Optimized FOREX Trading System Based on Empirical Mode Decomposition and Neural Networks

A Thesis

submitted to the designated

by the Assembly

of the Department of Computer Science and Engineering

Examination Committee

by

Michail Papatsimpas

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN DATA AND COMPUTER SYSTEMS ENGINEERING

WITH SPECIALIZATION

IN DATA SCIENCE AND ENGINEERING

University of Ioannina

School of Engineering

Ioannina 2022

Examining Committee:

- **Konstantinos E. Parsopoulos**, Professor, Department of Computer Science and Engineering, University of Ioannina (Advisor)
- **Lisimachos Pavlos Kondis**, Professor, Department of Computer Science and Engineering, University of Ioannina
- **Konstantina Skouri**, Professor, Department of Mathematics, University of Ioannina

Dedication

I dedicate this thesis to my family.

Acknowledgements

I would like to express my deep and sincere thanks to my supervisor Professor Konstantinos Parsopoulos for all the support and guidance during all these years. He was more than eager to provide me with his knowledge on the subject while making helpful suggestions all over my work. His guidance, support, and expertise have been only some of the ways through which he has shaped my first steps in research.

Next, I would like to thank my parents Alexandra and Giorgos, who have taught me to never give up on my dreams. Also, I would like to thank my older sister and brother, Nadia and Alexandros, who have always been there for me. As far as I remember myself, Nadia has always been by my side, so I would not have made it this far without her help all these years. Alexandros has also played a crucial role in this "journey". Specifically, I want to thank him for all the overnight discussions and arguments we had about trading. I first heard the word "trading" from him, since he is in the trading industry for more than a decade, and I remember him telling me how my computer science background could be combined with the financial industry. During all this time, he was always there to advise me with his expertise in the field.

Finally, I want to thank my old friend Giorgos Emmanouilidis for his unconditional support and all the scientific discussions we had about Machine Learning and its applications.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Glossary

ADF	Augmented Dickey-Fuller
AI	Artificial Intelligence
AM	Amplitude Modulation
ANN	Artificial Neural Network
CEC	Constant Error Carousel
DL	Deep Learning
EMD	Empirical Mode Decomposition
FOREX	Foreign Exchange
IMF	Intrinsic Mode Function
LSTM	Long-Short Memory
ML	Machine Learning
MDD	Maximum DrawDown
NLP	Natural Language Processing
RNN	Recurrent Neural Network
BPTT	Backpropagation Through Time

Abstract

Michail Papatsimpas, M.Sc. in Data Science and Engineering, Department of Computer Science and Engineering, School of Engineering, University of Ioannina, Greece, 2022. Optimized FOREX Trading System Based on Empirical Mode Decomposition and Neural Networks.
Advisor: Konstantinos Parsopoulos, Professor

The financial market is a complex and dynamical system, which is influenced by many factors subject to uncertainty. In order to raise the limited chances of beating the market, investors usually rely on diverse techniques that attempt to determine the underlying trading signal, and hopefully predict future market entry and exit points.

The present thesis proposes a new trading system for the Foreign Exchange Market (FOREX). The system consists of a hybrid algorithm that combines empirical mode decomposition (EMD), long short-term memory neural network (LSTM) and particle swarm optimization algorithm (PSO) (i.e., EMD-LSTM-PSO) to develop a prediction model for the daily closing prices of exchange rates. The EMD method is employed to decompose the initial signal to several intrinsic mode functions (IMFs) and residual. Then, for each IMF an LSTM neural network is constructed, and the final forecast is the aggregation of all the forecasts produced by the LSTM neural networks, optimized by the PSO algorithm. We measured our system's performance against three currency pairs, namely EUR/USD, EUR/CHF and USD/CHF. Numerical testing demonstrates that the EMD-LSTM-PSO method can accurately predict the three currency pairs.

The proposed system is fully automated, and it is able to handle all the trading requests (open/close positions), using the MetaTrader platform. In addition, it notifies the investor via push notifications using the Telegram application, about the produced trading signal.

