# Multivariate Time Series Error Detection

David Pham

September 12, 2018

## Contents

## 1 Project overview

In the financial investment industry, the speed and the quality of processing information is what determines if a business is sucessful or failure. In the recent period, an emphasis has been set on machine learning algorithms for investment strategies. One common application is the statement of the algorithms to perform classification of companies to put them into sector.

This project aims to create and review algorithms for classifying companies by using their share price. We will test a heuristic classifier using sample correlation and train a deep neural network to mimic the performance the sample correlation and study the hidden layer of the network. The projects uses data from Quandl and SimFin.

## 2 Problem Statement

The goal of this project is to create a method to embed financial time series into a vector space and analyze qualitatively the embedding. The main challenge to complete this projects are following.

1. Filter the universe of the data and acquire the data from different sources.
2. Transform data to be usable in machine learning algorithm.
3. Create artifical data to transform the problem into a supervised learning problem.
4. Create a base model and train an advanced classifier to create the embedding.
5. Analyze briefly the embedding.

## 3 Metrics

We will use the *accuracy* defined as the ratio between the true positive and negatives predictions divided by the dataset size. Or in formula

$$\text{accuracy} = \frac{\text{true positives} + \text{true negatives}}{\text{sample size}}$$

This metric is relevant because at a certain level the algorithms should be able to assign correctly the stock price of companies to the sector of the companies.

## 4 Data

The below table shows the data and how to access them.

| Type | Provider | Access to Datatest | Downloaded |
|---|---|---|---|
| Price | Quandl | Quandl API | 2018-08-21 |
| Company information | SimFin | S3 | 2018-08-21 |
| Stocks Returns | Processed | S3 | 2018-09-11 |
| Index Returns | Processed | S3 | 2018-09-11 |

In our analysis, we used the closed price of the US companies provided by Wikipedia and distributed by Quandl. As we additionally required the sectors from the company we needed to merge it with an additional databases provided by SimFin, which provides the taxonomy created by Global Industry Classification Standard (GICS). We could find the sectors of *726* companies which constituted our dataset. The prices range from 1962 to 2018.

## 5 Exploratory Analysis

## 6 Algorithms and Technique

There is two classifiers. One is algorithmic and the is other one is based on a Convolutional Neural Network (CNN).

The first classifier computes correlations of timeseries and then assign to an input timeseries the *sector* with which it has the highest correlation.

For two random variables $X$ and $Y$, with sufficient stability assumption, the Pearson correlation is defined as

$$\rho_p(X,Y) = E[XY] - E[X]E[Y] \approx \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i,$$

An interesting variant of the metric is the one applied on the rank of the $X$ and $Y$. This is called the Spearman's rho coefficient and is defined as $\rho_s(S,R)$, where $S$ and $R$ are

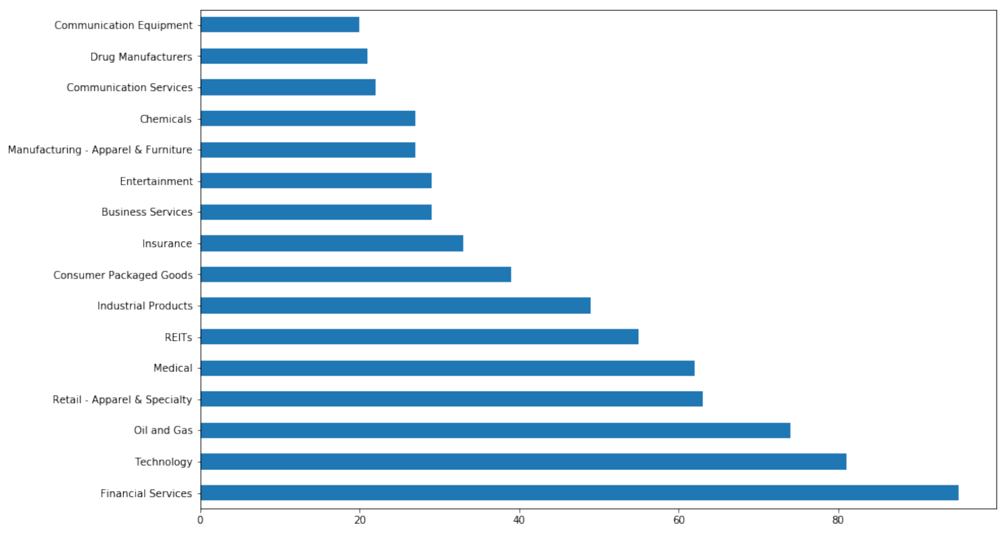

Figure 1: Distribution of sectors in the data.

