STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY

BIA-660 Final Presentation

Content Analysis and Text Classification of ICOs

May 2021

Presented By:
Agamdeep Chopra
Arnold Yanga
Ilesh Sharda

Research Questions

RQ 1: How does white paper content affect the reliability and probability of success in an ICO?

RQ 2: Is it possible to create a metric to filter the “copycats” and “frauds” who capitalize on the success of reliable projects through imitation and name recognition

RQ 3: Is it possible to cluster ICOs based on whitepaper similarity and measure their reliability based on key ICO features available online?

RQ 4: How can we use the presence of standard vs. informative to mitigate information asymmetry between investors and ICO issuers?

Dataset

1. We use the **Token Offering Research Database** (TORD) developed by P.P. Momtaz.
2. TORD is the most comprehensive database for information on ICOs, IEOs, and STOs available online and is continuously updated.
3. This database, particularly the ICO section, is susceptible to survivorship bias and contains large volumes of missing data.
4. The emphasis of this project is centered on ICO White Paper text data
    1. After scraping the collected links, we are left with a corpus of around 950 documents
    2. Many of the PDF links listed in TORD either no longer exist or are encrypted/broken
    3. ICOBench, the most popular ICO-based API has been out of service for months

Methodology

We have used the textual data extracted from the white papers for the purpose of our project. We explore two procedures to understand the utility of white papers in assessing ICO quality

1. Assessment of Information Content:
    - Unsupervised Learning Models to cluster ICO white papers by "industry"
    - Calculate the average of the TF-IDF term vectors based on sets partitioned by ICO start date and industry cluster
    - Implement Multiple Linear Regression to extract descriptive measurements for "standard" and "informative" content

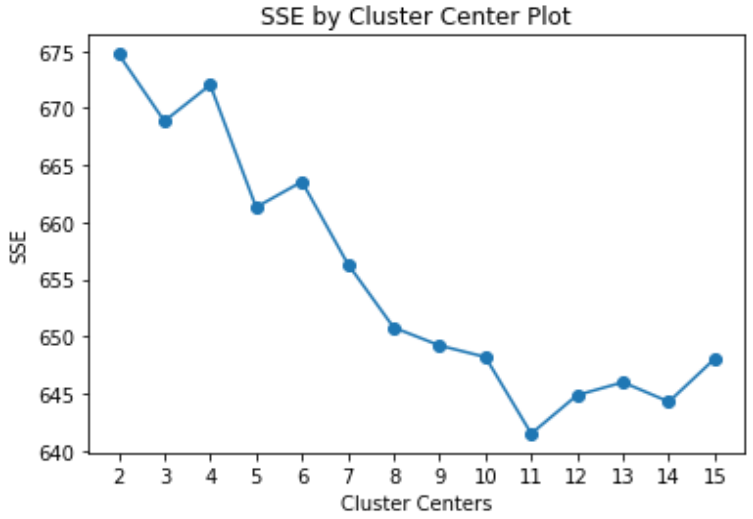
2. ICO Text Classification using Deep Learning Models
    - Performing text analysis of the data
    - Embedding the textual data in vector form
    - Building various machine learning models
    - Training and validating the vectorized data in those ML models.

