

Janderson Borges do Nascimento

StockNet: A Multivariate Deep Neural Architecture for stock prices prediction

Manaus

2019

Janderson Borges do Nascimento

StockNet: A Multivariate Deep Neural Architecture for stock prices prediction

Proposta de dissertação apresentada ao Programa de Pós-graduação em Informática da Universidade Federal do Amazonas como requisito parcial para obtenção do grau de Mestre em Informática. Área de concentração: Banco de Dados e Recuperação da Informação.

PODER EXECUTIVO
MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO AMAZONAS
INSTITUTO DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

Orientador: Prof. Dr. Marco Antônio Pinheiro de Cristo

Manaus

2019

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

N244s Nascimento, Janderson Borges do
StockNet: A Multivariate Deep Neural Architecture for stock prices prediction / Janderson Borges do Nascimento. 2019
55 f.: il. color; 31 cm.

Orientador: Dr. Marco Antônio Pinheiro de Cristo
Dissertação (Mestrado em Informática) - Universidade Federal do Amazonas.

1. Stock Forecast. 2. Neural Networks. 3. Natural Language Processing. 4. Time-Series Prediction. I. Cristo, Dr. Marco Antônio Pinheiro de II. Universidade Federal do Amazonas III. Título



PODER EXECUTIVO
MINISTÉRIO DA EDUCAÇÃO
INSTITUTO DE COMPUTAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA



UFAM

FOLHA DE APROVAÇÃO

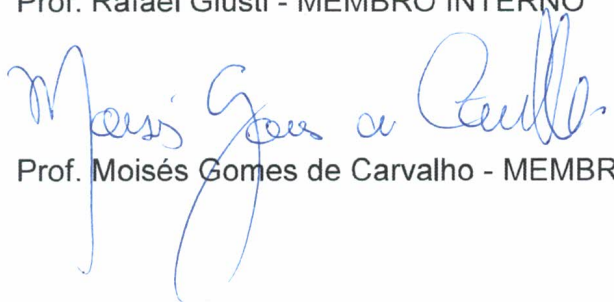
"StockNet: A Multivariate Deep Neural Architecture for stock prices prediction"

JANDERSON BORGES DO NASCIMENTO

Dissertação de Mestrado defendida e aprovada pela banca examinadora constituída pelos Professores:


Prof. Marco Antonio Pinheiro de Cristo - PRESIDENTE


Prof. Rafael Giusti - MEMBRO INTERNO


Prof. Moisés Gomes de Carvalho - MEMBRO EXTERNO

Manaus, 13 de Setembro de 2019

Summary

List of Figures	iii
List of Tables	iv
1 Introduction	4
1.1 Research Objective	6
1.1.1 Specific Goals	6
1.2 Research Contribution	6
1.3 Organization	6
2 Fundamentals	7
2.1 Discrete-time Time-series	7
2.2 Financial Time-series Prediction	8
2.3 Using news content as input	8
2.3.1 Input text as sequence	9
2.3.2 One-Hot Encoding	9
2.3.3 Bag of Words	10
2.3.4 Word Embeddings	10
2.3.4.1 Continuous Bag of Words	11
2.3.4.2 Skip-gram	11
2.3.5 Information Extraction	13
2.3.5.1 Open information extraction	13
2.3.6 Structured Events Embeddings	14
2.3.7 Sentiment Analysis	15
2.3.7.1 Sentiment Analysis of Financial Texts	16
2.4 Artificial Neural Networks	16
2.4.1 Activation Function	18
2.4.2 Loss Function	19
2.4.3 Backpropagation	20
2.4.4 Weight Initialization	21
2.4.5 Input Normalization	21
2.4.6 Recurrent Networks	22
2.4.7 1D Convolution	23
2.4.8 Causal Convolution	23
2.4.9 Residual Connections	23
2.5 Temporal Convolutional Networks	23
2.5.1 Kernel Dilatation	24
2.5.2 Wavelet Transform and 1D Convolution	24
2.5.3 Dilated Causal Convolutional Architecture	25

3	Related Work	28
3.1	Financial time-series prediction methods	28
3.2	Text based techniques	28
3.3	Deep Models for financial time-series prediction	29
3.4	SeriesNet - A novel and complex stock forecasting architecture	30
3.5	SEE - An older but robust model for stock prediction	31
4	StockNet: Multivariate Deep Neural Architecture for stock prices prediction	32
4.1	Integrating News in a Convolutional Temporal Model	32
4.1.1	Probabilistic View	32
4.1.2	Financial Semantic Field Ranking	33
4.1.3	High level architecture	34
4.1.4	Hyper-parameters	34
4.1.5	Model Objective	35
4.1.5.1	Price prediction and polarity forecasting	36
5	Experiments	37
5.1	Stock Prices Dataset	37
5.2	News Dataset	37
5.3	News Embeddings	38
5.4	Baseline	38
5.5	Experiment Setup	39
5.6	Methodology	39
5.7	Preliminary Experiment (S&P500)	39
5.7.1	Preliminary Results	40
5.8	StockNet Intraday Extensive Experiments	41
5.8.1	Temporal News Subsampling	42
5.8.2	Individual Stocks instead of Market Indexes	42
5.8.3	Final Results	43
6	Conclusions	47
6.1	Future Work	47
	References	49

List of Figures

Figure 1 – CBOW and Skip-gram architecture overview	12
Figure 2 – An illustration of Stanford Open IE’s approach. From left to right, a sentence yields a number of independent clauses. From top to bottom, each clause produces a set of entailed shorter utterances, and segments the ones which match an atomic pattern into a relational triple (ANGELI; PREMKUMAR; MANNING, 2015).	14
Figure 3 – Extracting events from news stream. (NASCIMENTO; CRISTO, 2015)	15
Figure 4 – Rosenblatt’s Perceptron Representation	17
Figure 5 – List of common activation functions.	18
Figure 6 – Illustration of (a) LSTM and (b) GRU memory cells. (a) i , f and o are the input, forget and output gates, respectively. c and \tilde{c} denote the memory cell and the new memory cell content. (b) r and z are the reset and update gates, and h and \tilde{h} are the activation and the candidate activation. (CHUNG et al., 2014)	22
Figure 7 – Dilated Causal Convolutional Neural Network Structure (OORD et al., 2016a)	26
Figure 8 – Residual block and WaveNet architecture (OORD et al., 2016a)	27
Figure 9 – SeriesNet Stock Forecasting Architecture (SHEN et al., 2019)	30
Figure 10 – Structured event embeddings (SEE) representation (NASCIMENTO; CRISTO, 2015)	31
Figure 11 – StockNet architecture Overview	35
Figure 12 – Preliminary Experiment GRU+SEE with GRU Recurrent Architecture	41
Figure 13 – Intraday news histogram with 30-minute time splitting	42
Figure 14 – Results for IBM	44
Figure 15 – Results for Cisco Systems	45
Figure 16 – Results for Verizon (not in initial dataset)	46

List of Tables

Table 1 – Top 10 more similar vectors in trained embeddings given two different queries.	38
Table 2 – StockNet results against baseline and alternative architectures for S&P500 (lower is better)	41
Table 3 – StockNet results against baseline (lower is better)	43

List of abbreviations and acronyms

EMH	Efficient Market Hypothesis
NLP	Natural Language Processing
DWT	Discrete Wavelet Transform
IE	Information Extraction
OpenIE	Open Information Extraction
BOW	Bag of Words
LSA	Latent Semantic Analysis
CBOW	Continuous Bag of Words
RNN	Recurrent Neural Network
LSTM	Long Short Term Memory
GRU	Gated Recurrent Unit
CNN	Convolutional Neural Network
TCN	Temporal Convolutional Network
SE	Structured Event
SEE	Structured Event Embedding
BERT	Bidirectional Encoder Representations from Transformers
S&P 500	Standard & Poor's 500 Stock index
MSE	Mean Squared Error
RMSE	Rooted Mean Squared Error
MAPE	Mean Absolute Percentage Error

Abstract

Stock price forecasting is an inherently difficult problem. According to the *efficient market hypothesis* financial prices are unpredictable. However, a great number of machine learning methods have obtained consistent results on anticipating market movements. Most recent time-series prediction methods attempt to predict prices polarity, that is, whether prices have increased or fallen compared to the last time-step. Such approaches are inefficient in real scenarios, as forecasting price polarity alone makes financial planning a hard task, due to the fees and operation costs. Most of these methods use only Recurrent Neural Networks, but recent advances in temporal convolutional networks also may prove to be promising in prediction of general time-series, making better predictions with easier to train models. Recent hybrid architectures have also obtained important results using additional unstructured information from financial news. We propose a novel deep neural architecture to predict stock prices based on Temporal Convolutional Networks and built upon on a state of the art acoustic model for voice synthesis. Experimental results show that our model can consistently improve individual stocks prediction when compared to traditional methods.

Keywords: Stocks Forecast; Neural Networks; Natural Language Processing; Time-Series Prediction

Resumo

A previsão dos preços de ações é um problema inerentemente difícil. De acordo com a *hipótese do mercado eficiente*, os preços financeiros são imprevisíveis. No entanto, muitos métodos de aprendizado de máquina têm obtido resultados consistentes na antecipação de movimentos de mercado. Modelos de previsão de séries temporais recentes têm tentado prever apenas a polaridade dos preços, ou seja, se eles subiram ou caíram em relação ao passo temporal anterior. Tal abordagem é ineficiente em cenários reais, pois dificulta o planejamento financeiro em virtude dos custos e taxas presentes em cada operação. A maioria desses métodos usa Redes Neurais Recorrentes, porém, avanços recentes em redes temporais convolutivas têm se mostrado promissores na previsão de séries temporais, possibilitando previsões melhores com modelos mais fáceis de treinar. Arquiteturas híbridas também têm obtido resultados importantes ao processar informações não estruturadas de notícias financeiras. Propomos neste estudo uma nova arquitetura neural profunda para previsão de preço de ações baseada em Redes Convolucionais Temporais e inspirada em um modelo acústico do estado da arte para síntese de voz. Resultados experimentais mostram que nosso modelo pode melhorar de forma consistente a previsão do preço de ações individuais quando comparado aos métodos tradicionais.

Palavras-chaves: Previsão de Ações; Redes Neurais; Processamento de Linguagem Natural; Previsão de Séries Temporais

Acknowledgment

I would like to thank God for giving me health and intelligence to overcome all difficulties and finish this mission.

I would also like to thank my thesis advisor Dr. Marco Cristo. The door to Prof. Dr. Cristo office was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this paper to be my own work, but steered me in the right the direction whenever he thought I needed it.

I also would like to thank Zhipeng Shen who kindly answered our questions and assisted us in implementing his method in our experiments phase.

Finally, I must express my very profound gratitude to my spouse and friends for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this master thesis. This accomplishment would not have been possible without them. Thank you.

Janderson Borges

1 Introduction

The automation of financial decisions has become popular in the stock market. Since BM&F Bovespa (the Brazilian stock exchange) started high-frequency trades¹ almost ten years ago, the number of high-frequency operations increased from 2.5% in 2009 to 36.5% in 2013 (ZHOU et al., 2018). This new scenario requires accurate and reliable forecasting systems to ensure some gain in operations.

In fact, financial price forecasting has been an important academic research field since long before the recent hype and the popularization of trading robots. However, there is no consensus about the feasibility to forecast the trading market. According to the strong version of the efficient market hypothesis (EMH) (FAMA, 1965), financial prices are mostly or completely unpredictable. However, the weak form of the Efficient Market Hypothesis (EMH) (ROBERTS, 1967) asserts that prices fully reflect only the information contained in the historical sequence of prices, thus, new public information can be explored by forecasters.

Other researchers have indeed studied factors which could have impacted on stock prices. For instance, BONDT; THALER has proposed that traders are subject to emotional overreactions that can make prices oscillate systematically. KAHNEMAN; TVERSKY has proposed that traders also can be overconfident in their ability to predict a future stock price. We believe that this kind of psychological bias, if exists, should be really difficult to be detected and handled by human analysts in traditional forecasting, but should be better exploited by machine learning methods.

Most of these recent methods do not try to predict exactly the future stock price, but rather its polarity. In other words, whether its movement in time t will be up or down relative to time $t - 1$. However, financial management is hampered by these models. Some investors, for instance, cannot estimate how much they will earn in a particular operation, but only if such an operation will yield profit or loss. Therefore, it is well known that small gains, when they do not exceed the cost of the operation, can result in losses.

Despite these difficulties, many advances have been reported in the last years on anticipating market movements using machine learning techniques, generally proposed to time series forecast. Most techniques explore the trend of people following rumors spread by News published, for instance, in web pages (GIDOFALVI; ELKAN, 2001; NASSIRTOUSSI et al., 2015) and in social media (OLIVEIRA; CORTEZ; AREAL, 2017; BOLLEN; MAO; ZENG, 2011; MITTAL; GOEL, 2012). Thus, the understanding of finance news content

¹ High-frequency trading (HFT) is a type of algorithmic trading characterized by high speeds, high turnover rates, and high order-to-trade ratios that leverages high-frequency financial data and electronic trading tools. (ALDRIDGE, 2010)

may provide useful information to forecast trader reactions and stock price movements.

Online news are valuable input sources for predictive models due to their high abundance and availability. In the context of stock forecasting they seem even more important, since traders tend to check these sources before their decisions. Several hybrid approaches have also emerged as an alternative to traditional stock time series forecasting, which use not only price and volume data as input features for the model, but also unstructured external information, such as interactions in social networks (BOLLEN; MAO; ZENG, 2011), news events (DING et al., 2014) and news events embeddings (NASCIMENTO; CRISTO, 2015).

In particular, deep auto regressive neural models are now widely used to improve the accuracy of predictions in time series. Most of these models are based on Recurrent Neural Networks (HOPFIELD, 1982) such as LSTMs (HOCHREITER; SCHMIDHUBER, 1997) and GRUs (CHUNG et al., 2015). More recently, however, temporal convolutional architectures (RENÉ; HAGER, 2017) inspired by WaveNet (OORD et al., 2016a) have been successfully applied on many time series prediction tasks with some advantages over classical RNNs such as larger long-term memory, easier and faster training. (BAI; KOLTER; KOLTUN, 2018)

As we will see later, temporal convolutional neural networks are not so far from a standard discrete wavelet transform operation². Such perception leads us to imagine that other fields, apparently very distinct, can bring important contributions to financial forecasting. One of the most interesting branches in digital signal processing is generative audio synthesis. WaveNet (OORD et al., 2016a) synthesizes human speech with utter naturalness in high-resolution and with high sampling rates. That model has inspired us to build a new stock forecasting architecture, StockNet, that aims at synthesizing a new price forecast model that will mimic human behavior in the buying and selling of assets in all its nuances, just as WaveNet does with vocal synthesis.

In this work, we intend to propose and evaluate how a deep architecture, based on temporal CNNs, will perform on stock forecasting using as input not only the stock prices, but also evidence extracted from financial News. Our hypothesis is that a deep neural network, used as extractor, can better capture useful latent patterns present in online news and rightly combine it with price series. Further, Temporal CNNs are expected to be easier to train in a scenario with long term events and larger sources of data. As result, we expect to observe improvements on financial prices prediction.

² Discrete Wavelet Transform (DWT) is a famous technique for extract hidden information patterns from a input signal in the scale domain.

1.1 Research Objective

Propose a neural-based method for representing sources of information extracted from financial News and, based on such information and the price history, predicting stock prices.

1.1.1 Specific Goals

- Build a model to represent information extracted from financial news to be used as evidence for stock price forecasting;
- Propose a stock prediction model based on a temporal convolutional architecture;
- Evaluate the proposed model with respect to its forecasting performance when compared with state-of-the-art methods in literature.

1.2 Research Contribution

The main contributions of this work for stock market prediction are (a) a model for representing financial news information as input for a timeseries forecasting model, (b) new intraday datasets with financial news and stock prices ready to be used in future research and (c) a state-of-the-art stock price prediction model.