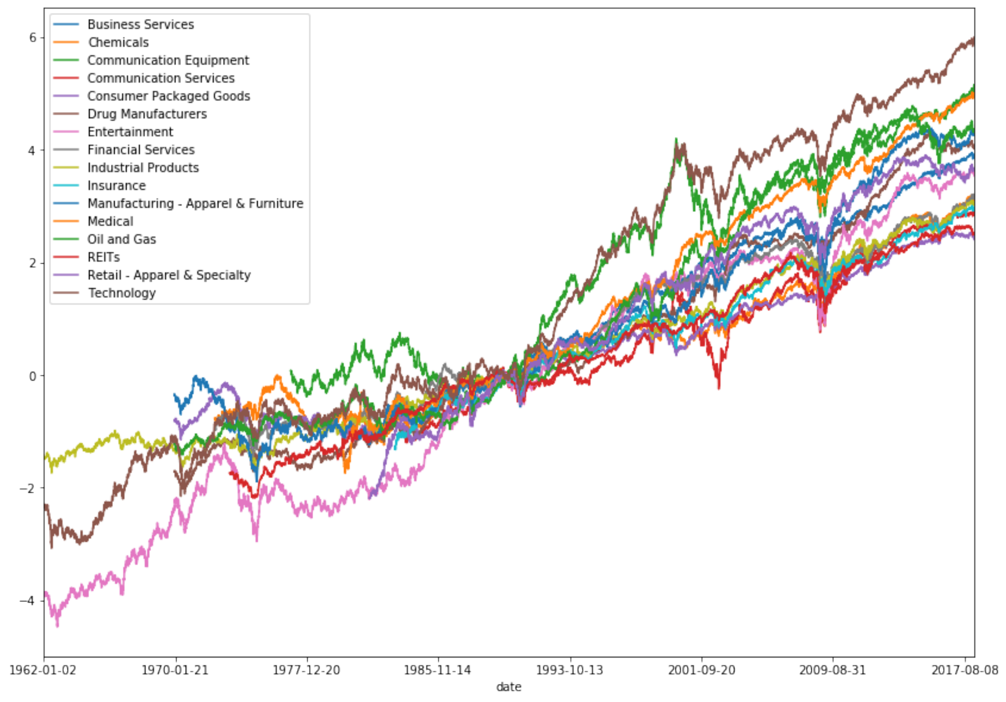

Figure 2: Synthetic indeces of sectors according to GICS. Index are set on 100 on the 1990-01-01.

the rank variable of the $X$ and $Y$. See here for more information. The advantage of this transformation is that it is invariant to monotonic transformation on $X$ and $Y$.

The second classifier use a deep neural network to create features and representation (or state) of the inputs to be able to separate linearly all the output sectors (or classes).

We train the neural network using *Adam*, which optimize the parameters of the network by minimizing the loss using stochastic gradient descent algorithm which standardizes the gradient and also updates it in iteration. The batch size was set to 32, the learning rate starts at 0.001 with a learning rate scheduler that diminish the learning by 10% every 10 epochs.

## 7 Methodology

### 7.1 Data preprocessing

From the Quandl dataset, the prepossessing involves keeping only the ticker and the close price for as many date as possible and as many companies as possible. Then the table is joined to the SimFin dataset containing sectors for :insert-number-of-companies:.

In total we have, from which we can extract data.

Then the sector indices were created by averaging the daily returns of the stocks within the sectors. The returns were floor and capped to 10% as it is unlikely that a indices of stocks lose or gain more than 5% in a single trading day.

### 7.2 Implementation

The implementation using Tensorflow and the keras API linked in the library. We launched AWS server with spot instance and launched a jupyter server there and made it accessible to our webbrowser. We develop code also in the terminal with emacs to adapt some code.