Εκτεταμενη Περιληψη

Μιχαήλ Παπατσίμπας, Δ.Μ.Σ. στην Επιστήμη και Μηχανική Δεδομένων, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πολυτεχνική Σχολή, Πανεπιστήμιο Ιωαννίνων, 2022.
Optimized FOREX Trading System Based on Empirical Mode Decomposition and Neural Networks.
Επιβλέπων: Κωνσταντίνος Παρσόπουλος, Καθηγητής

Η αγορά συναλλάγματος (FOREX) είναι η μεγαλύτερη παγκόσμια αγορά. Για τους επενδυτές αποτελεί συνεχή πρόκληση η πρόβλεψη των τιμών ή της τάσης της αγοράς. Στην πραγματικότητα πρόκειται για μια σύνθετη και περίπλοκη αγορά, όπου οι παράγοντες που την επηρεάζουν είναι πολλοί και συχνά απρόβλεπτοι (εκλογές, δημοψηφίσματα, γεωπολιτικά γεγονότα κ.α.). Αυτό μπορεί να επιφέρει απότομες διακύμανσεις και συνεπώς απώλεια ιδιωτικών κεφαλαίων και όχι μόνο. Προκειμένου να μπορέσουν οι επενδυτές να ανταπεξέλθουν στις ξαφνικές αλλαγές, αλλά και να εντοπίσουν πιθανά σημεία εισόδου/εξόδου της αγοράς, αναπτύσσουν σύνθετα υπολογιστικά μοντέλα με αυξημένη ικανότητα πρόβλεψης. Τέτοια τα συστήματα είναι κατεξοχήν υπεύθυνα για τη συμπεριφορά των επενδυτών στην αγορά.

Στην παρούσα μεταπτυχιακή διπλωματική εργασία μελετάται ακριβώς το πρόβλημα πρόβλεψης μελλοντικών τιμών ενος συναλλάγματος. Πιο συγκεκριμένα, προτείνεται ένα αυτοματοποιημένο σύστημα συναλλαγών για την αγορά συναλλάγματος. Για το σύστημα κατασκευάστηκε ένα υβριδικό μοντέλο που ενσωματώνει τρείς διαφορετικές μεθόδους, δηλαδή τη μέθοδο empirical mode decomposition (EMD), τα ανατροφοδοτούμενα νευρωνικά δίκτυα, LSTM (Long-Short Term Memory), καθώς και τον αλγόριθμο εξελικτικής βελτιστοποίησης PSO. Η μέθοδος EMD αποσυνθέτει την αρχική χρονοσειρά, παράγοντας τα intrinsic mode functions (IMF), καθώς και το υπόλοιπο (residual). Έπειτα, για κάθε IMF κατασκεύαζουμε ένα LSTM νευρωνικό δίκτυο και προβλέπουμε την επόμενη τιμή. Η τελική πρόβλεψη προκύπτει από το σταθμισμένο άθροισμα των προβλέψεων των LSTM δικτύων. Στο τελευταίο αυτό

βήμα, για την εύρεση των καταλληλότερων βαρών του σταθμισμένου αθροίσματος, εφαρμόζουμε τον αλγόριθμο βελτιστοποίησης PSO. Στη συνέχεια, λαμβάνοντας υπόψιν την πρόβλεψη του μοντέλου και την στρατηγική μας παράγεται ένα σήμα (BUY, SELL ή HOLD) το οποίο στέλνεται στην πλατφόρμα, όπου εκτελούνται οι ανάλογες οδηγίες. Με την ολοκλήρωση της κίνησης, ο επενδυτής ενημερώνεται για την κίνηση του συστήματος μέσω της εφαρμογής Telegram.

Το προτεινόμενο μοντέλο πρόβλεψης εφαρμόσθηκε σε τρία ζεύγη νομισματικών ισοτιμιών EUR/USD, EUR/CHF και USD/CHF. Εκτενή πειράματα έδειξαν ότι μπορεί να παράγει αποτελέσματα ανώτερης ποιότητας από άλλα μοντέλα πρόβλεψης.

Chapter 1

Introduction

1.1 Aims and objectives

Foreign exchange (FOREX) dates back to ancient times, when traders first began exchanging coins of different countries. However, FOREX is itself the newest of the financial markets. In the last hundred years, the FOREX market has undergone radical transformations. The basic concept behind FOREX market is the trading of currencies, one currency against another. According to the triennial central bank survey [1] conducted by the Bank for International Settlements (BIS) at September 2019, trading in FOREX markets reached $6.6 trillions per day in April 2019, up from $5.1 trillions three years earlier. This renders the currency market the largest financial market in the world. Moreover, the FOREX market consists of multiple international participants, including professionals as well as individuals, who invest and speculate for profit due to its nature of robust liquidity. Picking up patterns in financial time series is difficult but if you are good at it, the reward is huge. In the quest for a trading algorithm, artificial intelligence (AI) methods have been employed to construct systems that perform better or at least equivalently to human traders regarding the timing trade entry and exit opportunities.