1.3 Organization

This proposal is organized as follows:

- Chapter 1 (Introduction) presents the problem, general and specific research objectives and expected academic contribution.
- Chapter 2 (Fundamentals) explains some theoretical basis of the techniques used in this work.
- Chapter 3 (Related Work) presents the related literature review about stock markets time-series prediction.
- Chapter 4 (StockNet: A Multivariate Deep NeuralArchitecture for stock prices prediction) explains our proposed method.
- Chapter 5 (Experiments) describes our experiments, methodology and results.
- Conclusion briefly reviews results, their meaning and limitations.

2 Fundamentals

In this chapter, we describe concepts related to the prediction of stock prices using unstructured text and neural networks besides feature engineering steps that should be applied before training the model. Moreover, we also describe neural network methods we used in our architecture proposal.

2.1 Discrete-time Time-series

A time series is a numerical set of observations $X = \{x_0, \dots, x_{t-1}\}$, each one being recorded at a specific time t . It also can be understood as a stochastic process where random variables are indexed by time. A discrete-time time series is one in which the set T_0 of times at which observations are made is a discrete set, as is the case, for example, when observations are made at fixed time intervals. (BROCKWELL; DAVIS; CALDER, 2002)

A financial series comprises two components: (i) an index, in this case the time T , and (ii) a series of observed prices for each unit of that same time. These two components, by themselves, have inherent difficulties from their own nature. Prices, for example, may be used as an imperfect indication of the supply and demand of his related asset, but seem highly insufficient to inform all the factors present in its own market definition. At very large time scales, there is no stability of the currency to which prices refer, since currency is also an asset and can often experience changes resulting from government intervention, relative value perception, or inflation and interest rates. On the other hand, at small time scales there is only the last transaction. The time index, by itself, is not a good abstraction for the time in the real world. In fact, it does not even represent it, but rather the sampling frequency, that is, how many times per unit of time (and in discreet time) a price was observed.

Such inherent difficulties make it hard to place the problem itself and correctly identify its characteristics. There is no consensus, for example, on a kind of stochastic process that serves as adequate and generic abstraction for the stock market. It is not to be expected that, in the face of all these difficulties, the prediction of financial prices would be a simple task. Nevertheless, time series are heavily used in different fields of sciences such as statistics, weather forecasting, electroencephalography, signal processing, pattern recognition, econometrics, mathematical finance among many other fields.

2.2 Financial Time-series Prediction

We can describe a traditional auto regressive stock prediction as the joint probability of a sequence of prices $x = [x_1, \dots, x_N]$, given by the following product of conditional probabilities:

$$p(x) = \prod_{t=1}^N p(x_t | x_1, \dots, x_{t-1}) \quad (2.1)$$

where x_t is the price of the stock at timestep t and N is the price history size. In that way, a particular price is conditioned on all previous price information.

Given a news time-series input $\varphi = [\varphi_1, \dots, \varphi_N]$, where φ_t is the set of news headlines published at time frame t , there are two possible ways of feeding the auto-regressive model. The first one consists on performing arbitrary operations between price entry and news, as follows:

$$p(x|\varphi) \approx \prod_{t=1}^N p(x_t | x_1 + \varphi_1, \dots, x_{t-1} + \varphi_{t-1}) \quad (2.2)$$

The second one consists on adopting a predictive model which learns how to combine the two signals, as follows:

$$p(x|\varphi) \approx \prod_{t=1}^N p(x_t | x_1, \dots, x_{t-1}; \varphi_1, \dots, \varphi_{t-1}) \quad (2.3)$$

Therefore, the previous prices and News are always the inputs to make a particular price prediction at each time-step. Later, during the explanation of our proposed model, we will use the second method.

2.3 Using news content as input

Computational methods that use news texts as input for stock prediction models have been applied with reasonable success in recent years. Since all prediction computational algorithms require numeric inputs, these texts need to be handled and properly processed. The natural language processing (NLP) area has several techniques for word processing and information extraction that are commonly used in prediction models.

In particular, to use news content in machine learning methods, many information representation techniques have been adopted. They range from the simple bag-of-words to complex natural language processing (NLP) strategies such as named entity recognition (e.g., determine that *Apple* is an entity in sentence $S = \text{“Apple Inc has sued Samsung”}$),

noun phrase extraction (e.g, to detect *Apple Inc* as an entity in S), and structured event extraction (e.g: to detect $\{Apple\ Inc, sued, Samsung\}$ as an event in S).

The following subsections explain some NLP methods to process text input and how word embeddings and event extraction are made, since they are the techniques we used in our experiments.

2.3.1 Input text as sequence

One common way to feed a machine learning model with text is to represent its words (or characters) as integer numbers. For example, in the phrase “The love of money is the root of all evil”, we can choose a number for each word, as $\{the: 1, love: 2, of: 3, money: 4, is: 5, root : 6, all: 7, evil: 8\}$ and rewrite it as the following numeric sequence: $[1, 2, 3, 4, 5, 1, 6, 3, 7, 8]$ which is now treatable by a mathematical model.

After converting the text, we will have another setback. Most models need a fixed-size input to work, but, for instance, when feeding the model with the phrases “The love of money is the root of all evil” and “That is all about it”, represented here as sequences by $[1, 2, 3, 4, 5, 1, 6, 3, 7, 8]$, with length of 10, and $[9, 5, 7, 10, 11]$, with length of 5, we will have now two sequences of different lengths. To circumvent this problem we can cut the larger sequences (which would not be desirable as we would lose information) or pad the smaller sequences. For example, to feed a model with length of 12 by choosing the number 0 (zero) as the padding term, we will have both sequences with equalized sizes: $[1, 2, 3, 4, 5, 1, 6, 3, 7, 8, 0, 0]$ and $[9, 5, 7, 10, 11, 0, 0, 0, 0, 0, 0, 0]$.

2.3.2 One-Hot Encoding

One-hot encoding is a term borrowed from digital electronics that defines a binary method for mapping categorical data or words into numeric vectors for a feasible use as input by mathematical algorithms and machine learning methods. It is one of the simplest methods of converting input text into useful input for a natural language processing model. In one-hot encoding, each word in a dictionary represents a position in vector, where the vector size is also the size of the input language vocabulary. To represent a word, we simply fill its vector position with 1 (high signal level in electronics), while keeping all other positions marked zero (low level), in a such way that each input word is represented as an one-hot vector.

The main disadvantage of one-hot encoding is its sparsity. While only one bit of information is used to mark the position of a language term, all other positions maintain a zero. In languages with large vocabularies, such as English, each vector will need the size of this vocabulary to represent each single word. Of course, without any compression, there is also an inefficient memory consumption. The Second Edition of the 20-volume

Oxford English Dictionary contains full entries for 171,476 words. If we use one byte to represent each digit, as in most modern programming languages, and if we wanted to feed a model with Shakespeare's *Antony & Cleopatra* (26,456 words), using all English vocabulary as a one-hot encoding vector, we would use more than 4.6G of memory just to store this input. Feeding the model with larger corpus extracted from web would be unfeasible.

2.3.3 Bag of Words

A multi-set or a bag is a particular type of set where elements repetition is allowed. A Bag of Words (BoW) is a form of representing texts as input for machine learning and information retrieval models. A possible way of building a Bag of Words for multiple documents (or multiple fragments of an larger text) is using a sparse matrix, where a row i represents a distinct term from the input vocabulary and each column j can be seen as a vector of words representing a particular document. Thus, we can indicate the simple presence of a word w_i in the document d_j by filling in with 1 the position M_{ij} in the BoW matrix M , or even indicating how many times that term i appears in the document j by filling in the position with his frequency on that document.

The standard Bag of words modeling does not care about word order or syntactic and semantic aspects of the text. Therefore, there are some solutions to circumvent these limitations, such as using n-grams to capture syntactic text construction related to word order or using latent semantic analysis (LSA) (DEERWESTER et al., 1990).

2.3.4 Word Embeddings

Distributed representations of words as vectors in a continuous vector space is a common way to use text as input for natural language processing tasks. This technique, also known as word embedding, has been used since 1986 (RUMELHART; HINTON; WILLIAMS, 1986) in different areas related to natural language processing. This kind of word representation also relies on the distributional structure hypothesis (HARRIS, 1954), which states that the distribution of words in a text is not random and that close words in a text may also have similar semantics.

There are two ways to train a neural model for this task: the Continuous Bag of Words (CBOW) and the Skip-gram method. Those algorithms learn word representations that maximize the probabilities of a word given other contextual words (CBOW) or of a word occurring in the context of a target word (Skip-gram).

2.3.4.1 Continuous Bag of Words

The Continuous Bag of Words (CBOW) model is trained to predict a target word given the near words in the context (phrase or text fragment). For instance, given the phrase “The answer to life the universe and everything”, if we feed a CBOW model with the list of words [“the”, “answer”, “to”, “the”, “universe”, “and”, “everything”], we would expect the word “life” as its output. For sure, in this case we are inferring that the resulting output word “life” is the most common word in our corpus when the input context is [“the”, “answer”, “to”, “the”, “universe”, “and”, “everything”]. The same training is performed using all other phrases in the input text corpus. More formally, given a context C , a predicted word \hat{w}_t is inferred as:

$$\hat{w}_t = P(w_t | \{w_{t-c}, \dots, w_{t-1}, w_{t+1}, \dots, w_{t+c}\}) \quad (2.4)$$

where $\{-c \leq t \leq c; t \neq 0\}$ in which t is the word index and c represents half window size (how many words before and after the word w_t will be used to feed the model). To accomplish this, given the context C as input, we can infer the probability of the missing word by maximizing the average log probability over the T words in the vocabulary:

$$\hat{w}_t = \frac{1}{T} \sum_{t=1}^T \sum_{-c \leq j \leq c, j \neq 0} \log p(w_t | w_{t+j}) \quad (2.5)$$

This leads us to a particular sampling problem. In cases where a particular word is very common in a context, rare words in that context, while important, will never be predicted by the model, since they have a low probability. Thus, we will have a model that predicts only popular words and penalizes those rarely found in our input corpus. One alternative and simple way to avoid it is use the Skip-gram model.

2.3.4.2 Skip-gram

In Skip-gram (MIKOLOV et al., 2013), the objective is the reverse. A word serves as input to the model that tries to infer its “context”, that is, the words that are commonly found together. Therefore, in the phrase “The answer to life the universe and everything”, if we remove the word “life” and use it as input to the model, we can expect as the most likely output the sequence [“the”, “answer”, “to”, “the”, “universe”, “and”, “everything”]. For sure, in this case we are inferring that the resulting context phrase is the most common phrase, in our corpus, given the input word “life”. More formally, given a word w_t , a predicted context \hat{C} is inferred as:

$$\hat{C} = P(\{w_{t-c}, \dots, w_{t-1}, w_{t+1}, \dots, w_{t+c}\} | w_t) \quad (2.6)$$

where $\{-c \leq t \leq c; t \neq 0\}$ in which t is the word index and c represents the half window size (how many words before and after the word w_t will be used to feed the model). As before, given the word w_t as input, we now infer the probability of a related context by maximizing the average log probability:

$$\hat{C} = \frac{1}{T} \sum_{t=1}^T \sum_{-c \leq j \leq c, j \neq 0} \log p(w_{t+j} | w_t) \quad (2.7)$$

Figure 1 depicts the architecture of these two common (shallow) neural embedding models where $\{H_1, \dots, H_n\}$ are the neurons in the hidden layer, n is the embedding matrix length and c is the half window size. W_t represents a neuron at model border, which will receive a context and predict a word or receive a word and predict its context.

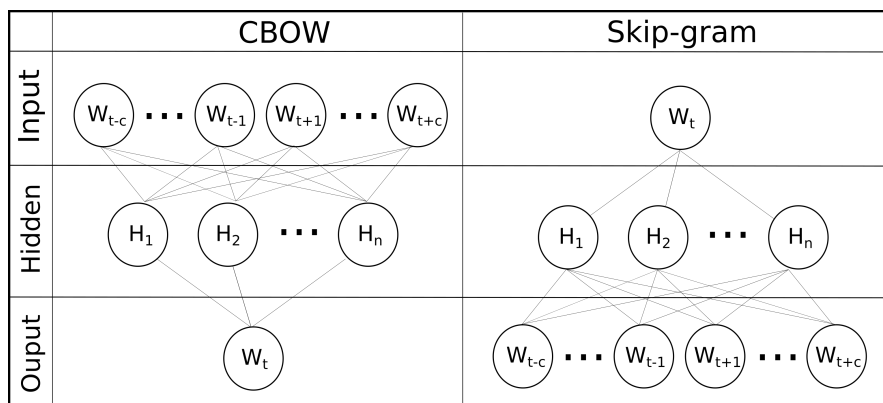


Figure 1 – CBOW and Skip-gram architecture overview

During the training of some of these models, such as word2vec (MIKOLOV; YIH; ZWEIG, 2013), it was possible to verify that words of similar meanings were found geometrically close together. For example, conceptually close words (e.g.: *uncle* and *aunt*, *lion* and *tiger*) are expected to be spatially close. Certain regularities became evident in these word representations and some arithmetic operations can even be used as semantically logic operations. For example, the operation $\text{vector}(\text{'Paris'}) - \text{vector}(\text{'France'}) + \text{vector}(\text{'Italy'})$ results in a vector very close to $\text{vector}(\text{'Rome'})$. Semantic arithmetic also implies that the meaning expressed by a set of word embeddings can be exploited by linear and non linear models.

This characteristic is due to the process by which these embeddings are trained. The purpose of these neural models is to predict the missing word in a context or vice versa, but what interests us is what happens in its hidden layers. Using a sufficiently large input text corpus, during training, the weights of the hidden layer neurons seem to be optimized in a such way that words commonly used together in the input corpus are also found geometrically close in a spatial projection of that hidden layer.

After training, the hidden layer weights of a neural network are used as embedding representation. As CBOW and Skip-gram are unsupervised, they are able to effectively learn patterns from billions of word occurrences.

2.3.5 Information Extraction

Information Extraction (IE) is growing as one of the active research areas in artificial intelligence for enabling computers to read and comprehend unstructured textual content (Etzioni et al., 2008).

Information extraction (IE) turns the unstructured information expressed in natural language text into a structured representation (Jurafsky and Martin, 2009). That structured representation can be modeled in the form of tuples containing phrase syntactic elements present in the semantic relationships among them. For instance, the tuple $\{arg_1; relation; arg_2\}$ represents a relations among elements arg_1 and arg_2 .

2.3.5.1 Open information extraction

Open information extraction (OpenIE) contrasts with traditional information extraction methods, since it is not limited by structured information domains, as in ontologies. Additionally, to reduce the knowledge acquisition cost, since defining ontologies is a manual task, OpenIE methods try to find indirectly potential relations looking on syntactic constructions among entities present in unstructured texts. After that, these constructions are represented as triples describing relations (P) between actors (A) and objects (O) as in $\{A, P, O\}$. For instance, $\{‘Romeo’, ‘loved’, ‘Juliet’\}$ or $\{‘The cat’, ‘sat’, ‘on the mat’\}$. OpenIE methods can be learned by machine learning models or even constructed using heuristics.

The first Open Information Extraction method was called TextRunner (BANKO et al., 2007). Later, improvements and new versions of OpenIE were introduced by other authors as Reverb (ETZIONI et al., 2011), OLLIE (MAUSAM et al., 2012), ClausIE and (CORRO; GEMULLA, 2013).

In the Stanford Open IE approach (ANGELI; PREMKUMAR; MANNING, 2015), a classifier was learned to split a sentence into a set of logically implied shorter statements, recursively traversing its dependency tree and predicting at each step whether an edge should produce an independent clause. After that, each independent clause was then shortened by executing a natural logical inference on it. Finally, a small set of 14 manually created patterns was used to extract a triple of predicate argument from each statement. An overview of this approach is shown in Figure 2.