The first step was to download the data from the several providers and to process them. Then we needed to use several classes from keras to support asynchronous loading of the data thanks to the `Sequence` object.

The implementation has been performed with a simple function defining the network. We ran several experiment of the network, using adapation of inception units and residual units, known in the neural network for images, but they did not lead to any improvement of the model. Moreover, to avoid overfitting, we added several batch normalization layers as well as gaussian noise layer with a really small standard deviation. A few layers in the network were penalized $L_2$ regularization to insure that the features stayed as independent as possible.

### 7.3 Structure of the network

The network is depicted in Figure. It is a convoluat

## 8 Results

The base model using only correlation for the period of 3 months achieves 59*reallygood*.

As for neural network model, it achieves around 55% percent accuracy on a single observation of three months. However, we provide 25 random samples of 3 months period to the classifier, the classifier achieves 80% accuracy. As it can be read in Table 1.

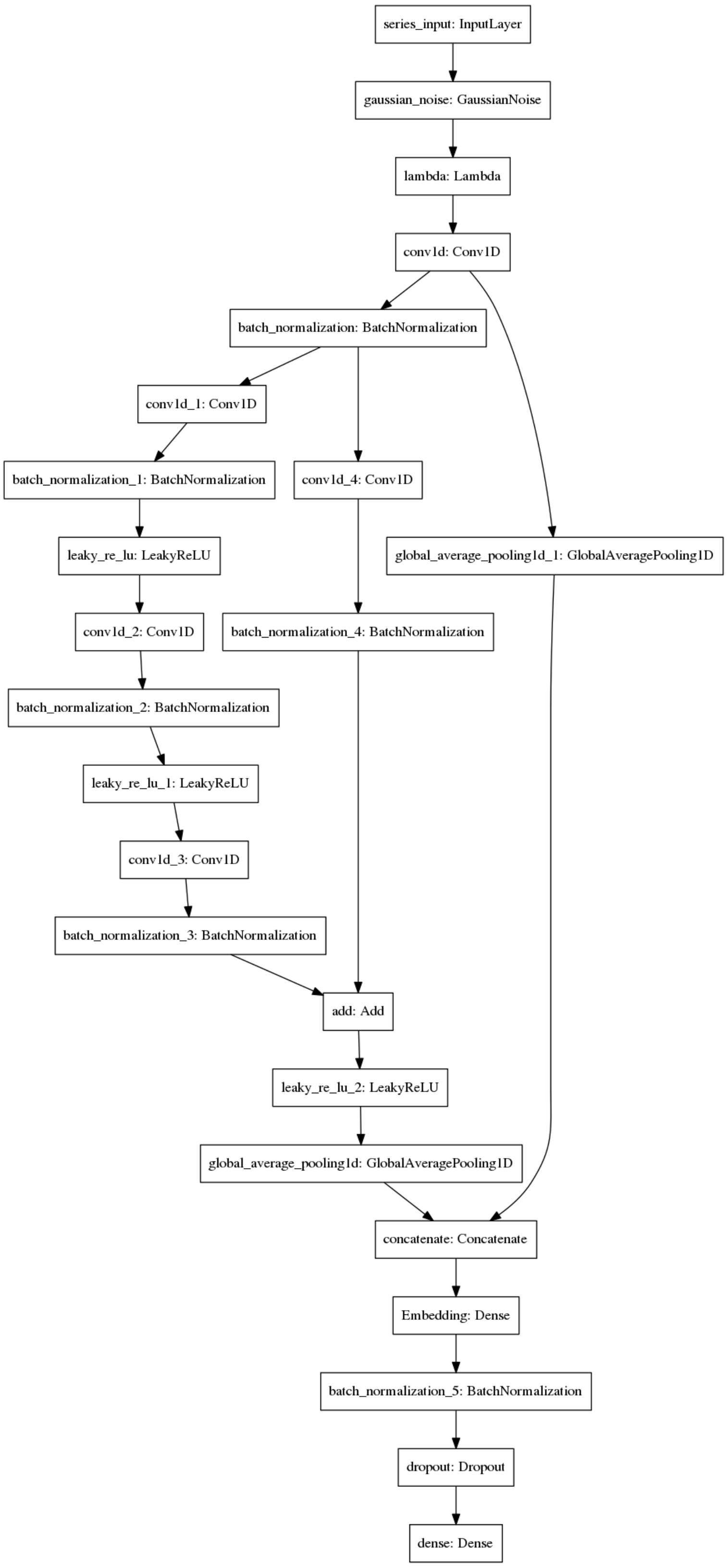
series_input: InputLayer
gaussian_noise: GaussianNoise
lambda: Lambda
conv1d: Conv1D
batch_normalization: BatchNormalization
conv1d_1: Conv1D
batch_normalization_1: BatchNormalization
conv1d_4: Conv1D
global_average_pooling1d_1: GlobalAveragePooling1D
leaky_re_lu: LeakyReLU
conv1d_2: Conv1D
batch_normalization_4: BatchNormalization
batch_normalization_2: BatchNormalization
leaky_re_lu_1: LeakyReLU
conv1d_3: Conv1D
batch_normalization_3: BatchNormalization
add: Add
leaky_re_lu_2: LeakyReLU
global_average_pooling1d: GlobalAveragePooling1D
concatenate: Concatenate
Embedding: Dense
batch_normalization_5: BatchNormalization
dropout: Dropout
dense: Dense