Machine learning (ML) is a branch of AI that focuses on the use of data and algorithms in order to imitate human learning positions [2]. Thus, it has become an

important component of the growing field of data science. Based on diverse mathematical methods, ML algorithms use training to make accurate classifications or predictions, thereby discovering knowledge in data mining projects. Neural networks, also known as artificial neural networks (ANN), constitute part of ML artillery, gaining increasing popularity due to deep learning algorithms. A neural network is a model that endeavors to recognize underlying relationships in datasets through a process that mimics the way the human brain operates [3]. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Thus, they can be reasonably considered as the next step in algorithmic trading, as they can directly learn market patterns and behaviors from historical trading data and transform this knowledge into trading decisions.

Algorithmic trading refers to the use of algorithms to make better trade decisions. Usually, human traders build mathematical models that monitor the market in real-time [4]. Such models are able to detect any factors that can possibly force security prices to rise or fall. Unlike human traders, algorithmic trading can simultaneously analyze huge volume of data and make thousands of trading decisions every day. ML methods lie in the basis of such systems, which give human traders an advantage over the market average. In addition, a great advantage of algorithmic trading is that it does not make trading decisions based on emotions, which is a common limitation among human traders whose judgment may be affected by emotions or personal aspirations. However, determining the appropriate structure of a neural network is a difficult task, often resulting in suboptimal solutions.

Another methodology that can prove to be useful is the EMD method, which facilitates the determination of characteristics of complex nonlinear or non-stationary time series, i.e., it can divide the singular values into separated IMFs and determine the general trend of the real time series. This can effectively reduce the unnecessary interactions among singular values and improve the performance when a single kernel function is used in forecasting.

The aim of the present thesis is to introduce a new automated trading system that is able to predict future prices of different securities and trade them on behalf of the trader. The proposed approach combines signal processing, ML methods, and a meta-heuristic optimization algorithm, in order to improve the system's performance. Also, the proposed system handles all trading requests through the MetaTrader platform [5], and notifies the investor for all of its actions via the Telegram application.

1.2 Structure of the thesis

The rest of the thesis is organized as follows: Chapter 2 offers the necessary background information. This includes the FOREX market, EMD algorithm, the LSTM neural network and the PSO algorithm. Chapter 3 presents the proposed approach, while Chapter 4 is devoted to the experimental analysis. Chapter 5 concludes the thesis. Appendix A reports the complete set of experimental results cases as well as the requirements needed for the system.

Chapter 2

Background Information

2.1 Foreign exchange market

The FOREX market is one of the most complex dynamic markets with the characteristics of high volatility, nonlinearity, and irregularity. The market is open 24 hours a day, five days a week. It opens in Australasia, followed by Far East, Middle East, Europe, and finally America. Upon the close of America, Australasia returns to the market and initiates the next 24-hour cycle. The implication of time-zone differences to forecasting is that, under certain circumstances, investors need to consider which data and subsequent time lags to embrace.

One could say that the foreign exchange market has been derived from the dissolve of Bretton Woods System [6] in the early 1970s when President Richard M. Nixon announced that the U.S. would no longer exchange gold for U.S. currency. The Bretton Woods Agreement was negotiated in July 1944 in order to establish a new international monetary system, called the Bretton Woods System. During that

time gold was the basis for the U.S. Dollar; other currencies were pegged to the U.S. Dollar's value. Along with the Bretton Woods Agreement, two important organizations also emerged, namely the International Monetary Fund (IMF) and the World Bank.

The FOREX market is a market where participants can buy, sell, exchange, and speculate on currencies. The market is made up of banks, commercial companies, central banks, investment management firms, hedge funds, and retail FOREX brokers and investors.

Furthermore, some important factors, such as economic growth, trade development, interest rates, and inflation rates have significant impact on the exchange rate fluctuation. These characteristics also make it extremely difficult to predict foreign exchange rates. Therefore, exchange rates forecasting has become a very important and challenging task for both academic and industrial communities.