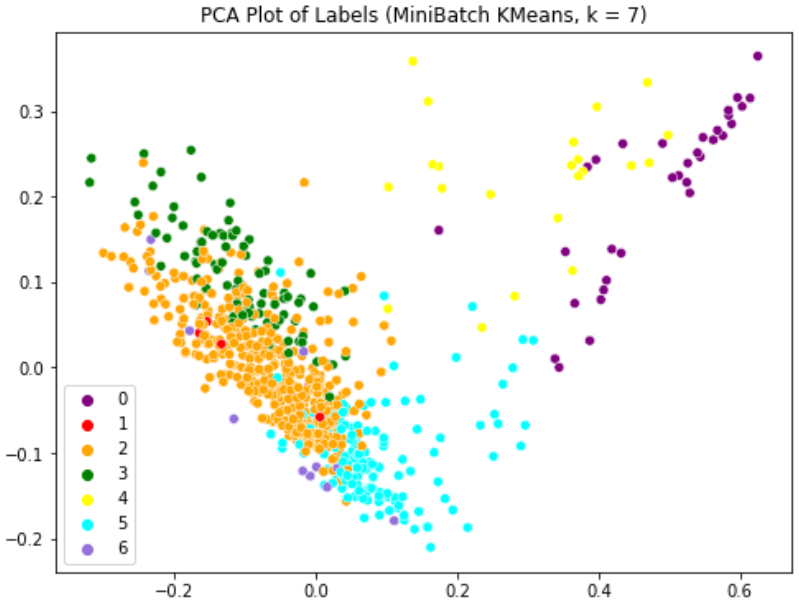
Standard v. Informative Content

1. The standard content of a white paper is characterized by the terms used by its industrial and temporal ICO counterparts
    - It is a measure of the degree to which ICOs borrow content from recent and industry white papers
2. Informative content describes the usage of terms that is acknowledged as a deviation from the norm
    - Can be indicative of either high or low quality ICO
3. Rapid growth and lack of regulation in the ICO Market leads us to believe that a proper metric to rate white paper content is necessary

Text Clustering by Industry

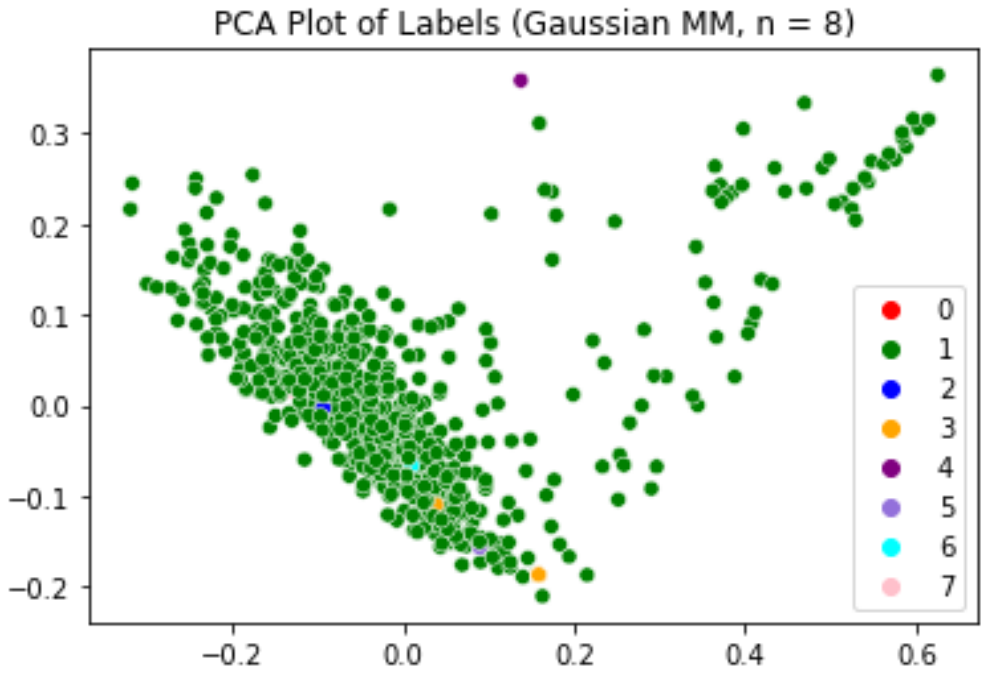
1. **Mini-Batch K Means Clustering:**
    - Cluster datapoints based on their proximity to the centroid
    - Mini-Batch K Means provides faster computation time at the expense of accuracy
2. Evaluation Methods:
    1. **Elbow Method**: (k = 7 was chosen)
    2. **Silhouette Score** ~ 0.01





Text Clustering (cont.)

1. **Gaussian Mixture Model:**
   - Assumes that each cluster follows a Gaussian Distribution
   - Uses the Expectation-Maximization (EM) to estimate the cluster for each sample
2. Grid Search to determine optimal number of clusters
   - Select the model with the lowest Bayesian Information Criterion (BIC) score (n = 8)
   - Based on the model analysis, Mini-Batch KMeans performs the best