Figure 3: Neural Network structure

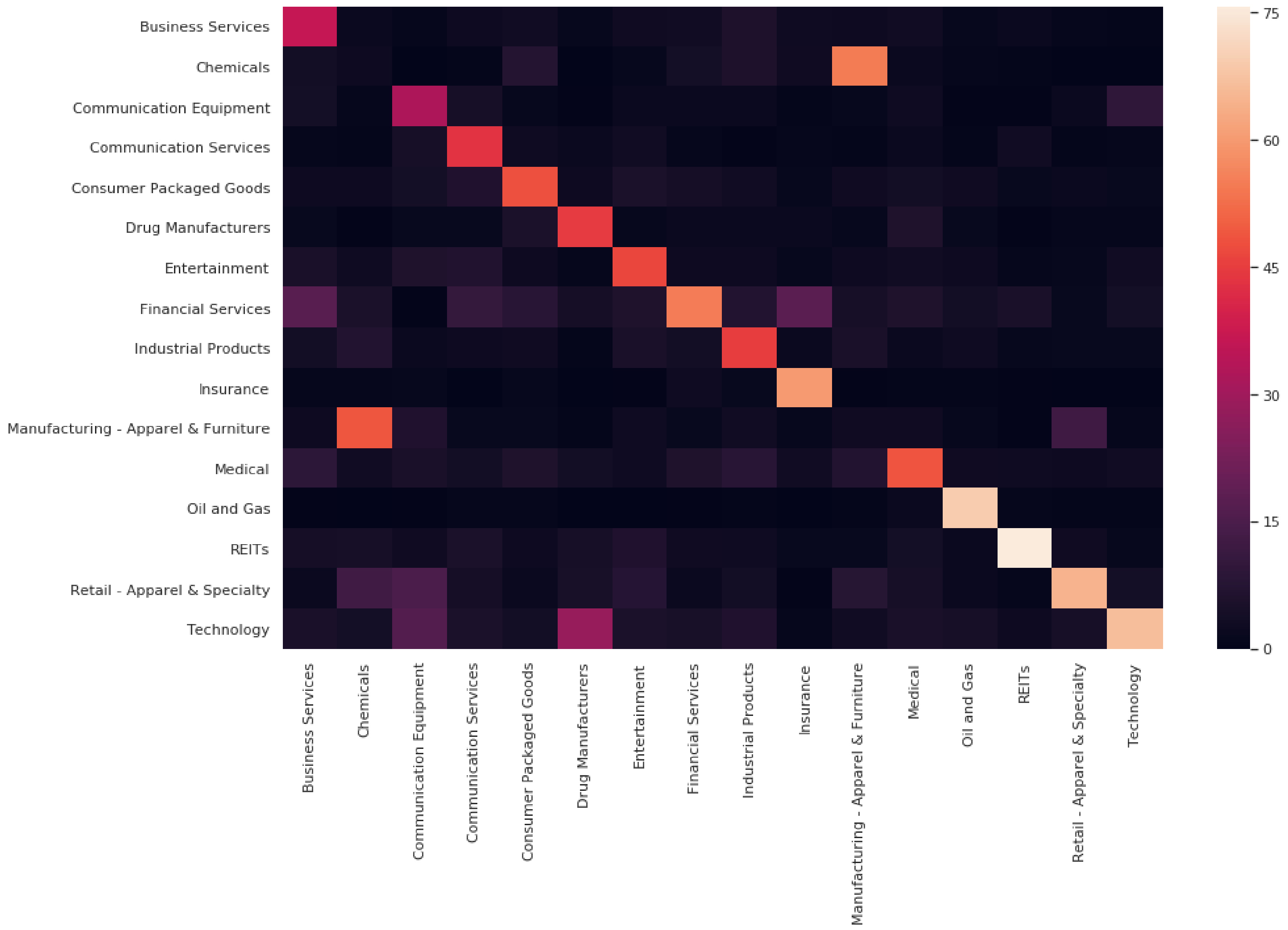

Figure 4: Confusion matrix of our predictor.

In Figure 8, we observe that the neural network model classifier does a fairly good job at classifying sectors with a notable exception of *Chemicals* and *Manufacturing - Apparels and Furniture*. The reason are probably that are little data.

Table 1: Confusion Report from the neural network classifier with resampled data.

| Sector | precision | recall | f1-score | support |
|---|---|---|---|---|
| Business Services | 1.00 | 1.00 | 1.00 | 3 |
| Chemicals | 0.00 | 0.00 | 0.00 | 3 |
| Communication Equipment | 0.00 | 0.00 | 0.00 | 2 |
| Communication Services | 1.00 | 0.50 | 0.67 | 2 |
| Consumer Packaged Goods | 0.60 | 0.75 | 0.67 | 4 |
| Drug Manufacturers | 0.67 | 1.00 | 0.80 | 2 |
| Entertainment | 1.00 | 1.00 | 1.00 | 3 |
| Financial Services | 0.90 | 0.90 | 0.90 | 10 |
| Industrial Products | 0.71 | 1.00 | 0.83 | 5 |
| Insurance | 1.00 | 1.00 | 1.00 | 3 |
| Manufacturing - Apparel & Furniture | 0.00 | 0.00 | 0.00 | 3 |
| Medical | 1.00 | 1.00 | 1.00 | 6 |
| Oil and Gas | 1.00 | 1.00 | 1.00 | 7 |
| REITs | 1.00 | 0.83 | 0.91 | 6 |
| Retail - Apparel & Specialty | 0.86 | 1.00 | 0.92 | 6 |
| Technology | 0.78 | 0.88 | 0.82 | 8 |
| avg / total | 0.79 | 0.82 | 0.80 | 73 |

## 9 Embedding

We are curious to look at the embedding produce by our neural network. We use t-SNE to create a low dimensional representation of it. This technique preserves the similarity between points.

In order to create an estimate of the embedding, we sampled the 25 periods of 3 months of each stock and averaged their embedding.

In Figure 9, it can be observed that stocks from the same sector tends to be near each other. The distinct cluster are finance and technology, which are also the most represented in our dataset. The model seems to have difficulty to differentiate somes chemical companies as their embedding seems to be close to some industrial production companies. In general, the more companies we had in the raw dataset the more precise the groups are.