2.2 Empirical mode decomposition

Empirical Mode Decomposition [7] is an adaptive approach that is used to analyze nonlinear and non-stationary signals. Its novelty lies in the fact that it does not assume a fixed basis for decomposition, like Fourier or Wavelet Transforms. Instead, the resulting IMFs are represented by purely oscillatory functions that can be both frequency and amplitude modulated. The ability to resolve these frequency and amplitude variations in a signal across a small subset of modes is what makes EMD applicable to non-linear data such as those encountered in financial markets. Beside its ability to extract signal features, EMD has also the advantage of locally decomposing the signals, as well as it applicability when the signal varies with time. Figure 2.1 illustrates the flowchart of the EMD algorithm.

2.2.1 Sifting procedure

The sifting procedure is an iterative scheme of removing the dissymmetry between the upper and lower envelopes in order to transform the original signal into an AM signal.

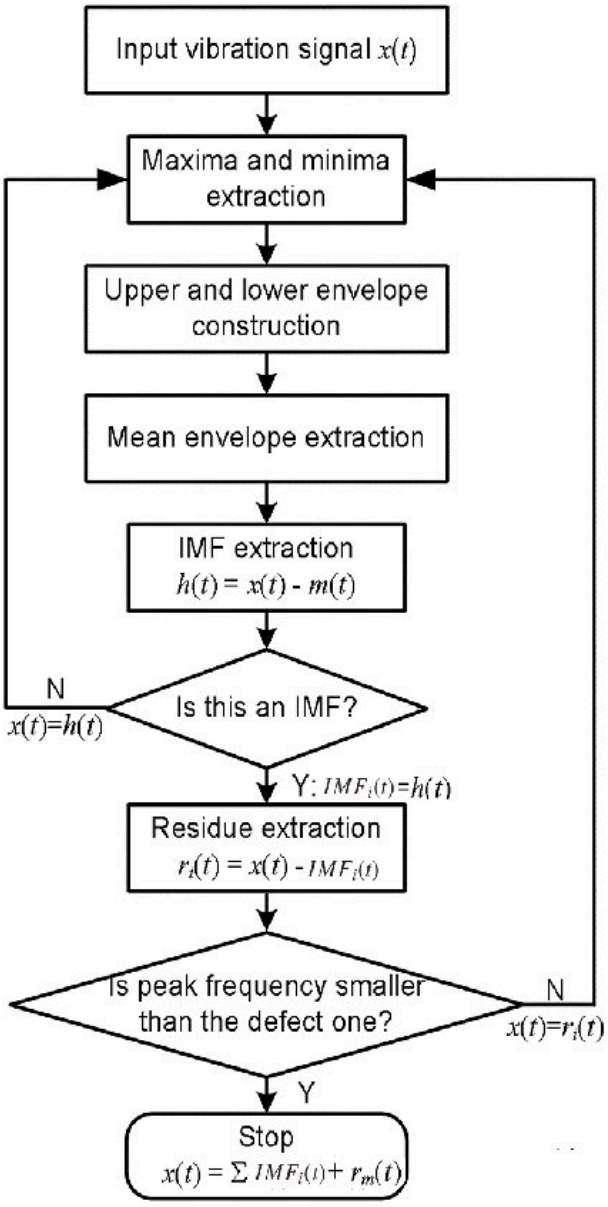


Figure 2.1: Flowchart of the EMD algorithm.

It decomposes a dataset $x(t)$ into IMFs $x_n(t)$ and a residual $r(t)$, such that the signal can be represented as:

$$\hat{x} = \mathcal{T}(t) + \sum_{i=1}^{M} IMF_i(t) \tag{2.1}$$

where, $\hat{x}(t)$ is the reconstructed signal, $\mathcal{T}(t)$ is the trend of $x(t)$ (or residual) and M is the number of sifted IMFs.

The sifting process can be summarized in the following algorithm.

1. Initialize the input $r(t)$ as $x(t)$ (the residue signal).
2. Identify extrema points of $r(t)$: maxima and minima.

3. Interpolate maxima and minima points to form the upper and lower envelopes $e_{max}(t)$ and $e_{min}(t)$ respectively.

4. Evaluate the mean: $m(t) = \frac{e_{min}(t)+e_{max}(t)}{2}$.

5. Extract the detailed signal: $h(t) = r(t) - m(t)$.

6. If $h(t)$ does not satisfy the stopping criteria, then the procedure is repeated and $h(t)$ becomes the input in Step 2.