Calculating Standard and Informative Content Scores

- Calculate average term usage in recent and industry white papers
- Use multiple regression with the norm vectors as the variables
- The sum of the coefficients are equal to the standard content
- The absolute sum of the residuals from the regression model is the informative content

$$norm_{rec,i} = \frac{1}{K}\sum_{k=1}^{K} norm_{tot,k}.$$

$$norm_{ind,i} = \frac{1}{P}\sum_{p=1}^{P} norm_{tot,p}.$$

$$norm_{tot,i} = \alpha_{rec,i}\, norm_{rec,i} + \alpha_{ind,i}\, norm_{ind,i} + \epsilon_i,$$

$$\alpha_{standard,i} = \alpha_{rec,i} + \alpha_{ind,i},$$

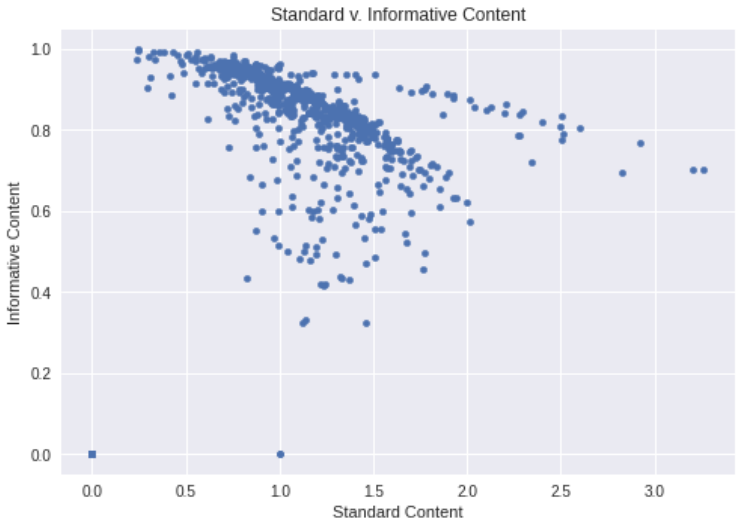
Results

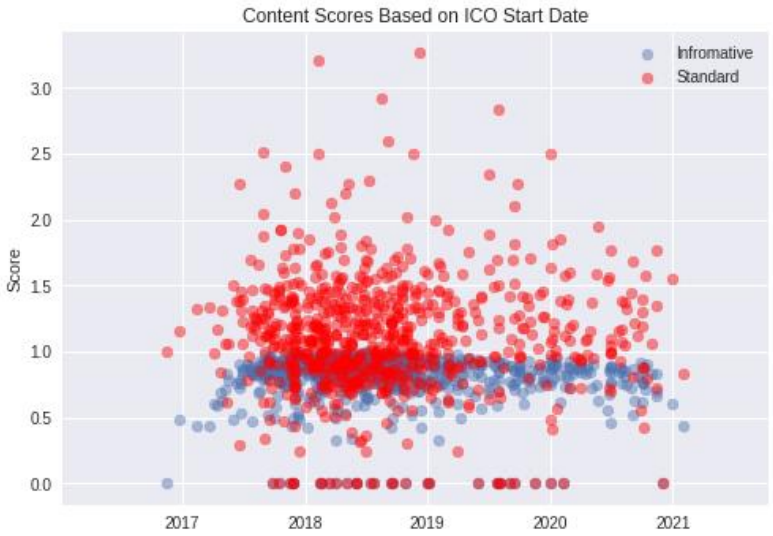
Standard Content:

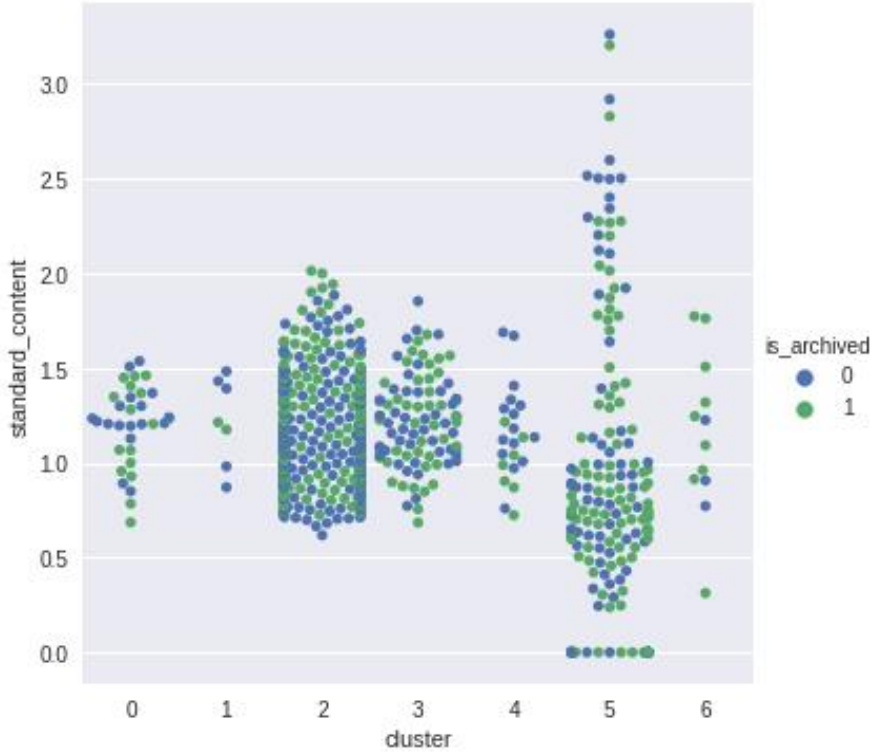
- Sample Mean: 1.1106
- Standard Deviation: 0.4402