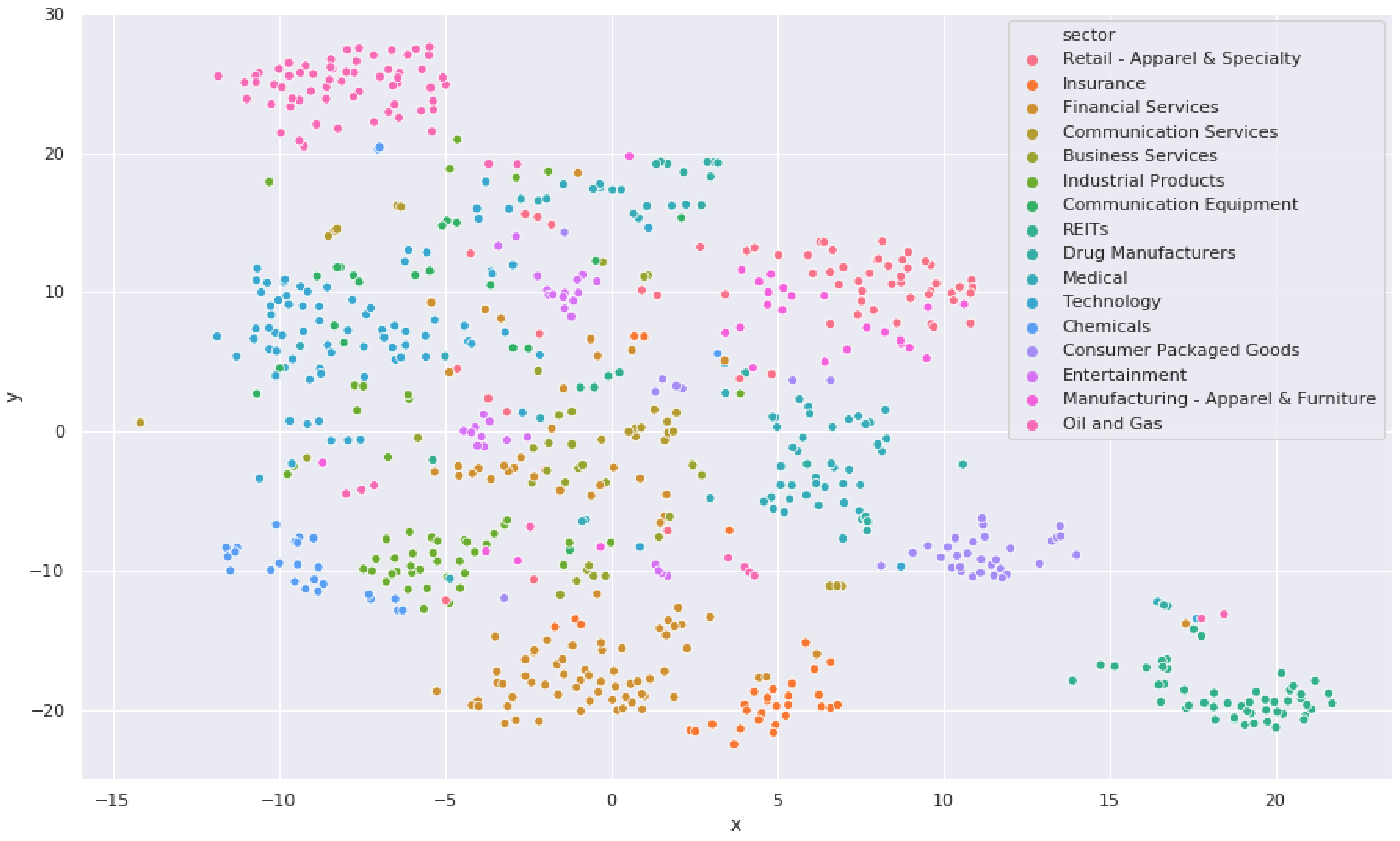

Figure 5: T-SNE of the embedding layers of the network.