7. If $h(t)$ satisfies the stopping criteria, then $h(t)$ is the j-th IMF. The residue is $x(t) = r(t) - IMF_{j(t)}$. If the number of zero crossings of the residue is less than two, then break the process and keep the last collected signal as a trend. Otherwise, go back to Step 1, using the residue as the input.

The sifting process serves two main purposes: (a) eliminate riding waves and (b) make the wave profiles more symmetric with respect to zero. Although it is required that the mean value of the upper and lower envelopes is zero when giving definition to IMF component in EMD, in fact, the average of the envelopes of the IMF components separated from the actual signal has no possibility to be zero. Of course, the more times sifting is taken, the closer to zero the average will be. Thus, on the one hand, in order to eliminate the riding waves and enforce a local zero, sifting as many times as possible is needed. On the other hand, too many sifting steps reduce the IMF to be a constant-amplitude frequency-modulated function, which would obliterate the intrinsic amplitude variations and render the results less meaningful physically.

In order to keep the natural amplitude variations of the oscillations, sifting must be limited to as few steps as possible; careful selection is required. Hence, the IMF criterion problem is brought forward: how to estimate in the sifting process whether the decomposed result satisfies the IMF condition? How to determine the sifting times to obtain one IMF component? What exactly the stopping criterion should be is a difficult decision. Other than the stopping criterion, the choice of the appropriate interpolation function is also important.

Stopping criteria is suggested to halt the sifting process at a point that ensures the physical meaning for the extracted IMF. In [8] Huang *et al.* proposed a stopping criteria called S-number. In their approach, S-number is a pre-defined parameter that is used to stop the sifting process whenever the total number of zero-crossings and extrema remains the same or almost differ by one after S-consecutive times. From

their experiments, they concluded that the S-number should be set between 4 and 8. Rilling *et al.* [9] introduced a new criterion based on three thresholds, $(\theta_1, \theta_2, \alpha)$ aiming at guaranteeing globally small fluctuations in the mean, while taking into account locally large excursions. According to their approach, for the $(1-\alpha)$ fraction of the data, sifting will be continued when $\sigma(t) < \theta_1$ while, for the remaining fraction, when $\sigma(t) < \theta_2$, where:

$$\sigma(t) = \left| \frac{m(t)}{a(t)} \right| \tag{2.2}$$

with $a(t) = \frac{e_{max}(t) - e_{min}(t)}{2}$, $m(t) = \frac{e_{max}(t) + e_{min}(t)}{2}$ and e being an envelope. Rato *et al.* [10] defined a resolution factor by the ratio between the energy of the signal at the beginning of the sifting, $x(t)$, and the energy of average of the envelopes, $e(t)$. If this ratio grows above the allowed resolution, then the IMF computation must stop. This criterion gives a scale independent stopping way, as opposed to criteria based on iteration count. A useful property of the RF is that it enables the researcher to set the number of IMFs. Reducing the resolution factor reduces the number of obtained IMFs.

Interpolation is used in EMD as the process of connecting the identified extrema (maxima/minima) to form the upper and lower envelopes, respectively. In [7], Huang *et al.* used cubic spline interpolation to fit all maxima (minima) data points. Using parabolic interpolation, each extremum is estimated from only the above defined three samples. A new interpolated extremum sample is obtained, usually without an integer abscissa, and this new sample is fed into the EMD algorithm, replacing the integer abscissa extremum sample as an envelope defining point.

2.2.2 Intrinsic mode functions

By the nature of the decomposition procedure, the data is decomposed into fundamental components each with distinct time scale. More specifically, the first component corresponds to the smallest time scale, which stands for the fastest time variation of the data. As the decomposition process proceeds, the time scale, increases and, hence, the mean frequency of the mode decreases. In addition, all IMFs must meet the following conditions:

- For a set of data sequences, the number of extremal points must be equal to the number of zero crossings or, at most, differ by one.

- For any point, the mean value of the envelope of the local maxima and local minima must be zero.

Therefore, the EMD essentially is a decomposition of the original signal into a set of AM/FM modulated signals. It can be easily perceived that the envelopes cannot vary as fast as the signal $x(t)$. In spectral terms, we can say that the bandwidth of the envelopes must be a fraction of the central frequency (normally called carrier). This means that sifting eliminates the low frequency components, leaving a high frequency signal. This explains why the IMFs appear in a high-to-low frequency order, as well as why the EMD is essentially a time-frequency decomposition.