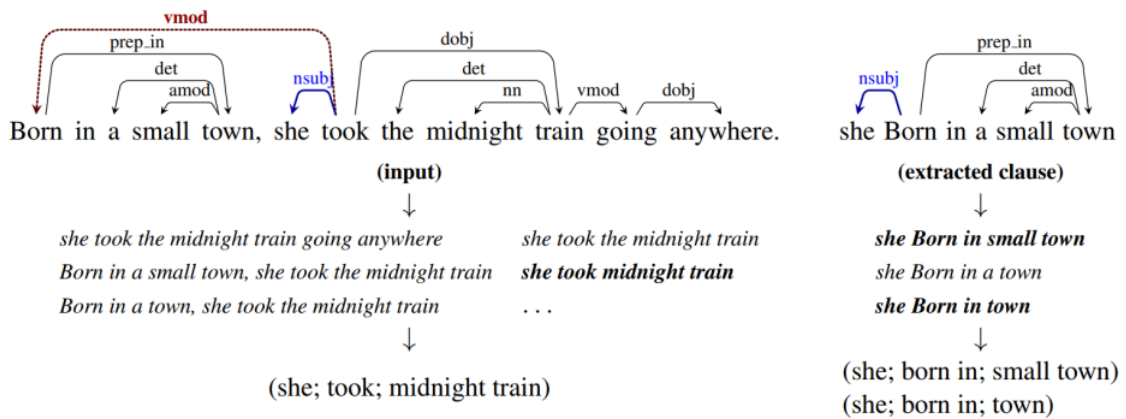


Figure 2 – An illustration of Stanford Open IE’s approach. From left to right, a sentence yields a number of independent clauses. From top to bottom, each clause produces a set of entailed shorter utterances, and segments the ones which match an atomic pattern into a relational triple (ANGELI; PREMKUMAR; MANNING, 2015).

2.3.6 Structured Events Embeddings

Traditional methods to forecast stock prices based on News rely on auto-regressive predictors that incorporate News content as overlay data. In such models, prices are taken as a time series where future prices are seen as a combination of past prices and concepts represented by sets of words, phrases, or structured events.

As previously described, words and phrases from unstructured text can be directly fed to a neural model using sequences, one-hot encoding, Bag of Words, Structured Events, and Embeddings (Continuous Bag of Words or Skip-gram representations). A common way of feeding structured events as input for neural models is applying Open Information Extraction (OpenIE) (BANKO et al., 2007) methods as a pre-processing step on unstructured text. A combined way would be represent the extracted events triples as embeddings.

Structured Events embeddings (SEEs) (NASCIMENTO; CRISTO, 2015) are structured events (SEs) (DING et al., 2015) represented in concept spaces learned such that SEs share a semantic distributed representation. SEs are a subset of all relationships extracted using OpenIE method. This subset contains only event based relations, that is, actors performing their actions on an object.

SEE treats each event as a set of words, where each word is represented using the Skip-gram algorithm. More specifically, each word is represented by a feature vector in a feature space. (NASCIMENTO; CRISTO, 2015) represented structured events as the summation of their constituting word vectors, in each day, using the skip-gram implementation by Mikolov *et al.* (MIKOLOV et al., 2013), referred to as *Word2Vector*

and publicly available.

In this method, a text document can be viewed as a sequential stream of events. An structured-event (SE) is composed of an action P , an actor O_1 that conducted the action, and an object O_2 on which the action were performed. Formally, an event is represented as:

$$E = (T, O_1, P, O_2) \quad (2.8)$$

where T is the timestamp, O_1 is the actor, P is the action and O_2 is the object (DING et al., 2014). As an example, the News fragment “Microsoft purchases Nokia’s phone business” should be extracted as $\{Microsoft, purchases, Nokia’s\ phone\ business\}$, as exemplified in Figure 3

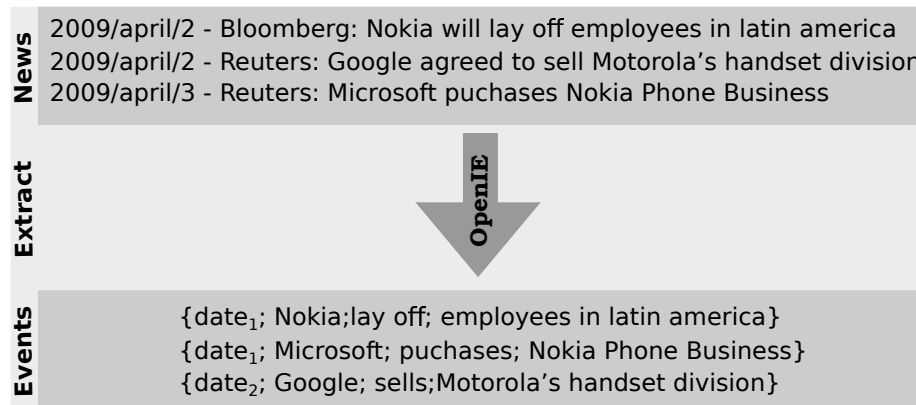


Figure 3 – Extracting events from news stream. (NASCIMENTO; CRISTO, 2015)

Raw text data are translated into events using the Open Information Extraction (BANKO et al., 2007) approach. As OpenIE toolbox as Reverb (ETZIONI et al., 2011), an extractor for verb-mediated relations, is used to extract the events super set.

2.3.7 Sentiment Analysis

Another way to extract high-level information from text for later use in predictive models is by using sentiment analysis. A process of sentiment analysis is the task of extract the polarity of sentiments in a input text, which can be positive, negative or even neutral in the cases where the sentiment polarity is not explicit in the text or when the model was not able to detect it. In addition, this analysis can be modeled to be performed at three granularity levels: Document Level, Phrase Level, and Aspect Level. (FANG; ZHAN, 2015)

At the document level, it is useful to determine if a particular document, such as an entire product review or a twitter comment for example, has positive or negative

sentiment polarity. At this level, the task is to analyze whether the whole opinion of that document communicates a positive or negative sentiment. At the sentence level, we try to identify the sentence polarity. At the aspect level, the observed entities and their characteristics (aspects) are evaluated for their eventual positive or negative polarity. For instance, given the review “Amazing food, desserts and wine selection. Great place for a dinner for two. Although the waiter was not so attentive, I am happy with the overall experience.”, it is clearly positive at document level with positive aspects (e.g., “food”, “desserts”, wine”) and, at least, one negative (“service”) observed in a negative sentence, “the waiter was not so attentive”.

Term frequency of words related to human emotions is a traditional form to infer the mood of a particular publication or comment on social networks. For example, we might infer that a high number of words like “good”, “excellent”, “great”, “joyful”, “amazing”, or “happy” in a particular document would denote that it has a positive content or expresses the happiness of its author.

2.3.7.1 Sentiment Analysis of Financial Texts

Detecting investors mood by these methods seems useful in stock prices prediction. BOLLEN; MAO; ZENG have even shown that the mood of Twitter users can even affect the stock price and be used for financial forecasting.

However, determining whether the content of news is useful in predicting stock prices is a hard task. Beyond the classic problems of sentiment analysis field, such as the difficulty in perceiving irony, it is also clear that capturing the happiness or sadness of a certain news document seems not so useful in forecasting. In that cases, we are interested in detecting how much a financial news document or headline will impact the stocks prices of a company to which it refers and not whether this news is happy or sad. Therefore, some lists of words more suitable for financial analysis have been made. A commonly used source is the list made by (LOUGHRAN; MCDONALD, 2011) that provides 353 positive and 2,337 financial negative words.

2.4 Artificial Neural Networks

We often need to construct less complex abstractive models to describe extremely complex nature systems and then apply them to solutions of particular problems. Von Neumann states that “Natural organisms are, as a rule, much more complicated and subtle, and therefore much less well understood in detail, than are artificial automata.” (NEUMANN, 1951).

In fact, there are even questions about how information is processed and stored in the brain. We have now two main neurobiological hypotheses to explain how information

is stored by the human brain. The symbolic hypothesis argues that, upon perception, information is encoded in our brain in a likely decipherable and similar manner among all humans. Such coding would establish a one-to-one relationship between information and that symbolic format. The connectionist hypothesis, on the other hand, holds that such a relationship is not so simple, and that the chain of neurons itself would be responsible for representing information through their synaptic connections. Those of artificial neurons we will present are based on the connectionist hypothesis. (ROSENBLATT, 1958) which was widespread and had important supporters when the perceptron model was proposed (KNOTT, 1951) (HAYEK, 1952) (UTTLEY, 1956) (ASHBY, 1952).

We know that a typical biological neuron consists basically of a cell body (also called soma), some cellular extensions and its branches (dendrites) and a third component called Axon, that carries nerve signals away from (and back to) the soma. The transmission of information between different neurons through these axons forms a biological neural network. Recent research findings have suggested that neuronal soma membrane excitability and synaptic connections among neurons throughout the axons produce behavior and cognition (RUTECKI, 1992).

Artificial neurons were born from the mathematic simplified abstraction of real biological neurons. The very first mathematical model of an artificial neuron was the Threshold Logic Unit (DUDA; SINGLETON, 1944), but the most famous model was the Perceptron (ROSENBLATT, 1958), briefly described in Figure 4, where the capital sigma represents the weighted sum of the n total inputs x_1, \dots, x_n and the tiny sigma represents the activation function which will “decide” if the summed signal will flow to the output or not. In the first version of perceptron, proposed by ROSENBLATT, the activation function was as simple as checking whether the weighted sum of the input weights was greater or less than a threshold, but several other types of functions have been used over the years.

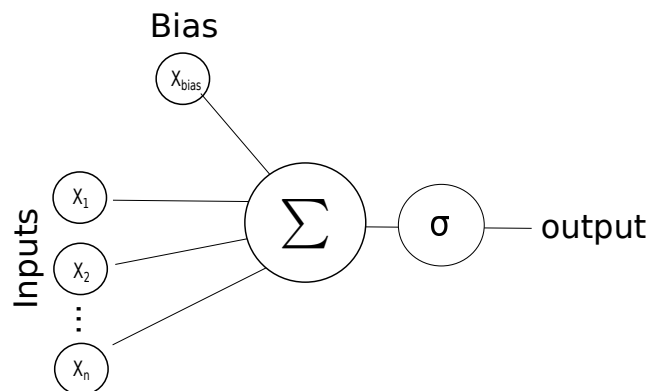


Figure 4 – Rosenblatt’s Perceptron Representation

The bias input (x_{bias}) is used to shift the activation function image positively or

negatively. More formally, the perceptron model can be described by:

$$output = \sigma(x_{bias} + w_1x_1 + w_2x_2 + \dots + w_nx_n) = \sigma(\mathbf{w}^T + b) \quad (2.9)$$

MINSKY; PAPERT demonstrated severe limitations in the single hidden layer perceptron model, such as the impossibility of learning the xor function. This publication had a striking effect on the artificial neural network field, considerably reducing investments in academic research based on the connectionist hypothesis by a few years.

The roots behind this general scientific “overreaction” are beyond our research scope. Fortunately, later more powerful techniques such as adding multiple perceptron layers (which solves xor problem) and the backpropagation (RUMELHART; HINTON; WILLIAMS, 1985) (which facilitates the network train), contributed to rekindle the research interest in the neural network field.

2.4.1 Activation Function

It is noticeable that matrix multiplications between inputs and neuron weights are later normalized by a mathematical function that is also responsible, in some cases, for adding a nonlinear transformation to these weights: the activation function, represented as σ at Figure 4. Several types of activation functions can be chosen for a particular layer in a neural network. In Figure 5, we list some popular activation functions, in order: Threshold (also called Heaviside step or binary step), Sigmoid (also called logistic), tanh, Rectified linear unit (or simply ReLU (NAIR; HINTON, 2010)) and Leaky ReLU (MAAS; HANNUN; NG, 2013).




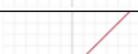

Name	Graph	Function Form	Image
Threshold		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	{0, 1}
Sigmoid		$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$	(0, 1)
tanh		$f(x) = \tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$	(-1, 1)
ReLU		$f(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases}$	[0, ∞)
Leaky ReLU		$f(x) = \begin{cases} 0.01x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	(-∞, ∞)

Figure 5 – List of common activation functions.

2.4.2 Loss Function

The loss function (also called cost function) give us a measure of how wrong the model is in terms of its ability to estimate the classification or prediction with respect to input data. This metric is usually calculated as a difference between the real correct value and the value predicted by the neural network and depends on the type of output we expect from the model output. The most commonly used loss functions for time series autoregressive prediction are Mean Absolute Error (MAE) and Mean Squared Error (MSE), formally described by the following equations:

$$MAE = \frac{1}{N} \sum_{t=1}^N |y_t - \hat{y}_t| \quad (2.10)$$

$$MSE = \frac{1}{N} \sum_{t=1}^N (y_t - \hat{y}_t)^2 \quad (2.11)$$

where N is the number of training instances, \hat{y} is the prediction made by the forecaster, and y is the real value.

However, it is well known that, in financial time series, errors that result in gains should not be punished in the same way as those that result in financial losses. GRANGER proposed a linear-linear loss function (LIN-LIN) that weights positive and negative forecast errors of similar magnitude differently (asymmetric loss). In the case of asymmetric loss functions, optimal forecasts are consistent with neither the Mean Square Error (MSE) nor Mean Absolute Error (MAE) criteria and both mean and median forecast errors can have expected values different from zero (KEANE; RUNKLE, 1998). HONG; LEE conducted a comprehensive analysis of different loss functions used in financial forecasting.

In fact, the mean absolute error loss function penalizes forecast optimism and pessimism equally and the respective explanation for the forecast bias is attributable to skewness in the distribution of earnings. However, some studies suggest that using asymmetric functions has no benefit over the mean absolute error loss function and that analyst motivation may be driven by the costs associated with under-predicting earnings being higher than the costs of over-predicting earnings, i.e., asymmetric loss functions. (SINGH, 2015).

The right choice of the cost function is also an important step for computational costs, since it will and its complexity will also increase the complexity of backpropagation weight calculations, as all derivatives and calculations will use it by base, as detailed in the next section.

2.4.3 Backpropagation

Methods for gradient error minimization have existed since the 19th century (CAUCHY, 1847). The iterative optimization of weights for equation systems is even older, and has been carried out since the advent of chain rule in differential calculus by Leibniz (LEIBNIZ, 1684) and l'Hôpital in the 17th century (L'HOPITAL, 1696). But, the back-propagation for weight optimization in the strict context of neural networks was first proposed by Werbos (WERBOS, 1974) with posterior contributions from Rumelhart (RUMELHART; HINTON; WILLIAMS, 1985). This method propagates the prediction error to all network weights after its calculation using some specified loss function.

According to GOODFELLOW; BENGIO; COURVILLE, we can calculate the backpropagation for any neural network using only the next four formulas described below. Considering L the number of layers in a neural network, C a arbitrary chosen loss function, we can obtain the error in the output layer δ^L by the following Hadamard product:

$$\delta^L = \nabla_a C \odot \sigma'(z^L) \quad (2.12)$$

where $\nabla_a C$ is the gradient of the cost function with respect to a , which is the result of the activation function from the previous layer. The error $\sigma'(z^L)$ as the derivative of the output activation function applied to z , which is the weighted input to the activation function for neurons in output layer. The error δ^l for each next layer l is calculated as follows:

$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l) \quad (2.13)$$

where $(w^{l+1})^T$ is the transpose of the weight matrix from the $(l+1)^{th}$ layer and δ^{l+1} is the error propagated from that layer.

Now we can also easily calculate the rate of change of the cost C with respect to any bias b_j^l by:

$$\frac{\partial C}{\partial b_j^l} = \delta_j^l \quad (2.14)$$

And the rate of change of the cost C with respect to any weight w_{jk}^l by:

$$\frac{\partial C}{\partial w_{jk}^l} = a_k^{l-1} \delta_j^l \quad (2.15)$$

All above equations are in the simplified fully matrix-based form.