Informative Content:

- Sample Mean: 0.8042
- Standard Deviation: 0.2002







Word Embedding

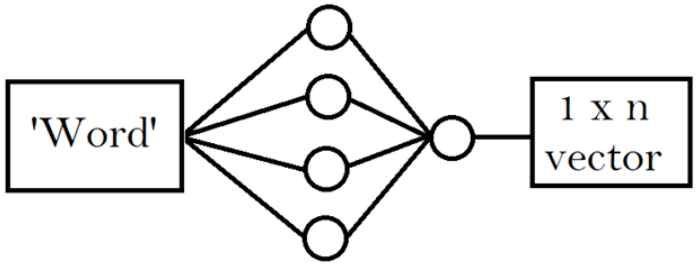
To train deep neural networks, we need a way to encode the words in the corpus in a representation that the ANN "understand" and can extract as much meaningful information from. We utilize PyTorch for creating a word embedding model.

The first step was to give each word some context for the model to learn. This was done by supplying each word with 2 words preceding it and 2 words in its succession as context. We shall refer to this tuple as the word-context ngram.

The next step was to assign each word a unique key/index to lookup its vector implementation after training the embedding model.

We created a NGramLanguageModular model as described in the PyTorch documentation to generate our word embeddings.

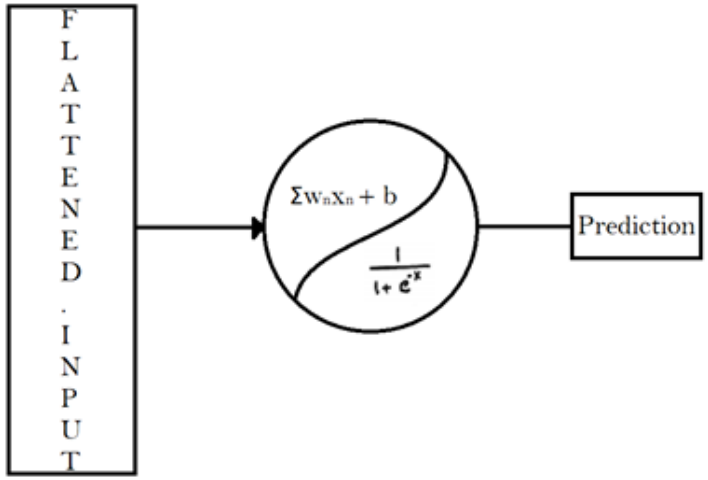
After training this model with the corpus subset, we created a few helper functions that translate the corpus information to our models in a readable vector format.



ML Models: Perceptron

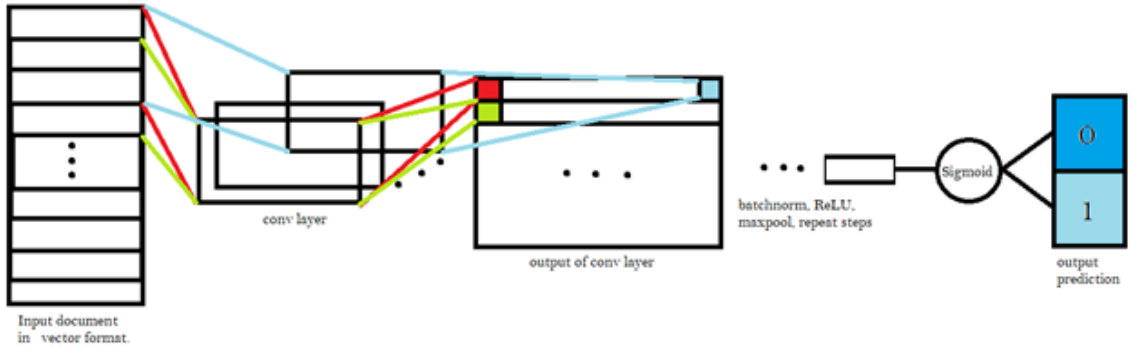
The perceptron is probably the simplest Feedforward neural network. It was set up as a benchmark model but gave decent results in the prototyping phase, so we decided to include it in this report as a benchmark to compare our model with.