2.3 Machine learning

Over the past two decades ML has become one of the mainstays of information technology and with that, a rather central, albeit usually hidden, part of our life. We can define ML as the field of computer science that aims, as its name implies, to create intelligent machines that automatically improve with experience gained through data.

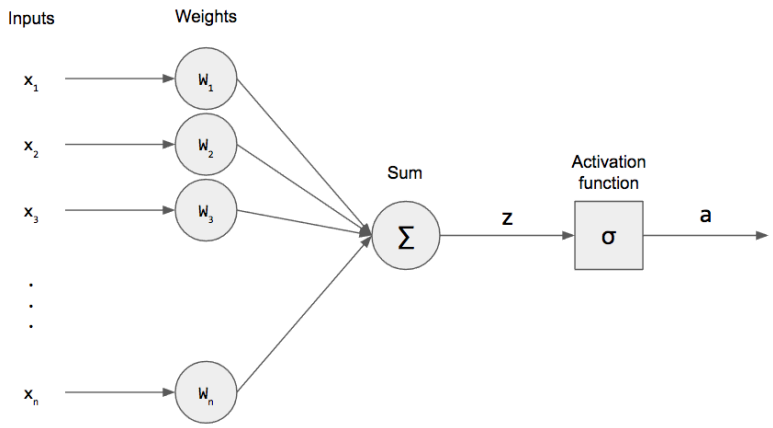


Figure 2.2: Rosenblatt's Perceptron

The work of McCulloch and Pitts [11] introduced one of the first biologically inspired cognitive models. This approach was novel and powerful as it was able to emulate a wide variety of boolean functions by combining simple binary computational units. However, its architecture lacked several characteristics of biological networks such as complex connectivity patterns, processing of continuous values (rather than just binary), and learning procedures. Furthermore, the fact that brain activity

tends to be noisy and seemingly stochastic did not fit well with the consistency and predictability required by the McCulloch and Pitts model.

Taking into consideration the aforementioned limitations of early neural network models, Frank Rosenblatt introduced the so-called "Perceptron" in 1958 [12]. This machine used both analog and discrete signals, including also a threshold element that converted analog signals into discrete ones. A perceptron can also be called a single-layer neural network as it is the part of the only layer in the neural network where computation occurs. The computation takes place on the summation of input data that is fed into it. In this model, we have n inputs (usually given as a vector) and exactly the same number of weights $W_1, ..., W_n$. We multiply these together and sum them up. The output is denoted as z, and called the pre-activation:

$$z = \sum_{i=1}^{n} W_i x_i = W^T x \tag{2.3}$$

There is another term, called the bias, that is just a constant factor added to z:

$$z = \sum_{i=1}^{n} W_i x_i = W^T x + b_0 \tag{2.4}$$

After calculating the weighted sum, we apply an activation function, σ, and produce an activation α.

$$\sigma(q) = \begin{cases} 1 & q \geq 0 \\ 0 & q < 0 \end{cases} \tag{2.5}$$

$$\alpha = \sigma(W^T x) \tag{2.6}$$

The activation function for perceptrons is frequently called a step function because, if we were to plot it, it would look like stairs. In other words, if the input is greater than or equal to 0, then we produce an output of 1. Otherwise, we produce an output of 0. This is the mathematical model for a single neuron, the most fundamental unit for a neural networks. In Fig. 2.2 we show the architecture of Perceptron, and Alg. 2.1 shows the corresponding training algorithm.

2.4 Deep learning

We are living in the era of big data where all scientific and industrial applications generate massive amounts of data. This confronts us with unprecedented challenges

Algorithm 2.1 Perceptron Algorithm.

Require: $P \leftarrow$ inputs of label 1

Require: $N \leftarrow$ inputs of label 0

Require: $\eta > 0$ learning rate

Ensure: w initialized randomly

```
while not converge do
    x ← random(), x ∈ P ∪ N
    if x ∈ P and w^T x < 0 then
        w ← w + ηx
    end if
    if x ∈ N and w^T x ≥ 0 then
        w ← w − ηx
    end if
end while
return w
```

regarding their analysis and interpretation. For this reason, there is an urgent need for novel ML and AI methods that can help in utilizing these data. Deep learning (DL) is such a novel methodology currently receiving much attention [13]. DL describes a family of learning algorithms rather than a single method that can be used to learn complex prediction models, e.g., multi-layer neural networks with many hidden units [14]. A common characteristic of the many variations of supervised and unsupervised deep learning models is that these models have many layers of hidden neurons. In order to build neural networks (NNs), the neurons need to be connected with each other. The simplest architecture of an NN is a feedforward structure. In Fig. 2.3, we show an example for a deep architecture.