2.4.4 Weight Initialization

One of the initial choices when training a neural network is how to initialize the network weights. The initial state of our training will depend on that decision. We can choose from several types of initialization, such as simply setting all weights to zero, initializing them to random values according to a previously chosen statistical distribution, or even combining activation functions that also are “aware” of the initialization of weights, such as ReLU (NAIR; HINTON, 2010) and Leaky ReLU (MAAS; HANNUN; NG, 2013) that use the method in Equation 2.16 proposed by (HE et al., 2016) or such as *tanh* that uses Equation 2.17, proposed by (GLOROT; BENGIO, 2010). Both equations multiply a Gaussian random weights initialization W_σ , in each network layer l , in the following way:

$$W_l = W_\sigma \cdot \sqrt{\frac{2}{N^{l-1}}} \quad (2.16)$$

$$W_l = W_\sigma \cdot \sqrt{\frac{1}{N^{l-1}}} \quad (2.17)$$

where W_l is the initialized weights of layer l and N is the number of input units in the weight tensor.

This choice of the initialization technique can hugely impact network learning. A weight too close to the upper activation limit can easily saturate the neuron, while values too close to the lower activation limit will fatally turn the neuron off. Both conditions, occurring in a sufficiently large part of neurons, would difficult the network learning.

2.4.5 Input Normalization

We usually need to transform model input values to suitable scales. For a network that uses the sigmoid as activation function, for instance, it would be appropriate if the input values were also contained within the same range, in this case, between 0 and 1. In fact, we know that neural networks and specially convolutional networks are highly sensitive to scale (HU et al., 2018a). To avoid that problems, we need to perform a scale conversion operation to a specific numerical range, this process is also called normalization.

The simplest and most common way to normalize an input dataset is through the formula:

$$z_i = \frac{x_i - \min(X)}{\max(X) - \min(X)} \quad (2.18)$$

where $X = (x_1, \dots, x_n)$ is the raw input and z_i is the i^{th} normalized input.

2.4.6 Recurrent Networks

The networks mentioned so far, also called feedforward, use only the current and unchanging state of the network, that is, they use their weights in all their layers, for all their predictions. However, in some problems, such as sequence prediction, each sample is temporally dependent on the previous ones. Recurring networks are used in these cases when we need previous states of the network to be used in future forecasts. To achieve this, recurring networks feed back into the network with previous outputs, as shown in Figure 6. In this way the network will see all previous inputs and outputs at each forecast step, while learning to remember those with impact on the current forecast.

Since they were proposed, we know that recurring networks are hard to train (HOPFIELD, 1982). In addition to the high memory consumption required to maintain all previous states, we will often not be able to converge network weights due to the huge amount of successive gradients applied during backpropagation computation, which will inevitably converge to zero, and because of the limitations of discrete computational operations will result in zero, preventing the error propagation. This problem is known as vanishing gradient.

To help solve these two main problems, some recurring architectures have been proposed, the most famous being LSTM (HOCHREITER; SCHMIDHUBER, 1997) and GRU (CHUNG et al., 2015) which function as electronic circuits that have memory cells capable of learning which stimuli are important in each recurrence layer and which of these stimuli should be attenuated or even forgotten (acting as a deterministic regularization). That is, these architectures, although they seem more complex, are in the background reducing the complexity of the final model.

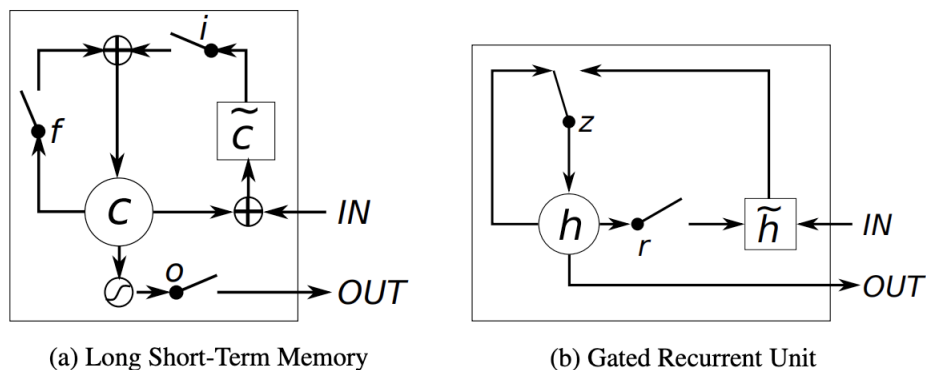


Figure 6 – Illustration of (a) LSTM and (b) GRU memory cells. (a) i , f and o are the input, forget and output gates, respectively. c and \tilde{c} denote the memory cell and the new memory cell content. (b) r and z are the reset and update gates, and h and \tilde{h} are the activation and the candidate activation. (CHUNG et al., 2014)

2.4.7 1D Convolution

Despite being widely used in image classification, 2D convolution does not seem suitable for sequence classification and prediction. A numerical sequence is by definition an one-dimensional vector, so one-dimensional convolution is required. In this type of convolution, one-dimensional vectors will be learned, working as kernels that will be convolved with inputs during prediction or classification.

Suppose an 1D input signal $f = [f(0), \dots, f(N - 1)]$ and an 1D receptive field $g = [g(0), \dots, g(M - 1)]$, where M and N are their sizes. The discrete convolution of g and f is given by:

$$\text{Convolution}(f, g)(i) = (f * g)(i) = \sum_{j=0}^{M-1} f(j) \cdot g(i - j) \quad (2.19)$$

2.4.8 Causal Convolution

The problem with feeding convolutional networks with a sequence is that both past and future inputs can be “seen” by the model. In order to avoid that, it is necessary to mask the input so that only the current input and the strides are seen by the net at each training step. That technique is also called causal convolution, since this establishes a causal relationship to be maintained between previous and current inputs.

2.4.9 Residual Connections

To prevent the neural network from “wasting time” trying to learn raw input rather than optimizing its weights accordingly, HE et al. suggested the residual connection. Bypass connections between layers are used to bring the unchanged input information to the next layers. In other words, each entry is multiplied by an identity matrix and added to the layer to which we want to make the residual connection. It works as a wire directly from the entrance of one layer to another.

Such a procedure may seem extremely simplistic, and even questionable, at first glance, but it has obtained great results in several deep neural networks, reducing training time and increasing the number of layers by avoiding the vanish gradients. (HE et al., 2016) (TAI; YANG; LIU, 2017) (OORD et al., 2016a) (WANG et al., 2017)

2.5 Temporal Convolutional Networks

The Temporal Convolutional Network term was firstly adopted by (RENÉ; HAGER, 2017) in the activity segmentation field. BAI; KOLTER; KOLTUN also adopt this term not as a label for a truly new architecture, but as a simple descriptive term for a family of architectures for convolutional sequence prediction.

According to (BAI; KOLTER; KOLTUN, 2018), a Temporal Convolutional architecture has two distinct characteristics: 1D convolutions and Causality. In these models, the input of the network has the same size of the output, which increases in each step, having as an additional input for the next prediction the result predicted in the previous step.

2.5.1 Kernel Dilatation

That requirement about to keep the length of the output vector the same as the input leads us to a new problem, as soon as each convolution or pooling operation will inevitably reduce the output vector dimensionality in all network layers. To solve this problem MITTELMAN proposed a five-layers 1D-convolutional model with different receptive field sizes upsampled through the layers to preserve the input length. BOROVYKH; BOHTE; OOSTERLEE introduced the dilated conditional convolutions for time-series forecasting.

The purpose of the dilatation is to use larger kernels with internal zero-paddings to keep the output with the same size as the input after the convolution and pooling operations. The introduction of dilation method has highly increased the depth of causal 1D convolution networks.

2.5.2 Wavelet Transform and 1D Convolution

The financial time series are indexed numerical sequences, so they can be analyzed and treated by techniques applied to other types of indexed numerical series.

It is also possible to understand a time series as an analog signal in the time domain and thus make use of methods from the electronics and signal processing areas in its analysis. A Fourier transform could be used to bring this financial series into the frequency domain and thus discover which spectral levels are present in the signal. These spectral levels could be understood as a histogram of the volatility densities found in the financial series.

We can compute a Fourier transform \mathfrak{F} at a particular frequency k for an input signal x_i in its discrete form by:

$$\mathfrak{F}_k = \frac{1}{N} \sum_{i=0}^{N-1} x_i e^{i2\pi \cdot k \frac{i}{N}} \quad (2.20)$$

where N is the number of sampled points extracted from the input signal, $e^{i2\pi}$ is the spinning operation around the circle at frequency k .

The Fourier transform breaks the input signal into the sum of sinusoids representing all the different frequencies that make up the signal. Such behavior reminds the extraction

of time series components by the classical econometric models, which decompose the time series into its components like trend, seasonality and white noise. However, such an approach is not suited to work in non stationary or transitory characteristics of financial timeseries. Furthermore, it would not be able to indicate in the time domain what it indicates in the frequency domain. In other words, an acceleration in the volatility of an asset would be detected using this kind of transform, but such a procedure would be unable to tell us where exactly that perturbation occurred in time.

We also could use another important math tool that would convert the input signal to the scale domain, the wavelet transform, as a useful method to indicate in which time point ρ such volatilities are located and at what scale s it would be possible locate a pattern of interest ψ , as follows:

$$\hat{F}(s, \rho) = \int f(\vec{t})\psi(s, \rho) \cdot dt \quad (2.21)$$

where ψ is the chosen wavelet function and the inputs s and p are respectively its scale and the time shifting on the chosen wavelet function.

If we, intuitively, imagine the entire process applied by a wavelet transform, that breaks down a signal in a sum of wavelet coefficients (set of different scaled and shifted versions of a previous chosen wavelet function) convolved with a input signal in order to filter out surges, attenuation or other patterns, we would inevitably imagine it as a convolution operation inside a networks neural. The wavelet coefficients, in this case, would be nothing more than the weights to be learned by the kernel and its scale could be understood as the dimensions of that kernel.

In fact, at first glance, Wavelet transform seems really similar to previous explained convolution operations, but it has as premise that the the input signal should be at least partially continuous and integrable. Even using its discrete and less computationally expensive Wavelet forms, to manually choose a pattern from a bank of wavelets functions for each prediction problem is not a feasible task. The main advantage of 1D convolutions over traditional wavelet transforms lies exactly in the fact that the interest patterns (kernels) are automatically learned by the network during training.

2.5.3 Dilated Causal Convolutional Architecture

Based on WaveNet (OORD et al., 2016a), a recent convolutional architecture for audio waveforms, Borovykh's (BOROVYKH; BOHTE; OOSTERLEE, 2017) model seems better captures the long short term dynamics, since each model layer has a resolution on time-series sampling twice larger than the prior layer.

Compared with other RNN architectures, such as LSTMs and GRUs, dilated CNNs have two important advantages. First they better take advantage of the parallelism since

that, in RNNs, a recurring step is just performed after the previous one. As a dilated CNN “sees” the entire sequence at once, it is much more parallelizable. Further, convolution operations are also much less susceptible to the vanishing gradient problem and learning is very effective with the adoption of residual connections (HE et al., 2016), usually used in CNNs. The overview of the dilated causal convolutional network structure is presented in the Figure 7.

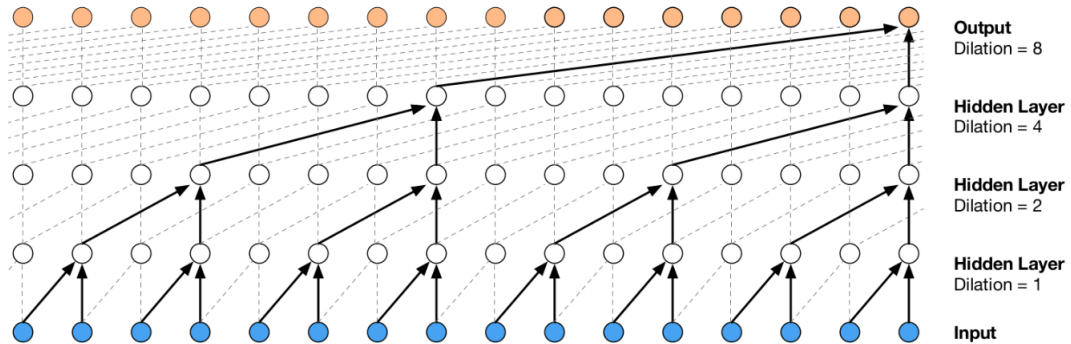


Figure 7 – Dilated Causal Convolutional Neural Network Structure (OORD et al., 2016a)

In this figure we observe that all the network weights in the previous steps are used in each future prediction, however not all weights are considered in the prediction. The output uses only two weights from the previous layer, which in turn also use only two weights from its previous layer and so on. If we look at the image from bottom to top (from input to output), we realize that the next layer weights are no longer considered by a ratio of 2, which is the value of the network dilation. Thus, the output will be connected with all input examples, but it “forgets” about them each layer according to the dilation rate. In other words, the network “sees” the input signal in lower resolution at each layer.

This different signal resolution in each layer of a dilated CNN helps it to dynamically capture patterns at different temporal granularities. In WaveNet (OORD et al., 2016a), for example, each layer has twice the resolution of the previous one. This peculiarity helps the model to use both long-term and short-term patterns simultaneously in prediction. Skip-connections, as presented on Figure 8, also help accelerating training time while enabling deeper models. Its Residual connections, in turn, help the model to quickly rebuild the input signal through a continuous stream, preventing the network from wasting time trying to learn the input signal during the forecast.

In Figure 8 we note the model masks the previous inputs through a causal convolution and initiates the dilated convolution to avoid reducing the output size using the product of tanh and a sigmoid as activation function. The result of this composite activation function then goes through a 1D convolution, here called 1x1, and is added to the input for residual propagation as well as being propagated outside the layer and added

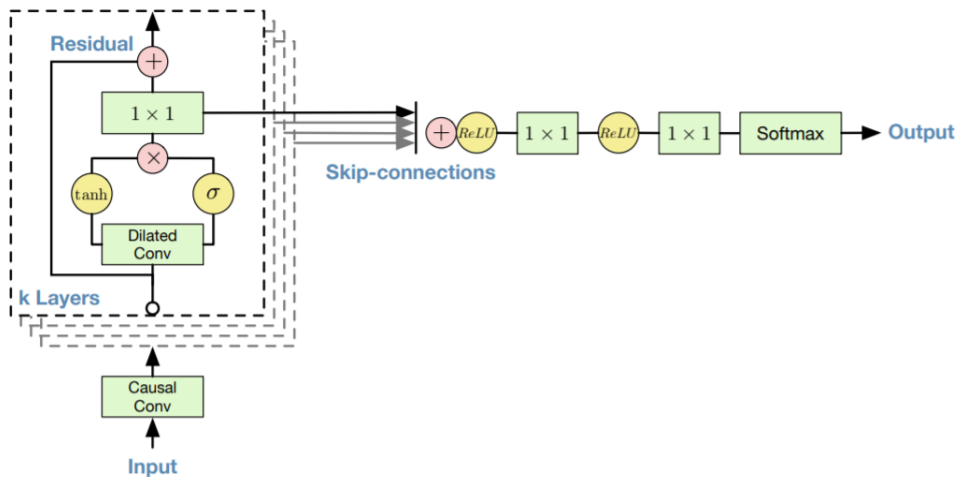


Figure 8 – Residual block and WaveNet architecture (OORD et al., 2016a)

to the results of the other next layers. Finally, two sets of ReLU/1D convolution are then applied to a last softmax to predict the probability distribution at the output.

It is important to note that WaveNet was originally proposed for voice synthesis, in addition to some experiments conducted with music synthesis in its generative form. Using WaveNet for financial forecasting is an alternative but fully viable proposal given the autoregressive nature of the problem.

3 Related Work

3.1 Financial time-series prediction methods

The classic econometric models commonly used in financial time-series forecasting include the autoregressive model (AR), the moving average model (MA), the autoregressive moving average model (ARMA), the autoregressive integrated moving average model (ARIMA) and vector autoregressions (VAR) (HAMILTON, 1994). All of these methods make a future price estimate using a linear auto regression. That is, the same variable is sampled at a discrete frequency and this history is used for future predictions through a linear regression learned by the model.