Our Perceptron consists of a singular “neuron” that takes the input per document as a flattened 1d array and uses sigmoid nonlinearity to predict a classification for the document.



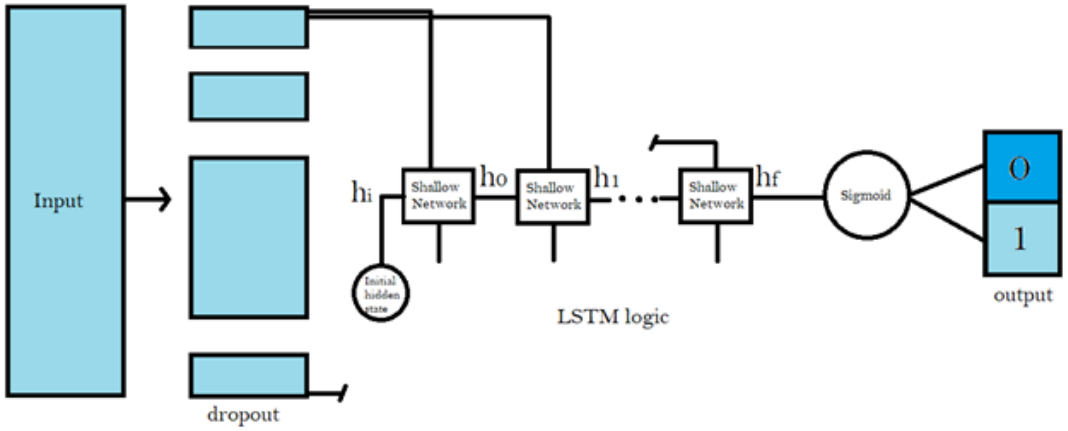
ML Model: CNN

The CNN (Convolutional Neural Network) model utilizes concepts from computer vison and applies them to the word vector document implementation. By using filters and windows, we can drastically reduce computational cost and parameter size while achieving similar or better performance as traditional dense neural nets. CNNs are a class of sparce networks since they share parameters per learnable window.