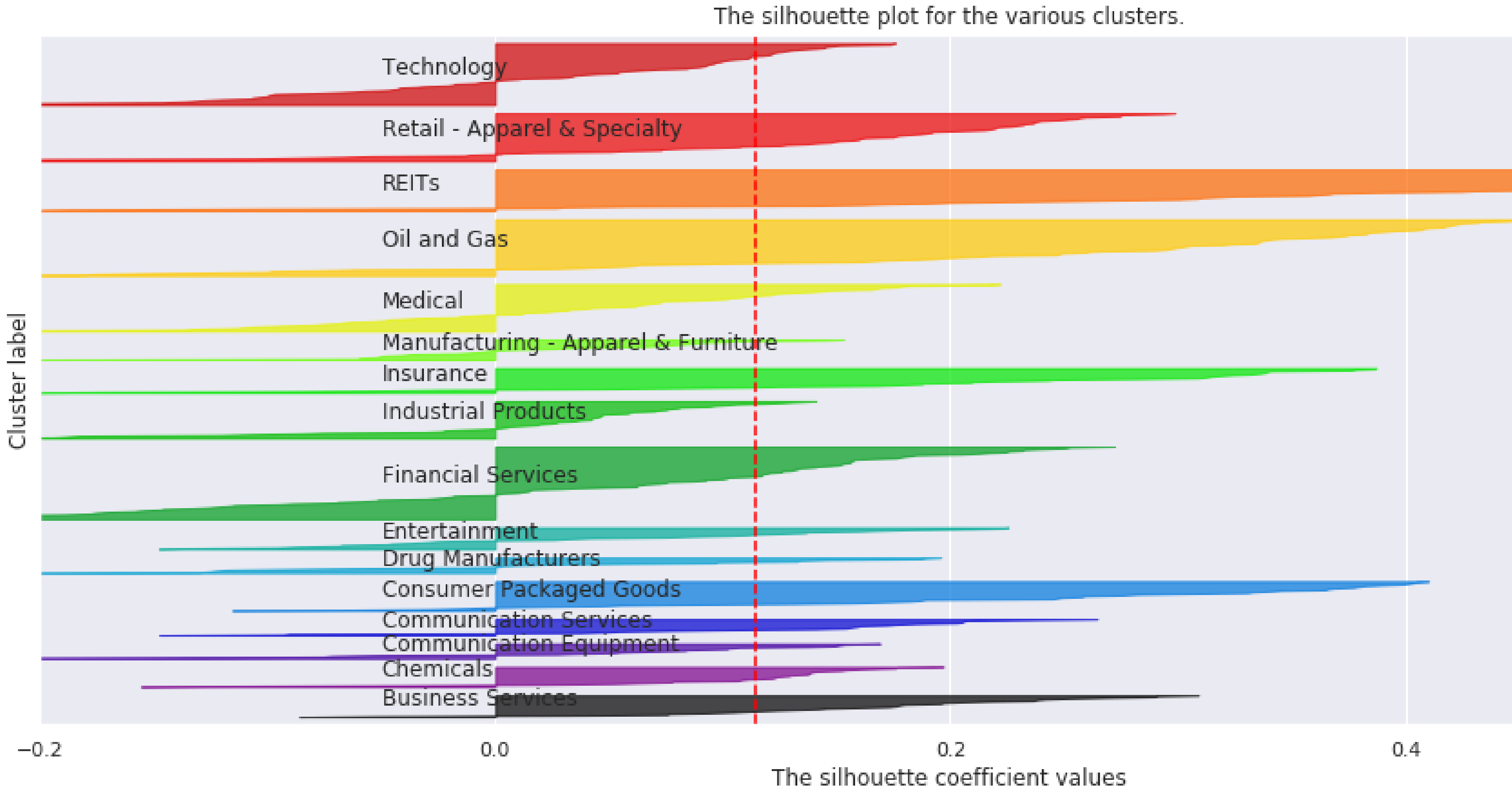

Figure 6: Silouhette score of the average embedding of the stocks

In Figure 9, the silouhette score is depicted for the several sectors in our dataset. The first observation is we see the advantage of a neural network classifier over k-means clustering, because the silhouette score overweights the mislabeled sample of financial and technology, which are the most precise sectors. Otherwise we can observe that REiTS, Oil and Gas, and the Insurance sector are grouped tightly making them quite distinct group.

## 10 Conclusion and Further applications

The first lesson I learned is data preparation and acquisition is much harder than thought and we should thank the machine learning community for providing so many labeled data set for our development. Indeed the financial industry still leverage on providing exclusive data and create a difficult task to leverage on alternative dataset, which could potentially provide added value.

Second, in deep learning, a bigger network does not necessarily translate into a better performance: training is much more difficult with more parameters even with regularizers and advanced optimization method.

However, these most so-called quantitative strategies require data in the most accurate form and event the most trusted data provider fails to deliver the data without flaws.

Inconvenient: for now it is impossible to detect new group in as we would need to train them. Zero shot learning would be an interesting project to the study.