Signal processing and machine learning based techniques also have been used in stock price forecasting, such as artificial neural networks (ROJAS, 2013) on (SONG; ZHOU; HAN, 2018), support vector machines (HEARST et al., 1998), and ensembles (DIETTERICH, 2000), since that discovering useful patterns in time series values is a non-trivial task. For a full revision of the prior literature, we refer the readers to the surveys published by (YU; KAK, 2012) and (FU, 2011). Even more recently, deep learning models using core traditional signal processing methods as wavelet transform remain being applied successfully (LIANG et al., 2019).

3.2 Text based techniques

Recent years also have seen a large increase in the adoption of textual data extracted from the web and social networks to attempt to create better predictive models in various application fields (YU; KAK, 2012). In fact, Twitter mood was already used to improve stock predictions (BOLLEN; MAO; ZENG, 2011) and (HUANG et al., 2015) even had used Granger Causality Analysis to demonstrate the importance of a Twitter mood time-series in financial time-series prediction. Twitter based frameworks for stock prediction have been heavily used by investors by now (DAS et al., 2018).

Traditional models used raw text features as input, such as bag-of-words. However, Ding *et al.* (DING et al., 2014) achieved a substantial improvement on stock-price polarity (up and down) prediction by using structured events and a multilayered feed-forward neural network. Using the same dataset, Peng and Jiang (PENG; JIANG, 2015) also adopted structured events to analyze market movement polarity using a deep neural network.

Embeddings were first introduced by (BENGIO et al., 2003). In that work, words were represented by neuron activations of a hidden layer of a neural network built to

distinguish correct sentences (“the cat sat on the bed”) from random sequences of words (“bed the on sat”). As result, related words are represented by similar patterns of neuron activations.

More recently, very effective ways to obtain similar representations have been introduced, such as the unsupervised algorithms Conceptual Bag of Words (CBOW) and Skip-gram with Negative Sampling (Skip-gram) (MIKOLOV et al., 2013).

Sentiment Analysis techniques also have been used for a long time in unstructured text based machine learning models. A recent paper (PROSKY et al., 2017) made some experiments on Sentiment Predictability for Stocks using Vader (GILBERT, 2014) and Google Sentiment API without reach important results, since his model training did not converge. For a full revision of the sentiment analysis and NLP based stock forecasting literature, we refer the readers to the surveys published by (ABIRAMI; GAYATHRI, 2017) and (XING; CAMBRIA; WELSCH, 2018).

3.3 Deep Models for financial time-series prediction

LSTMs (HOCHREITER; SCHMIDHUBER, 1997) were also used to improve language models (SUNDERMEYER; SCHLÜTER; NEY, 2012). A newer version of LSTM using Gated Return Units (CHUNG et al., 2015) was also used to predict stock prices, but only as a replacement of traditional methods for time series, instead to adding textual information from news. A recent survey of time-series prediction using deep learning methods was made by (GAMBOA, 2017)

Dilated Conditional Convolutional Neural Network (DC-CNN) (OORD et al., 2016a) architectures and the Augmented WaveNet have demonstrated better performance than traditional CNN design and comparable or even better performance than LSTM on forecast time-series (BOROVYKH; BOHTE; OOSTERLEE, 2017).

Bidirectional-LSTM with self attention mechanism also have been used to encode news text and capture the context information in order to predict directional changes in both S&P500 and individual companies stock. (LIU, 2018). More recently, a method similar to a Discrete Wavelet Transform (DWT) have used a LSTM model with Phase-Space Reconstruction (PSR) method to predict market polarity. (YU; YAN, 2019)

The Temporal Convolutional Network term was first adopted by (RENÉ; HAGER, 2017) in the action segmentation field. BAI; KOLTER; KOLTUN also adopt this term, not as a label for a truly new architecture, but as a simple descriptive term for a family of architectures for convolutional sequence prediction. Temporal Convolutional Networks are, in a bottom line, just causal 1D networks with causal constraints between model layers. The first causal 1D convolution model was used by (MITTELMAN, 2015). New models

have used more complex and allegedly more effective approaches to predicting stock trends only (polarity) using Hybrid Attention Networks (HU et al., 2018b), Adversarial Networks (FENG et al., 2018) and Multi-filters Neural Networks (LONG; LU; CUI, 2019).

It is important note that all previous deep architectures were used to predict only the price polarity and not the price of stock market assets. In other words, while most related works are trying to predict if a particular stock price will increase or decrease, we are trying to predict which will be their exact stock price.

3.4 SeriesNet - A novel and complex stock forecasting architecture

A recent work using a hybrid architecture of dilated convolutions and LSTMs has presented good results in stock forecasting: The SeriesNet (SHEN et al., 2019). This model does not use news as input, but we decided to compare our model against it, since it is a state of the art work and the metrics and methodologies used in our proposed architecture can also easily be applied to SeriesNet.

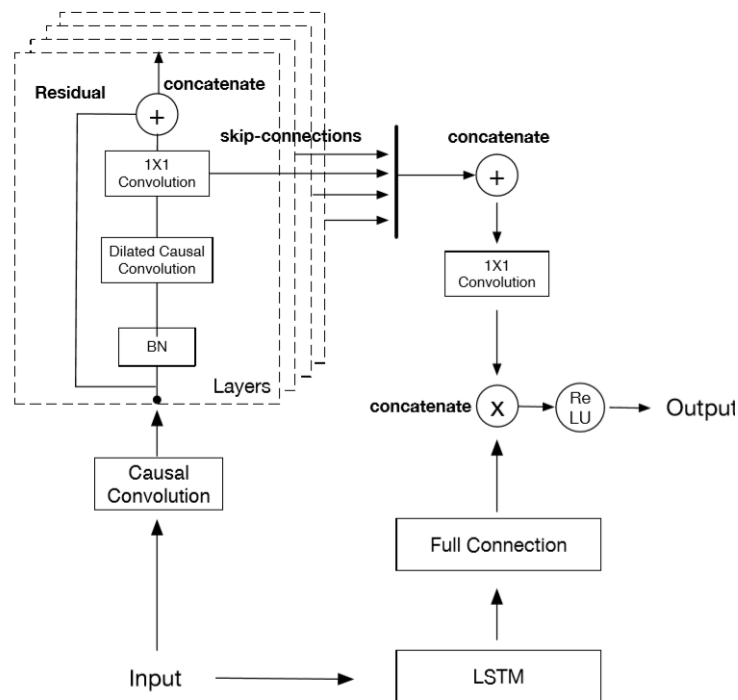


Figure 9 – SeriesNet Stock Forecasting Architecture (SHEN et al., 2019)

In Figure 9 we show a high level view of the SeriesNet model. The authors concatenated the output of a LSTM (HOCHREITER; SCHMIDHUBER, 1997) with a dilated causal convolution output, certainly inspired by WaveNet (OORD et al., 2016a) model. The authors also used a ReLU (MAAS; HANNUN; NG, 2013) activation function on model output and applied batch normalization (IOFFE; SZEGEDY, 2015) after causal

convolution output. They used the daily values from SP&500 and other indices as input for both LSTM and Causal Dilated Convolution networks.

3.5 SEE - An older but robust model for stock prediction

In our first work, called Structured event embeddings (SEE), we modelled the target price at time t (\hat{y}_t) as a linear combination of prices at times $t - 1, t - 2, \dots, t - n$ (that was, lag variables $y_{t-1}, y_{t-2}, \dots, y_{t-n}$). To help capture long-term temporal patterns, we also incorporated into the model quadratic and cubic time transformations (t^2 and t^3). We also represented the target price using categorical temporal attributes (month and quarter) and interaction variables associated with each lag variable ($(t - 1)y_{t-1}, (t - 2)y_{t-2}, \dots, (t - n)y_{t-n}$).

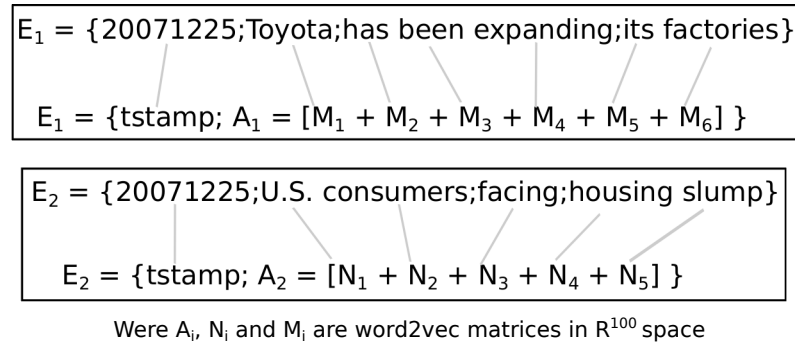


Figure 10 – Structured event embeddings (SEE) representation (NASCIMENTO; CRISTO, 2015)

In the original paper, the News information was represented as overlay data by summing the term vectors, as described in Figure 10. More specifically, in our last approach, the resulting 100-D vector embedding of timestamp t (that included all the structured events observed at t) was also used to represent the price at t . That method improved the time-series prediction results in about 10%, leading us to try more deep architectures to improve our results. (NASCIMENTO; CRISTO, 2015)

4 StockNet: Multivariate Deep Neural Architecture for stock prices prediction

In this section we describe our proposed model and its hyper-parameters.

4.1 Integrating News in a Convolutional Temporal Model

Based in Borovykh’s (BOROVYKH; BOHTE; OOSTERLEE, 2017) model we propose a novel convolutional causal architecture for stock prices prediction. This architecture will be trained to learn the next price for a particular stock and, at the same time, also learn how the news published between time interval t_{i-1} and t_i can contribute to improve the asset price prediction at time t_i .

4.1.1 Probabilistic View

The joint probability $p(x|\varphi)$ given a sequence of prices $x = [x_1, \dots, x_N]$ and the news semantic features sequence $\varphi = [\varphi_1, \dots, \varphi_N]$ give us the normalized predicted price as follows:

$$p(x|\varphi) \approx \prod_{t=1}^N p(x_t | x_1, \dots, x_{t-1}; \varphi_1, \dots, \varphi_{t-1}) \quad (4.1)$$

where x_t is the price at time interval t and φ_t is the semantic rankings features extracted from the news headlines published at time frame t .

To compute this probability we masked the inputs to make available only the past inputs using a causal conviction 1D for each input, one for price input x and another five convolutions for our semantic ranking inputs φ , which will be described in the next section.

We also proposed a new activation method called linear gated activation unit. Such method is similar to that used by gated PixelCNN (OORD et al., 2016b) architecture but with a rectified linear unit (MAAS; HANNUN; NG, 2013) instead of the sigmoid function. In our experiments, this replacement gave us similar results, but the model converged much faster, making training easier. So, formally,

$$z_k = \tanh(\mathbf{W}_{f,k} * (\mathbf{x} + \varphi)) \odot \text{ReLU}(\mathbf{W}_{g,k} * (\mathbf{x} + \varphi)) \quad (4.2)$$

where z_k is the causal dilated convolution output at layer k , $W_{f,k}$ and $W_{g,k}$ are the kernel weights at layer k , give us the normalized output z which will be used as main activation function in each layer of our convolutional neural network. Being u , the complete

computation for each layer is in the form:

$$u = z_0 * \mathbf{W}_{h,0} + \sum_{k=1}^L (z_k * \mathbf{W}_{h,k} + R_{k-1;k>0}) \quad (4.3)$$

where $W_{h,k}$ represents the kernel weights at layer k , L is the total number of layers in our model and R_{k-1} is the residual connection coming from the previous layer, the model output can be finally computed by:

$$output = Sigmoid(ReLU(\mathbf{W}_j * ReLU(u * \mathbf{W}_i))) \quad (4.4)$$

where W_i and W_j are the kernel weights at output convolutions.

4.1.2 Financial Semantic Field Ranking

Traditional sentiment analysis does not usually fit financial news context. This inadequacy occurs as consequence of the specific nature of the social networks in which such methods were born. For instance, an important financial headline about MSFT as “*Microsoft hits 1 trillion USD stock-price value*” can not be classified as “happy” or “sad” and is commonly classified as neutral by most sentiment analysis methods.

To overcome that limitation (LOUGHRAN; MCDONALD, 2011) proposed five “financial mood” lists. As an important contribution, they sought not only to map positive and negative words for financial news analysis, but also created lists of words commonly found in five situations: positive, negative, litigious, constraints and uncertain.

We propose a simple ranking technique to extract the semantic field from the news words using the financial sentiment lexicon proposed by (LOUGHRAN; MCDONALD, 2011). Our method outputs a probability distribution relative to the financial sentiment of the news input. This probabilistic distribution is later used as input to the neural network.

We also built five word embedding vector spaces based on these lists, also creating five different semantic vector spaces (or semantic fields). To feed our model, instead of using pure news word embeddings as input, we calculated the cosine distances from the five respective semantic spaces to each word present on news headlines. The resulting distribution is composed by these resulting spatial distances that are treated as five individual probability scores (pos_score, neg_score, lit_score, con_score and unc_score). We then feed them into the model during neural architecture training in a multivariate way. Our experiments demonstrated that this way of feeding the model is superior to the traditional method proposed by the baseline.

Formally, we can represent the semantic ranking φ_t for the news published at time interval t as a five-dimensional matrix where each element is represented by φ_t^k with $k \in \{pos, neg, lit, con, unc\}$, that are the five semantic fields labels. Therefore, we can

compute the particular semantic ranking vector φ_t^k using all words vectors w_i from the news headlines published at time t as follows:

$$\varphi_t^k = \frac{1}{n.m} \sum_{i=0}^n \sum_{j=0}^m \cos(w_i, u_j^k) \quad (4.5)$$

or using the vector form by:

$$\varphi_t^k = \frac{1}{n.m} \sum_{i=0}^n \sum_{j=0}^m \frac{w_i \cdot u_j^k}{\|w_i\| \cdot \|u_j^k\|} \quad (4.6)$$

where u_j^k is the word vector matrix for the semantic field k , and n is the number of word vectors in the news set published at time t and m is the number of word vectors in semantic field k .

For sure, we could directly rank these probabilities or perform an additional average to feed the model with a single discrete variable instead of using an output distribution, but we decided to use the dilated convolution layers for this task since the score in each semantic field might be useful to improve prices prediction. These new semantic scores are weighted during the training as an additional five-dimensional feature matrix that will later be propagated to the other layers of the model via dilated convolutions and residual connections.

4.1.3 High level architecture

As the main contribution of our work, we will make extensive experiments to figure out how much our proposed architecture can contribute to improve financial time-series price prediction in comparison to our baseline models and also suggest an effective way of integrating a raw news flow into temporal convolutional neural networks while optimize the entire architecture for predicting stocks using TCNs. Our final proposed architecture is detailed in Figure 11.

4.1.4 Hyper-parameters

The StockNet architecture has about 4,354,234 parameters. We used 20% dropout regularization (SRIVASTAVA et al., 2014) in the convolutional layer inputs and Xavier uniform initialization (GLOROT; BENGIO, 2010) for the weights of each neuron. As loss function, we use the Mean Squared Error function (MSE), formally defined in the equation 2.11.