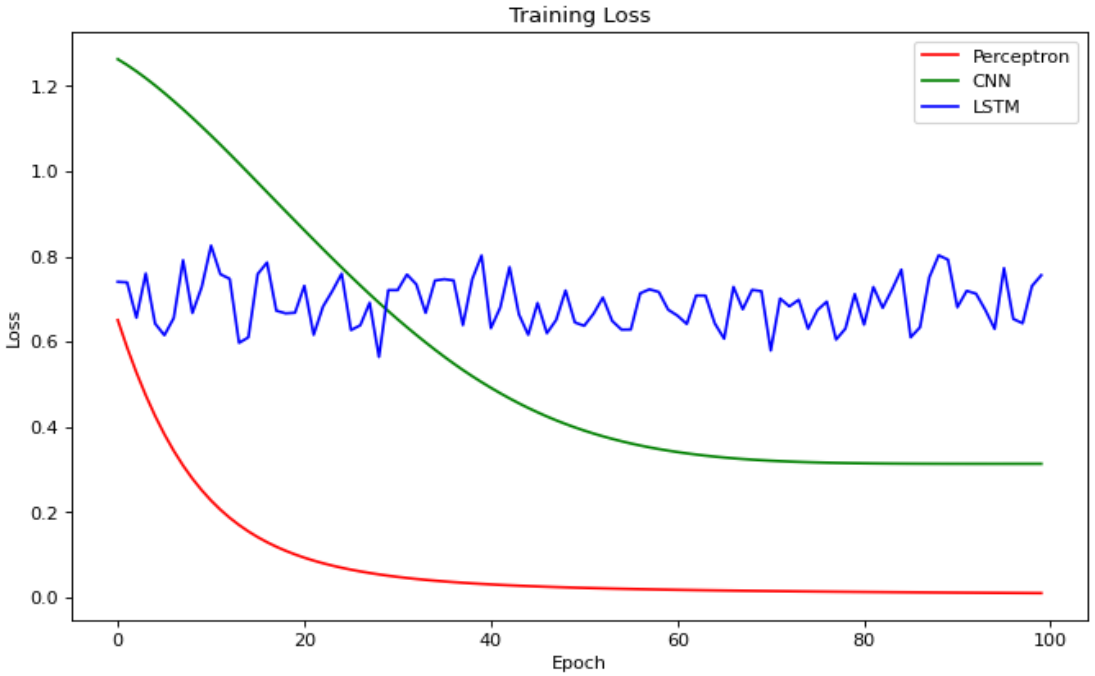


ML Model: LSTM

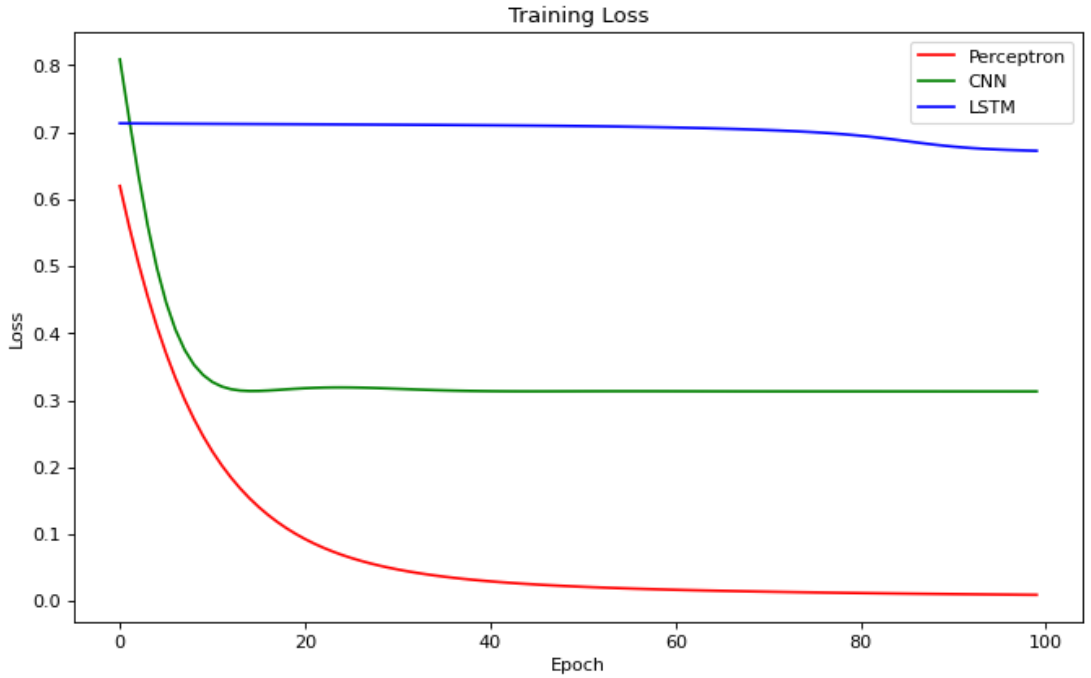
LSTM model implements PyTorch's provided LSTM class. Each LSTM module has 1 hidden layer with 3 nodes. Before passing the data to the LSTM layer during training, we drop 20% of the data randomly which translates to a probability of 0.8 for each input word to be passed through. This is known as dropout and in theory with large datasets, it should give better performance by reducing overfitting. The final hidden state of the LSTM model is passed to a linear to sigmoid sequence that outputs the predicted binary classification.



Model Results




```
Perceptron accuracy: 1.0
CNN accuracy: 1.0
LSTM accuracy: 1.0
```




```
Perceptron accuracy: 1.0
CNN accuracy: 1.0
LSTM accuracy: 0.5
```

Business Implications

- Using our project, a potential investor in upcoming ICO can analyze the white paper and predict whether the ICO is genuine or likely a scam. This can act as good safety net for the investor
- With limited data available online about Initial Coin Offerings, informative and standard content can be used as indicators for further ICO modelling and alleviate the issue of sparse data
- The development of text classification models for ICO white papers will help combat the issue of information asymmetry between issuers and investors
- As methods of ICO analysis are enhanced, people will feel more inclined to participate in the ICO Market

Limitations & Future scope

- Without resources like the ICOBench API we were unable to gain access to a sufficient sample size for features such as USD Raised, Tokens Sold, etc.
- For clustering purposes, the limited number of documents in our corpus may suggest that our data is imbalanced/not an accurate reflection of the ICO Market
- Test the effectiveness of standard and informative content as a means for ICO quality assurance

stevens.edu

Thank You