, and M. Weidlich, “Minimizing human effort in reconciling match networks,” in *ER*, 2013, pp. 212–226.

[13] A. Gal, T. Sagi, M. Weidlich, E. Levy, V. Shafran, Z. Miklós, and N. Q. V. Hung, "Making sense of top-k matchings: A unified match graph for schema matching," in *IIWeb*, 2012, p. 6.

[14] N. Q. V. Hung, N. T. Tam, N. T. Lam, and K. Aberer, "BATC: a benchmark for aggregation techniques in crowdsourcing," in *SIGIR*, 2013, pp. 1079–1080.

[15] N. Q. V. Hung, C. T. Duong, N. T. Tam, M. Weidlich, K. Aberer, H. Yin, and X. Zhou, "Argument discovery via crowdsourcing," *VLDB J.*, pp. 511–535, 2017.

[16] N. Q. V. Hung, H. H. Viet, N. T. Tam, M. Weidlich, H. Yin, and X. Zhou, "Computing crowd consensus with partial agreement," *TKDE*, pp. 1–14, 2018.

[17] N. Q. V. Hung, D. C. Thang, N. T. Tam, M. Weidlich, K. Aberer, H. Yin, and X. Zhou, "Answer validation for generic crowdsourcing tasks with minimal efforts," *VLDB J.*, pp. 855–880, 2017.

[18] N. Q. V. Hung, D. C. Thang, M. Weidlich, and K. Aberer, "Minimizing efforts in validating crowd answers," in *SIGMOD*, 2015, pp. 999–1014.

[19] N. T. Tam, M. Weidlich, D. C. Thang, H. Yin, and N. Q. V. Hung, "Retaining data from streams of social platforms with minimal regret," in *IJCAI*, 2017, pp. 2850–2856.

[20] H. Yin, L. Chen, W. Wang, X. Du, N. Q. V. Hung, and X. Zhou, "Mobi-sage: A sparse additive generative model for mobile app recommendation," in *ICDE*, 2017, pp. 75–78.

[21] T. T. Nguyen, T. C. Phan, Q. V. H. Nguyen, K. Aberer, and B. Stantic, "Maximal fusion of facts on the web with credibility guarantee," *Information Fusion*, vol. 48, pp. 55–66, 2019.

[22] N. Q. V. Hung, K. Zheng, M. Weidlich, B. Zheng, H. Yin, N. T. Tam, and B. Stantic, "What-if analysis with conflicting goals: Recommending data ranges for exploration," in *ICDE*, 2018, pp. 1–12.

[23] N. T. Toan, P. T. Cong, N. T. Tam, N. Q. V. Hung, and B. Stantic, "Diversifying group recommendation," *IEEE Access*, vol. 6, pp. 17 776–17 786, 2018.

[24] N. Q. V. Hung, H. Jeung, and K. Aberer, "An evaluation of model-based approaches to sensor data compression," *TKDE*, pp. 2434–2447, 2013.

[25] N. Q. V. Hung and D. T. Anh, "Combining sax and piecewise linear approximation to improve similarity search on financial time series," in *ISITC*, 2007, pp. 58–62.