To train the proposed neural architecture we used the Adam optimizer (KINGMA; BA, 2014) with the following Hyper-parameters: Learning rate: 0.001 and decay ra-

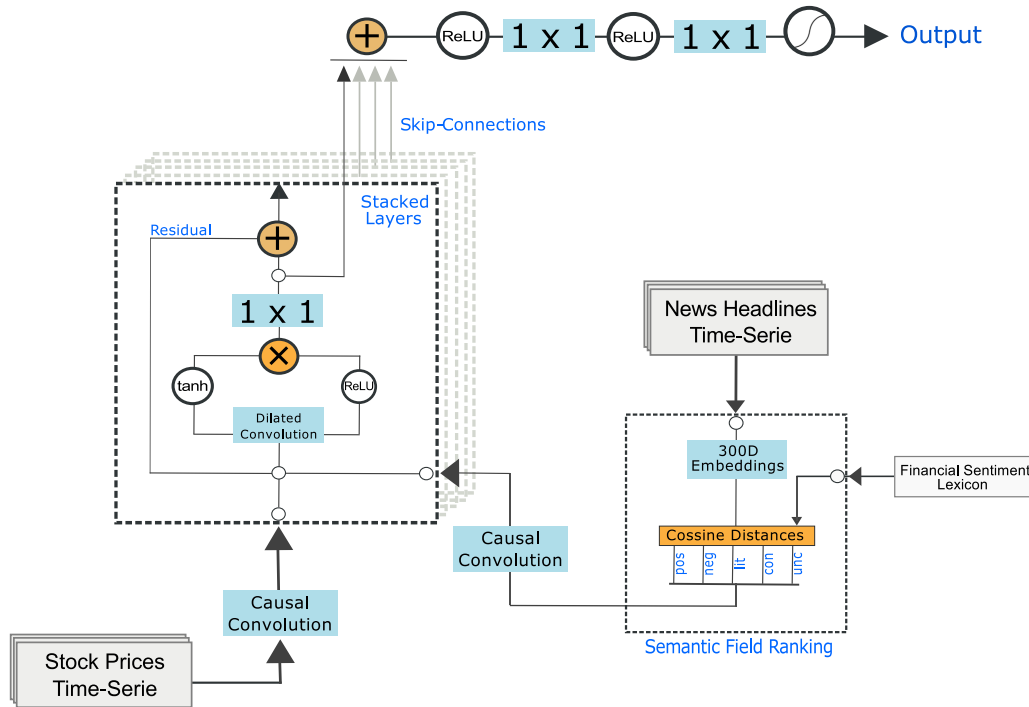


Figure 11 – StockNet architecture Overview

tio of zero, since these were the same optimizer and parameters used in our WaveNet implementation.

4.1.5 Model Objective

Unlike previous work, we are trying to predict the stock price rather than its polarity (whether the price will go up or down) as in a regression model. This objective needs a continuous function rather than a distribution as an output target. In the WaveNet model (OORD et al., 2016a), even for a regression task, the authors chose to use a distribution as output instead of continuous probability values. That was an affordable alternative since the output synthetic voice can be successfully represented as a quantizable signal in the time domain, allowing, in a reasonable computational time, the proposed 246-bit sampling of the signal amplitude.

It is also possible, in a similar way, to use Softmax for price prediction, simply by maximizing the probability of the output distribution. So, in that case, $Sigmoid(\hat{y})$ would give us the maximum likelihood price for a specific prediction \hat{y} . However, our model is intended to apply to assets described at totally different price ranges. For example, while AAPL stock track record has prices ranging from a dime to USD 250,00 GOOGL, a similar stock with similar business and related news, has a price range between USD 54.16 and USD 1,300.

Training a model capable of automatically normalize these price ranges or adapt to

scale changes while learning the importance of price-sensitive variations caused by financial news could largely increase training time. Additionally, since our model will predict only continuous values representing precise prices, we decide to propose an architecture with a Sigmoid function at the model output instead of an $\arg \max(\text{Softmax}(\hat{y}))$.

4.1.5.1 Price prediction and polarity forecasting

As mentioned in the related works section, the overwhelming majority of stock forecasting methods do not attempt to predict the exact price of assets at a future point in time, but rather whether the price will rise or fall. Such an approach is more straightforward since it converts the task into a binary classification problem, as well as using simpler metrics such as Precision, Recall or Matthew's correlation coefficient. Moreover, it is noticeably more difficult to use textual information with news in purely numerical auto-regressive models.

However, we disregard this approach in our proposed model, since we can argue that no rational operator initiates an operation without calculating its probable return, and the simple polarity prediction cannot indicate how much gain or loss an operation will return. Moreover, all operations in the real world have lateral costs, even those with free brokerage. Thus, often a small gain can result in losses at operation closing. In a bottom line, polarity-based methods are unable to calculate returns against real scenarios with operating costs, taxes and risks. our method is an accurate pricing model that can be used in real scenarios using only a time series and text as input.

5 Experiments

In this chapter, we present an overview of the adopted methodology applied in our experiments and also the preliminary and final results achieved by our experimented methods.

5.1 Stock Prices Dataset

Our first work (NASCIMENTO; CRISTO, 2015) used the publicly available daily S&P500 index. Now, we will use a more extensive 30 minute window stock prices dataset including the 10 more liquid stocks extracted from Dow Jones and NASDAQ from January, 2004 to August 2018.

This is a personal dataset that we have decided to make publicly available along with this work.¹

5.2 News Dataset

The experiments in our first work (NASCIMENTO; CRISTO, 2015) were made using the same dataset used by (DING et al., 2014) and (PENG; JIANG, 2015). This dataset consists of publicly available financial news from Reuters (106,521 documents) and Bloomberg (447,145 documents), gathered from October 2006 to November 2013, and stock market prices from Standard & Poor's 500 (S&P 500) index. It includes the international financial crisis that occurred in 2008, particularly important, since many stock operations appear to have been driven by overreaction, apparently caused by the huge amount of bad news about the market. All the news files included in this dataset are named with the day, month and year of each publication, since the publishers (DING et al., 2014) used the daily S&P500 index in their method.

We could not use a daily flow news dataset in conjunction with our financial time-series dataset composed by prices sampled every 30 minutes. Fortunately, the dataset used by (DING et al., 2014) also contained the publication date and time metadata inside each document. So, we rearranged the daily news series into a 30-minute news series, using the date information contained in news metadata. This allowed us to create a great 30-minute sliced news time-series for all headlines since October 2006 which we now published at github.¹

¹ Available at <https://github.com/hypernote/stocknet>

5.3 News Embeddings

There are several pre-trained skip-gram word embeddings available for us to use in our experiments, however we have selected a trained news template, since it fits best in financial scenarios. Furthermore embeddings trained over on generalist content bases such as Wikipedia corpus (WU; WELD, 2010) may have been really unsuitable for financial news data. So, in our experiments we used a previous trained embedding as input, trained over 100 billion words extracted from Google News. That embedding data-set contains 300-dimensional vectors for 3 million words and phrases. (MIKOLOV et al., 2013)

We specifically chose these particular word vectors since they presented interesting semantic results when queried about financial terms, including company names and stock codes. For instance, the summing of word vectors *miner + brazil + stock* returns, between the most near vectors, the name and stock code from the company Vale do Rio Doce S/A, the larger miner Brazilian company while summing the vectors *oil + brazil + company* returns the name and the stock code from Petrobras, the largest oil company in Brazil, as showed in Table 1. Each term in our headlines time-series dataset was mapped in a 300-d vector before being used as input by our model.

Vector Query 1: miner + brazil + stock	Vector Query 2: oil + brazil + company
giant Vale VALE5.SA	LKC interval n
Barrick ABX	producer OAO Rosneft
Tamaya Resources	Statoil ASA Norway
CVRD RIO	Holdings LKOH.RS
Namisa iron ore	Brazil Petrobras
Southern Copper PCU	Pemex PEMX.UL
VALE5.SA	Statoil Norway
Vale RIO	Petrobras PETR4.SA Quote
Freeport McMoRan FCX	Bolivia nationalizes
giant Vale VALE	Petroleos Mexicanos Pemex

Table 1 – Top 10 more similar vectors in trained embeddings given two different queries.

5.4 Baseline

Our objective is to outperform our first News based stock prediction model called SEE (NASCIMENTO; CRISTO, 2015) and the novel state of the art deep neural architecture SeriesNet₁₀ (SHEN et al., 2019). We also performed other experiments using alternative neural time-series prediction methods based on GRU (CHUNG et al., 2015) and WaveNet (OORD et al., 2016a).

5.5 Experiment Setup

We performed all the experiments of this work using C++ (STROUSTRUP, 2000) and Python (OLIPHANT, 2007) programming languages. All neural models were implemented using Keras (CHOLLET et al., 2015) and Tensorflow (ABADI et al., 2016) frameworks. We also heavily use numpy, pandas and matplotlib libraries, as well as various GNU environment tools. All experiments were made using a personal desktop computer equipped with the NvidiaTM GTX 1050 video card, running the Arch LinuxTM operational system.

5.6 Methodology

The prediction task consists in determining 10 highly liquid stocks listed in the S&P 500 index. The S&P 500, or the Standard & Poor’s 500, is a publicly available stock market index based on the market capitalization of 500 large companies having common stocks listed on the The New York Stock Exchange (NYSE) and in National Association of Securities Dealers Automated Quotations (NASDAQ).

We will compare the methods using two traditional loss metrics methods used in regression applications, that is, the Rooted Mean Squared Error (RMSE) and the Mean Absolute Percentage Error (MAPE). In a nutshell, RMSE and MAPE are given by Equations 5.1 and 5.2, respectively.

$$RMSE = \sqrt{\frac{1}{N} \sum_{t=1}^N (y_t - \hat{y}_t)^2} \quad (5.1)$$

$$MAPE = \frac{1}{N} \sum_{t=1}^N \left| \frac{y_t - \hat{y}_t}{y_t} \right| \quad (5.2)$$

where N is the number of instances, \hat{y} is the prediction made by the forecaster, and y is the correct value.

5.7 Preliminary Experiment (S&P500)

As a preliminary experiment, we tried to overcome the **SEE** and the state of the art architecture **SeriesNet**₁₀ on S&P 500 daily index prediction. We applied our main proposed architecture **StockNet** against that methods and also against six other alternative neural architectures we had proposed earlier during our research.

We first proposed two alternative architectures using well known time-series forecasting methods: the **GRU** (CHUNG et al., 2015) and the **WaveNet** (OORD et al.,

2016a) in their pure forms. In this first experimental phase, we simply applied a vanilla implementation of GRU and WaveNet to the daily prices time-series, without any textual news information, and output the results. The GRU, as the most recent version of Recurrent Neural Network (WILLIAMS; ZIPSER, 1989), seemed to us able to better capture the temporal dynamics observed on stock forecasting and the WaveNet is the first convolutional temporal network able to overcome recurrent architectures in sequence and time-series prediction problems.

The third proposed alternative architecture, which we call now **GRU+SEE** combine two GRU networks (CHUNG et al., 2015) for the input prices and news embeddings time-series and a traditional fully connected network(FCN) for the price output instead of the random forest based method proposed by the baseline. In fact, it seemed better to use the structure event features as input instead of just summing term-vectors as proposed by the baseline. We better describe this preliminary model GRU+SEE in Figure 12.

The fourth proposed alternative architecture, which we call now **GRU+RAW** also used a GRU (CHUNG et al., 2015) with raw embeddings as input, extracted directly from news text embeddings instead of SEE's structured events. We intent to solve the event sparsity problem in the daily news data-set and figure out if the model could infer other latent relationships and patterns hard to be detected by methods using open information extraction.

The fifth alternative architecture, which we call now **WN+SEE** used a multivariate version of the WaveNet architecture (OORD et al., 2016a) but with SEE's structured events as input. Therefore, instead of summing the term vectors and using them as input to a random forest, as proposed by the baseline, we feed the multivariate WaveNet with the structured event embeddings.

And the last proposed alternative architecture, which we call now **WN+RAW** also used a multivariate version of the WaveNet architecture (OORD et al., 2016a) but with raw embeddings as input, extracted directly from news text embeddings instead of SEE's structured events, in a similar way we had mad in GRU+RAW architecture.

5.7.1 Preliminary Results

As described in Table 2, in our preliminary results, SeriesNet₁₀ architecture and the vanilla models (GRU and WaveNet) do not overcome the SEE baseline, the third method (GRU+SEE), despite winning from the baseline, showed little significant differences. The raw text based models (GRU+RAW and WN+Raw) were unable to converge, maintaining high levels of validation error and vanishing gradients even after several parametric adjustments. Likely, the relative small amount of relevant news in the data-set when predicting generic S&P500 behavior and the apparent inability of the network to learn the

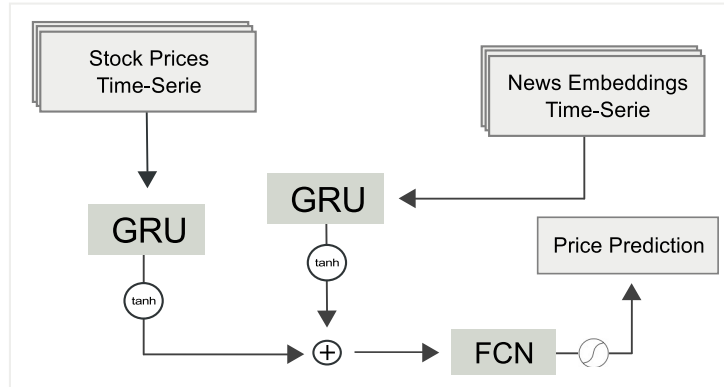


Figure 12 – Preliminary Experiment GRU+SEE with GRU Recurrent Architecture

Method	RMSE	MAPE
SEE (Baseline)	8,6430	0,5878
SeriesNet ₁₀ (Prices Only)	10,3464	0,6985
GRU (Prices Only)	9,8188	0,6493
Wavenet (Prices Only)	8,6203	0,5855
GRU+SEE	8,4151	0,5741
GRU+RAW	No Convergence	No Convergence
WN+SEE	7,9833	0,5542
WN+RAW	No Convergence	No Convergence
StockNet	7,4699	0,5190

Table 2 – StockNet results against baseline and alternative architectures for S&P500 (lower is better)

delta impact of news on stock price arises as the main bottlenecks.

The WN+SEE model that used structured events on a multivariate WaveNet performed well, surpassing the baseline by more than 7% and finally, our better model, the StockNet, surpassed the baseline by over 13%.

5.8 StockNet Intraday Extensive Experiments

Even a large financial news dataset, containing between 2006~2013, became small on a daily granularity, especially for neural networks training. From a daily perspective, the original news dataset has only 2,609 days (samples). Neural networks perform better when a huge amount of data is available for training.

Thus, we performed a new experiment using our StockNet architecture against the second best method in our previous experiment using a shorter time frame, with samples taken every 30 minutes. We also used individual stocks rather than S&P500 index. These seem to increase our model's advantage over that baseline.

5.8.1 Temporal News Subsampling

Our final intraday experiment tried to avoid some drawbacks with low input size and the small time-frame problems reported in our preliminary results. We note that, even using this 30-minute fashion, a total of 21,330 timesteps still have no more than 1 headline published. This is more than 26% of the entire news flow. In order to mitigate this imbalance, we reduced the number of headlines per timestep to maximum of 20 in the whole dataset, selecting the headlines in a random way on those timesteps to create a subsampled timeseries. A published news histogram after this intraday splitting procedure is shown in Figure 13.

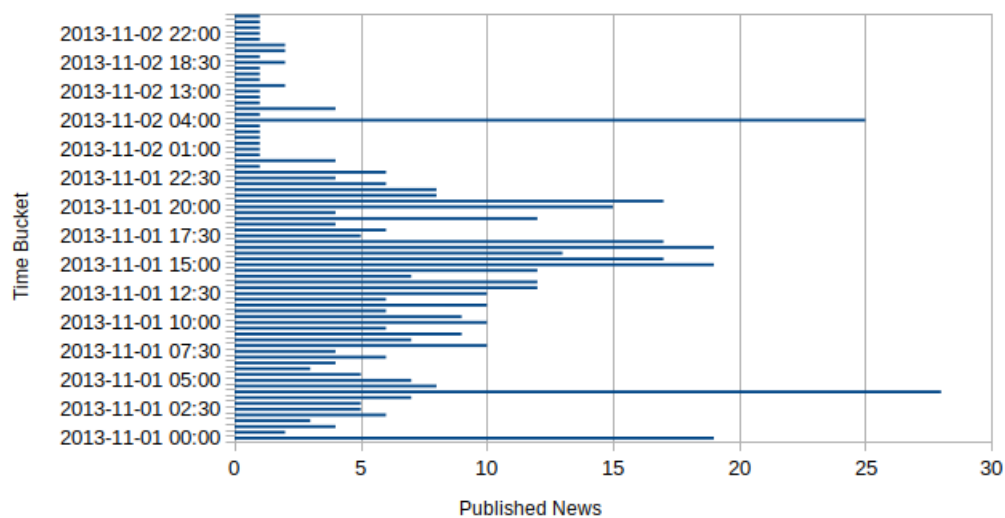


Figure 13 – Intraday news histogram with 30-minute time splitting

We note in Figure 13 that the intraday splitting procedure evidenced a huge imbalance in the number of published news at each time bucket. For instance, while there is only a unique new published between 9:00PM and 9:15PM in 01/03/2006, there is 123 news published between 07:30AM and 07:45AM in 06/08/2012. In the whole dataset, we found an average number of 6.76 news published by time step (after removing the gaps with no news published) in this 30 minute time frame.

5.8.2 Individual Stocks instead of Market Indexes

As explained in the section about our price dataset, another major modification was use individual stock time-series rather than market indexes in our final experiment, since we have inferred that particular important news about Microsoft Company could strongly affect MSFT stock prices, but their importance will be strongly diluted in the S&P500 index.

5.8.3 Final Results

In the Table 3 below we show the final results obtained from our new multivariate architecture (StockNet) against our last model (NASCIMENTO; CRISTO, 2015) for ten highly liquid stocks listed at NYSE and NASDAQ:

Stock	Market Name	SEE RMSE	SEE MAPE	StockNet RMSE	StockNet MAPE
AAPL	Apple Inc	0.8105	1.1766	0.1465	0.1869
AMZN	Amazon.com, Inc	8.5422	2.3042	2.3961	0.5473
CSCO	Cisco Systems, Inc	0.1005	0.4937	0.4434	1.9754
F	Ford Motor Company	0.1466	1.1261	0.0732	0.5518
IBM	IBM Common Stock	1.1556	0.7382	2.6192	1.6593
JPM	JPMorgan Chase & Co	0.4044	0.8165	0.2844	0.5726
MSFT	Microsoft Corporation	0.2861	0.8581	0.1302	0.3616
NFLX	Netflix, Inc	1.8634	3.8745	0.5022	0.9789
WMT	Walmart Stores, Inc	0.1507	0.1883	0.1203	0.1274
XOM	Exxon Mobil Corporation	0.3919	0.4812	0.1836	0.2127
Average Error		1.38519	1.20574	0.68991	0.71739

Table 3 – StockNet results against baseline (lower is better)

As described in Table 3, for a 30-minute sampling and using individual stocks (which is much closer to a real-world application), our proposed model considerably increased the gains against baseline, reaching an average advantage of almost 50% on the most liquid stocks, but did not do better than baseline for Cisco Systems and IBM stocks.

To better understand the cases where our model was underperformed, we now plot in Figure 14 the prediction for IBM Inc. We can see an abrupt change in the direction of the share price in the previous steps. The previous model (SEE) was able to capture this change almost immediately, while StockNet seems to have suffered a delay or considered other information, more distant in past time. In other words, the new model seemed to “disbelieve” that price direction would experiment such a drastic change.

A similar behavior was observed in Figure 15, in the case of Cisco Systems Inc. We note a drastic change in the direction of the prices near to the prediction. SEE seems a better model for these particular cases, since it clearly reached low RMSE and MAPE values.

In both cases where Stocknet underperformed, prices changed sharply. We then wondered if this occurred whenever there were sudden movements. Thus, we ran new experiments using other highly liquid stocks (out of our initial dataset) with unexpected market movements and obtained similar results.

Extreme and unexpected market movements are also known as Black Swans. Figure 16 shows the results for Verizon. It is possible note that its expected direction changes sharply and in that scenario we also find StockNet’s behavior worse than the

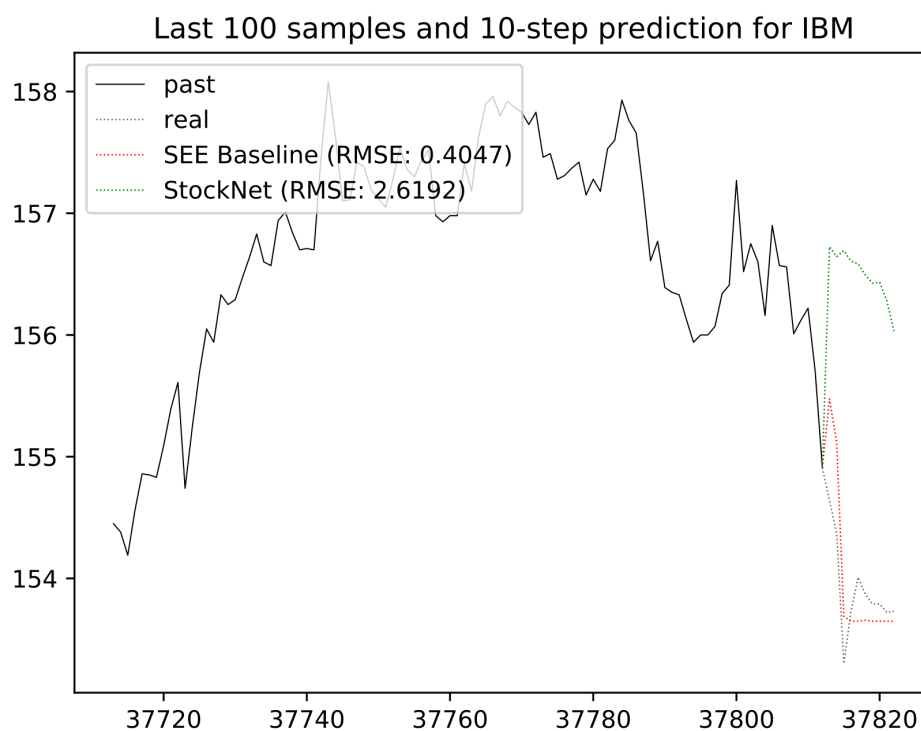


Figure 14 – Results for IBM

baseline. Such evidence may lead us to hypothesize that the SEE method only overcomes StockNet in Black Swan events, but we could not support this hypothesis without a future systematic study.

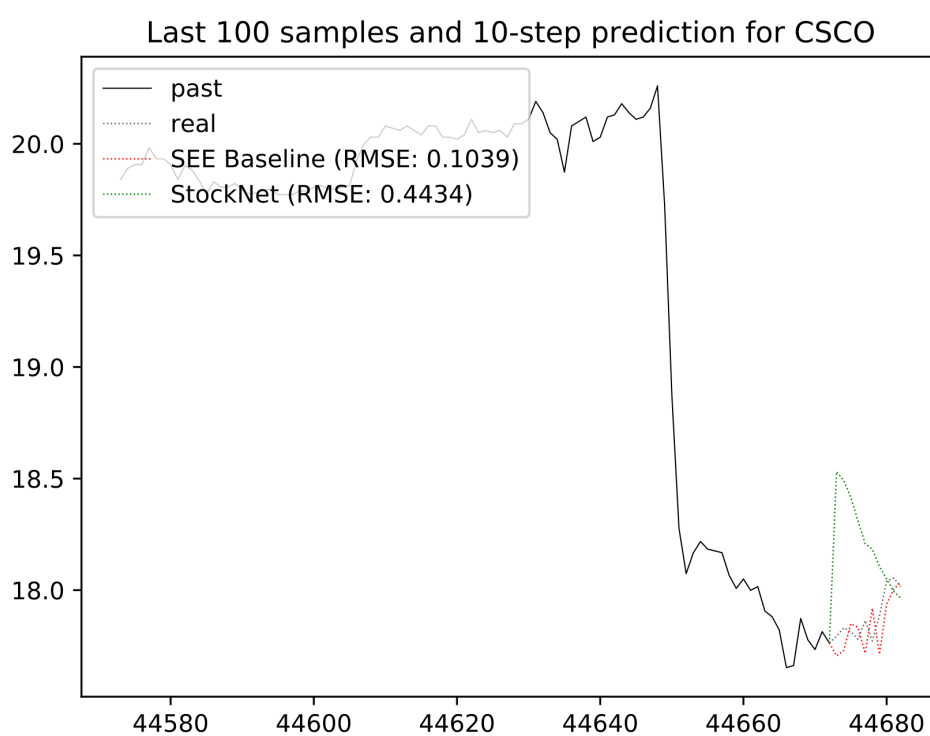


Figure 15 – Results for Cisco Systems

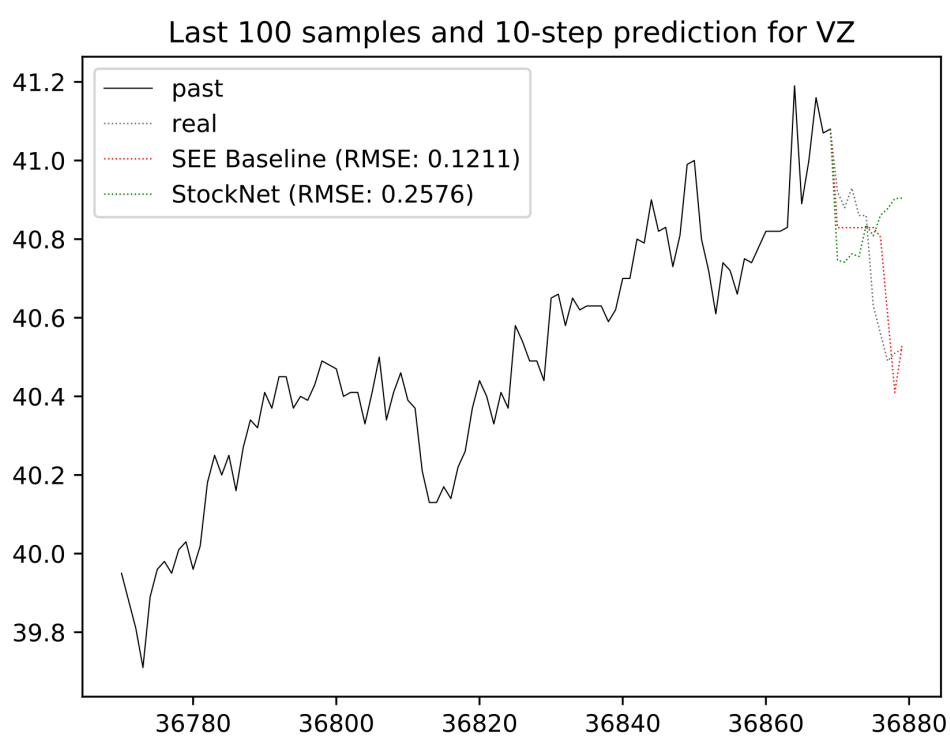


Figure 16 – Results for Verizon (not in initial dataset)

6 Conclusions

In the present work we propose StockNet, a novel state of the art architecture for stock prices prediction. In addition to overcoming the intended baseline, StockNet was also able to surpass the state of the art method by about 27 of RMSE on the S&P500 index forecasting task and almost 50% of RMSE in the individual company price prediction scenario.

StockNet is a novel robust and highly precise forecasting architecture for non-stationary financial time series in scenarios with sufficient news information. It has demonstrably superior performance for both index prediction and price prediction in real-world situations (huge number of news and high sample rates) while needs reduced attribute engineering, making it a more straightforward neural method for real financial forecasting. However, it has shown some limitations, in particular, associated with Black Swan events.

6.1 Future Work

Future investigations can assess the impact of StockNet’s lower forecast accuracy on black swan scenarios in comparison with other models and also experiment it in low liquidity stocks scenarios or that where financial news about these assets are scarce.

A larger dataset with more news and more information sources can also be experimented in the StockNet input. The architecture hyper-parameters can also be handled. For instance, one can change the number of 1D-conv layers or experiment alternative types of activation functions as the recent proposed Gaussian Error Linear Units (HENDRYCKS; GIMPEL, 2016) instead of ReLU.

In the model hyper parameters, we also would experiment different forms of weight initialization for each layer, apply other regularization methods and even use other types of symmetric and asymmetric loss functions.

In the news text model input, we use a pre-trained embedding learned with the Skip-gram algorithm. Other newer embedding techniques like FastText (JOULIN et al., 2016) or even the more recent BERT (DEVLIN et al., 2018) can be experimented in order to extract word vectors from the news flow.

Recently a new more comprehensive lexicon for sentiment analysis in financial news has been proposed (SARDELICH; KAZAKOV, 2018) in addition to that proposed by (LOUGHRAN; MCDONALD, 2011). A future work can repeat our experiments using this lexicon as input to the semantic ranking and evaluate if gains are obtained in RMSE or

MAPE.

This work was initially inspired by WaveNet results as a deep acoustic model for speech synthesis. New deep acoustic generative models such as Tacotron (WANG et al., 2017) or SV2TTS (JIA et al., 2018) can be experimented on stock prediction instead of pure Wavenet in order to evaluate its results.

StockNet model used a pre-trained embedding model, trained on a corpus extracted from Google News. Such a corpus is made up of news of all kinds. We consider it important to conduct future experiments using a new embedding model trained on a specific corpus composed only of financial news.

In our final experiment, we used stocks from companies from different industrial fields and extracted news related to those stocks. New experiments can be made using as input only news headlines extracted from the sector of activity of the target company. For example, we could include news headlines related to the oil market into the news dataset of an oil company.

Along with the textual input extracted from the news, we used only the stock prices time series. This decision was intended to simplify the model, but may have contributed to the inappropriate behavior of our architecture in Black Swan scenarios. Future experiments may consider the use of volume time series, which is often used by graphic financial analysts to anticipate sudden movements and market manipulation.

New experiments could also study news selection techniques, take into account their qualitative aspects such as importance, freshness, diversification, publisher importance and correlation with the target asset.

References

- ABADI, M. et al. Tensorflow: a system for large-scale machine learning. In: *OSDI*. [S.l.: s.n.], 2016. v. 16, p. 265–283. Cited on page 39.
- ABIRAMI, A.; GAYATHRI, V. A survey on sentiment analysis methods and approach. In: IEEE. *Advanced Computing (ICoAC), 2016 Eighth International Conference on*. [S.l.], 2017. p. 72–76. Cited on page 29.
- ALDRIDGE, I. *A Practical Guide to Algorithmic Strategies and Trading Systems-High Frequency Trading*. [S.l.]: John Wiley & Sons, Inc, 2010. Cited on page 4.
- ANGELI, G.; PREMKUMAR, M. J. J.; MANNING, C. D. Leveraging linguistic structure for open domain information extraction. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. [S.l.: s.n.], 2015. p. 344–354. Cited 3 times on pages iii, 13, and 14.
- ASHBY, W. *Design for a brain: The origin of adaptive behaviour*. [S.l.]: Springer Science & Business Media, 1952. Cited on page 17.
- BAI, S.; KOLTER, J. Z.; KOLTUN, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *CoRR*, abs/1803.01271, 2018. Disponible em: <<http://arxiv.org/abs/1803.01271>>. Cited 4 times on pages 5, 23, 24, and 29.
- BANKO, M. et al. Open information extraction for the web. In: *IJCAI*. [S.l.: s.n.], 2007. v. 7, p. 2670–2676. Cited 3 times on pages 13, 14, and 15.
- BENGIO, Y. et al. A neural probabilistic language model. *The Journal of Machine Learning Research*, JMLR. org, v. 3, p. 1137–1155, 2003. Cited on page 28.
- BOLLEN, J.; MAO, H.; ZENG, X. Twitter mood predicts the stock market. *Journal of Computational Science*, Elsevier, v. 2, n. 1, p. 1–8, 2011. Cited 4 times on pages 4, 5, 16, and 28.
- BONDT, W. F. D.; THALER, R. Does the stock market overreact? *The Journal of finance*, Wiley Online Library, v. 40, n. 3, p. 793–805, 1985. Cited on page 4.
- BOROVYKH, A.; BOHTE, S.; OOSTERLEE, C. W. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*, 2017. Cited 4 times on pages 24, 25, 29, and 32.
- BROCKWELL, P. J.; DAVIS, R. A.; CALDER, M. V. *Introduction to time series and forecasting*. [S.l.]: Springer, 2002. v. 2. Cited on page 7.
- CAUCHY, A. Méthode générale pour la résolution des systemes d'équations simultanées. *Comptes rendus hebdomadaires des séances de l'Académie des sciences, Paris*, v. 25, n. 1847, p. 536–538, 1847. Cited on page 20.
- CHOLLET, F. et al. Keras: Deep learning library for theano and tensorflow.(2015). *There is no corresponding record for this reference*, 2015. Cited on page 39.

- CHUNG, J. et al. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014. Disponível em: <<http://arxiv.org/abs/1412.3555>>. Cited 2 times on pages iii and 22.
- CHUNG, J. et al. Gated feedback recurrent neural networks. *CoRR*, abs/1502.02367, 2015. Cited 6 times on pages 5, 22, 29, 38, 39, and 40.
- CORRO, L. D.; GEMULLA, R. Clausie: clause-based open information extraction. In: ACM. *Proceedings of the 22nd international conference on World Wide Web*. [S.l.], 2013. p. 355–366. Cited on page 13.
- DAS, S. et al. Real-time sentiment analysis of twitter streaming data for stock prediction. *Procedia computer science*, Elsevier, v. 132, p. 956–964, 2018. Cited on page 28.
- DEERWESTER, S. et al. Indexing by latent semantic analysis. *Journal of the American society for information science*, Wiley Online Library, v. 41, n. 6, p. 391–407, 1990. Cited on page 10.
- DEVLIN, J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. Cited on page 47.
- DIETTERICH, T. G. Ensemble methods in machine learning. In: SPRINGER. *International workshop on multiple classifier systems*. [S.l.], 2000. p. 1–15. Cited on page 28.
- DING, X. et al. Using structured events to predict stock price movement: An empirical investigation. 2014. Cited 4 times on pages 5, 15, 28, and 37.
- DING, X. et al. Deep learning for event-driven stock prediction. 2015. Cited on page 14.
- DUDA, R.; SINGLETON, R. *Training a threshold logic unit with imperfectly classified patterns*. [S.l.], 1944. Cited on page 17.
- ETZIONI, O. et al. Open information extraction: The second generation. In: *IJCAI*. [S.l.: s.n.], 2011. v. 11, p. 3–10. Cited 2 times on pages 13 and 15.
- FAMA, E. F. The behavior of stock-market prices. *Journal of business*, JSTOR, p. 34–105, 1965. Cited on page 4.
- FANG, X.; ZHAN, J. Sentiment analysis using product review data. *Journal of Big Data*, Nature Publishing Group, v. 2, n. 1, p. 5, 2015. Cited on page 15.
- FENG, F. et al. Improving stock movement prediction with adversarial training. *arXiv preprint arXiv:1810.09936*, 2018. Cited on page 30.
- FU, T.-c. A review on time series data mining. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 24, n. 1, p. 164–181, 2011. Cited on page 28.
- GAMBOA, J. C. B. Deep learning for time-series analysis. *arXiv preprint arXiv:1701.01887*, 2017. Cited on page 29.
- GIDOFALVI, G.; ELKAN, C. Using news articles to predict stock price movements. *Department of Computer Science and Engineering, University of California, San Diego*, 2001. Cited on page 4.

- GILBERT, C. H. E. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In: *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*. Available at (20/04/16) <http://comp.social.gatech.edu/papers/icwsm14.vader.hutto.pdf>. [S.l.: s.n.], 2014. Cited on page 29.
- GLOROT, X.; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. In: *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10)*. Society for Artificial Intelligence and Statistics. [S.l.: s.n.], 2010. Cited 2 times on pages 21 and 34.
- GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <<http://www.deeplearningbook.org>>. Cited on page 20.
- GRANGER, C. W. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: Journal of the Econometric Society*, JSTOR, p. 424–438, 1969. Cited on page 19.
- HAMILTON, J. D. *Time series analysis*. [S.l.]: Princeton university press Princeton, NJ, 1994. v. 2. Cited on page 28.
- HARRIS, Z. Distributional structure. *Word*, v. 10, n. 23, p. 146–162, 1954. Cited on page 10.
- HAYEK, F. A. *The sensory order: An inquiry into the foundations of theoretical psychology*. [S.l.]: University of Chicago Press, 1952. Cited on page 17.
- HE, K. et al. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. Disponível em: <<http://arxiv.org/abs/1512.03385>>. Cited on page 23.
- HE, K. et al. Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2016. p. 770–778. Cited 3 times on pages 21, 23, and 26.
- HEARST, M. A. et al. Support vector machines. *Intelligent Systems and their Applications, IEEE, IEEE*, v. 13, n. 4, p. 18–28, 1998. Cited on page 28.
- HENDRYCKS, D.; GIMPEL, K. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415, 2016. Disponível em: <<http://arxiv.org/abs/1606.08415>>. Cited on page 47.
- HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997. Cited 4 times on pages 5, 22, 29, and 30.
- HONG, Y.; LEE, T.-H. Inference on predictability of foreign exchange rates via generalized spectrum and nonlinear time series models. *Review of Economics and Statistics*, MIT Press, v. 85, n. 4, p. 1048–1062, 2003. Cited on page 19.
- HOPFIELD, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, National Acad Sciences, v. 79, n. 8, p. 2554–2558, 1982. Cited 2 times on pages 5 and 22.
- HU, X. et al. Sinet: A scale-insensitive convolutional neural network for fast vehicle detection. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, v. 20, n. 3, p. 1010–1019, 2018. Cited on page 21.

HU, Z. et al. Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In: *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. New York, NY, USA: ACM, 2018. (WSDM '18), p. 261–269. ISBN 978-1-4503-5581-0. Disponível em: <<http://doi.acm.org/10.1145/3159652.3159690>>. Cited on page 30.

HUANG, Y. et al. Boosting financial trend prediction with twitter mood based on selective hidden markov models. In: SPRINGER. *International Conference on Database Systems for Advanced Applications*. [S.l.], 2015. p. 435–451. Cited on page 28.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. Disponível em: <<http://arxiv.org/abs/1502.03167>>. Cited on page 30.

JIA, Y. et al. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2018. p. 4480–4490. Cited on page 48.

JOULIN, A. et al. Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*, 2016. Cited on page 47.

KAHNEMAN, D.; TVERSKY, A. On the interpretation of intuitive probability: A reply to jonathan cohen. Elsevier Science, 1979. Cited on page 4.

KEANE, M. P.; RUNKLE, D. E. Are financial analysts' forecasts of corporate profits rational? *Journal of Political Economy*, The University of Chicago Press, v. 106, n. 4, p. 768–805, 1998. Cited on page 19.

KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. Cited on page 34.

KNOTT, J. R. *The organization of behavior: A neuropsychological theory: DO Hebb, Ph. D. John Willey & Sons, New York, 1949, viii+ 335 pp., 4.00*. [S.l.]: Elsevier, 1951. Cited on page 17.

LEIBNIZ, G. *Nova methodus pro maximis et minimis itemque tangentibus*. [S.l.: s.n.], 1684. Cited on page 20.

L'HOPITAL, G. F. A. Marquis de. *Analyse des infiniment petits pour l'intelligence des lignes courbes*. [S.l.: s.n.], 1696. Cited on page 20.

LIANG, X. et al. Lstm with wavelet transform based data preprocessing for stock price prediction. *Mathematical Problems in Engineering*, Hindawi, v. 2019, 2019. Cited on page 28.

LIU, H. Leveraging financial news for stock trend prediction with attention-based recurrent neural network. *arXiv preprint arXiv:1811.06173*, 2018. Cited on page 29.

LONG, W.; LU, Z.; CUI, L. Deep learning-based feature engineering for stock price movement prediction. *Knowledge-Based Systems*, Elsevier, v. 164, p. 163–173, 2019. Cited on page 30.

- LOUGHRAN, T.; MCDONALD, B. When is a liability not a liability? textual analysis, dictionaries, and 10-ks. *The Journal of Finance*, Wiley Online Library, v. 66, n. 1, p. 35–65, 2011. Cited 3 times on pages 16, 33, and 47.
- MAAS, A. L.; HANNUN, A. Y.; NG, A. Y. Rectifier nonlinearities improve neural network acoustic models. In: *Proc. icml*. [S.l.: s.n.], 2013. v. 30, n. 1, p. 3. Cited 4 times on pages 18, 21, 30, and 32.
- MAUSAM et al. Open language learning for information extraction. In: *Proceedings of Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CONLL)*. [S.l.: s.n.], 2012. Cited on page 13.
- MIKOLOV, T. et al. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. Cited 4 times on pages 11, 14, 29, and 38.
- MIKOLOV, T.; YIH, W.-t.; ZWEIG, G. Linguistic regularities in continuous space word representations. In: *HLT-NAACL*. [S.l.: s.n.], 2013. p. 746–751. Cited on page 12.
- MINSKY, M.; PAPERT, S. *Perceptrons*. Cambridge, MA: MIT Press, 1969. Cited on page 18.
- MITTAL, A.; GOEL, A. Stock prediction using twitter sentiment analysis. *Stanford University, CS229 (2011 <http://cs229.stanford.edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis.pdf>)*, v. 15, 2012. Cited on page 4.
- MITTELMAN, R. Time-series modeling with undecimated fully convolutional neural networks. *arXiv preprint arXiv:1508.00317*, 2015. Cited 2 times on pages 24 and 29.
- NAIR, V.; HINTON, G. E. Rectified linear units improve restricted boltzmann machines. In: *Proceedings of the 27th International Conference on International Conference on Machine Learning*. USA: Omnipress, 2010. (ICML'10), p. 807–814. ISBN 978-1-60558-907-7. Disponível em: <<http://dl.acm.org/citation.cfm?id=3104322.3104425>>. Cited 2 times on pages 18 and 21.
- NASCIMENTO, J. B.; CRISTO, M. The impact of structured event embeddings on scalable stock forecasting models. In: ACM. *Proceedings of the 21st Brazilian Symposium on Multimedia and the Web*. [S.l.], 2015. p. 121–124. Cited 8 times on pages iii, 5, 14, 15, 31, 37, 38, and 43.
- NASSIRTOUSSI, A. K. et al. Text mining of news-headlines for forex market prediction: A multi-layer dimension reduction algorithm with semantics and sentiment. *Expert Systems with Applications*, Elsevier, v. 42, n. 1, p. 306–324, 2015. Cited on page 4.
- NEUMANN, J. von. The general and logical theory of automata. In: JEFFRESS, L. A. (Ed.). *Cerebral Mechanisms in Behaviour*. [S.l.]: Wiley, 1951. Cited on page 16.
- OLIPHANT, T. E. Python for scientific computing. *Computing in Science & Engineering*, IEEE, v. 9, n. 3, 2007. Cited on page 39.
- OLIVEIRA, N.; CORTEZ, P.; AREAL, N. The impact of microblogging data for stock market prediction: using twitter to predict returns, volatility, trading volume and survey sentiment indices. *Expert Systems with Applications*, Elsevier, v. 73, p. 125–144, 2017. Cited on page 4.

OORD, A. V. D. et al. Wavenet: A generative model for raw audio. In: *SSW*. [S.l.: s.n.], 2016. p. 125. Cited 11 times on pages iii, 5, 23, 25, 26, 27, 29, 30, 35, 38, and 40.

OORD, A. van den et al. Conditional image generation with pixelcnn decoders. *CoRR*, abs/1606.05328, 2016. Disponível em: <<http://arxiv.org/abs/1606.05328>>. Cited on page 32.

PENG, Y.; JIANG, H. Leverage financial news to predict stock price movements using word embeddings and deep neural networks. *arXiv preprint arXiv:1506.07220*, 2015. Cited 2 times on pages 28 and 37.

PROSKY, J. et al. Sentiment predictability for stocks. *arXiv preprint arXiv:1712.05785*, 2017. Cited on page 29.

RENÉ, C. L. M. D. F.; HAGER, V. A. R. G. D. Temporal convolutional networks for action segmentation and detection. In: *IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2017. Cited 3 times on pages 5, 23, and 29.

ROBERTS, H. V. Statistical versus clinical prediction of the stock market. 1967. Cited on page 4.

ROJAS, R. *Neural networks: a systematic introduction*. [S.l.]: Springer Science & Business Media, 2013. Cited on page 28.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958. Cited on page 17.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. *Learning internal representations by error propagation*. [S.l.], 1985. Cited 2 times on pages 18 and 20.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning Representations by Back-propagating Errors. *Nature*, v. 323, n. 6088, p. 533–536, 1986. Disponível em: <<http://www.nature.com/articles/323533a0>>. Cited on page 10.

RUTECKI, P. A. Neuronal excitability: voltage-dependent currents and synaptic transmission. *Journal of clinical neurophysiology: official publication of the American Electroencephalographic Society*, v. 9, n. 2, p. 195–211, 1992. Cited on page 17.

SARDELICH, M.; KAZAKOV, D. Extending the loughran and mcdonald financial sentiment words list from 10-k corporate filings using social media texts. In: POPESCU, O.; STRAPPARAVA, C. (Ed.). *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Paris, France: European Language Resources Association (ELRA), 2018. ISBN 979-10-95546-11-5. Cited on page 47.

SHEN, Z. et al. A novel time series forecasting model with deep learning. *Neurocomputing*, Elsevier, 2019. Cited 3 times on pages iii, 30, and 38.

SINGH, B. K. Loss functions in financial sector: An overview. *Asian Journal of Mathematics & Statistics*, Asian Network for Scientific Information (ANSINET), v. 8, n. 1, p. 35, 2015. Cited on page 19.

SONG, Y.-G.; ZHOU, Y.-L.; HAN, R.-J. Neural networks for stock price prediction. *arXiv preprint arXiv:1805.11317*, 2018. Cited on page 28.

- SRIVASTAVA, N. et al. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, JMLR. org, v. 15, n. 1, p. 1929–1958, 2014. Cited on page 34.
- STROUSTRUP, B. *The C++ programming language*. [S.l.]: Pearson Education India, 2000. Cited on page 39.
- SUNDERMEYER, M.; SCHLÜTER, R.; NEY, H. Lstm neural networks for language modeling. In: *Interspeech*. [S.l.: s.n.], 2012. p. 194–197. Cited on page 29.
- TAI, Y.; YANG, J.; LIU, X. Image super-resolution via deep recursive residual network. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2017. p. 3147–3155. Cited on page 23.
- UTTLEY, A. M. Conditional probability machines and conditioned reflexes. *Automata Studies*, "CE Shannon and J. McCarthy, eds., Princeton University Press. Princeton, NJ, *Annals of Mathematics Study*, p. 253–275, 1956. Cited on page 17.
- WANG, Y. et al. Tacotron: Towards end-to-end speech synthesis. *arXiv preprint arXiv:1703.10135*, 2017. Cited 2 times on pages 23 and 48.
- WERBOS, P. New tools for prediction and analysis in the behavioral sciences. *Ph. D. dissertation, Harvard University*, 1974. Cited on page 20.
- WILLIAMS, R. J.; ZIPSER, D. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, MIT Press, v. 1, n. 2, p. 270–280, 1989. Cited on page 40.
- WU, F.; WELD, D. S. Open information extraction using wikipedia. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 48th annual meeting of the association for computational linguistics*. [S.l.], 2010. p. 118–127. Cited on page 38.
- XING, F. Z.; CAMBRIA, E.; WELSCH, R. E. Natural language based financial forecasting: a survey. *Artificial Intelligence Review*, Springer, v. 50, n. 1, p. 49–73, 2018. Cited on page 29.
- YU, P.; YAN, X. Stock price prediction based on deep neural networks. *Neural Computing and Applications*, Springer, p. 1–20, 2019. Cited on page 29.
- YU, S.; KAK, S. A survey of prediction using social media. *arXiv preprint arXiv:1203.1647*, 2012. Cited on page 28.
- ZHOU, X. et al. Stock market prediction on high-frequency data using generative adversarial nets. *Mathematical Problems in Engineering*, Hindawi, v. 2018, 2018. Cited on page 4.