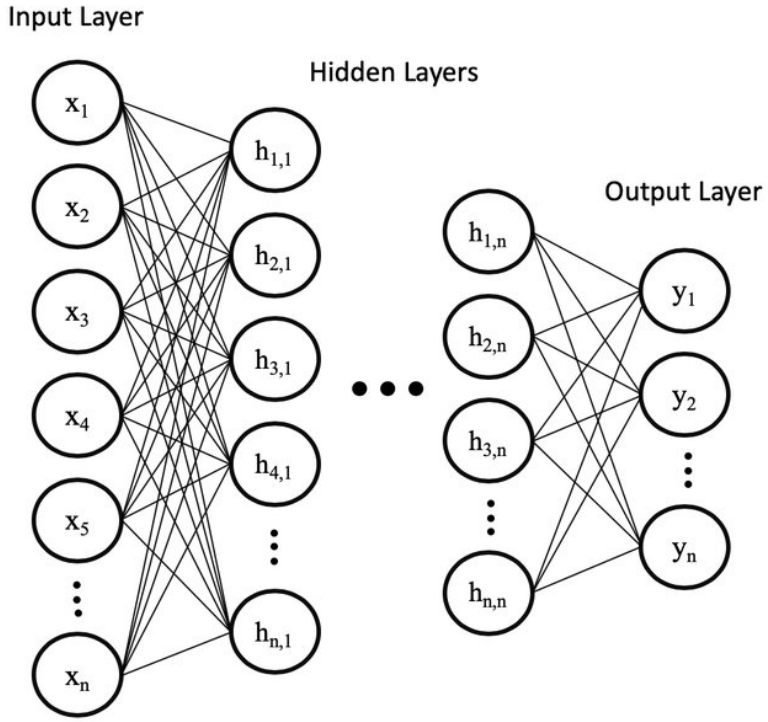


Figure 2.3: A Deep FeedForward Neural Network

ML and DL models are capable of different types of learning as well, which are usually categorized as supervised learning, unsupervised learning, and reinforcement learning. Supervised learning utilizes labeled datasets to categorize or make predictions; this requires some kind of human intervention to label input data correctly. In contrast, unsupervised learning does not require labeled datasets. Instead, it detects patterns in the data, clustering them according to various distinguishing characteristics. Reinforcement learning is a process where a model learns to become more accurate for performing an action in an environment based on feedback in order to maximize its reward.

2.5 Recurrent neural networks

The recurrent NN (RNN) was first developed in the 1980s [15]. Its structure consists of an input layer, one or more hidden layers, and an output layer. RNNs have chain-like structures of repeating modules, which are used as a memory to store important information from previous processing steps. Unlike feedforward neural networks, RNNs include a feedback loop that allows the neural network to accept a sequence

of inputs. This means that the output from step $t-1$ is fed back into the network to influence the outcome of step t, for each subsequent step. Therefore, RNNs have been successfully applied in learning sequences. Figure 2.4 illustrates the sequential processing in RNNs.

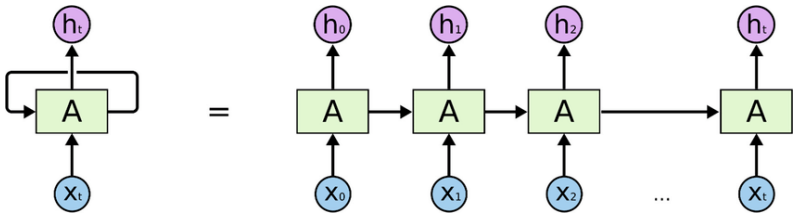


Figure 2.4: Sequential processing in a Recurrent Neural Network (RNN).

Another distinguishing characteristic of RNNs is the parameters sharing across each layer of the network. While feedforward networks have different weights across each node, RNNs share the same weight parameter within each layer of the network.

RNNs leverage the backpropagation through time (BPTT) algorithm to determine the gradients, which is slightly different from traditional backpropagation as it is specific to sequence data. The principles of BPTT are the same as traditional backpropagation, where the model trains itself by calculating errors from its output layer to its input layer. These calculations allow us to appropriately adjust and fit the parameters of the model. BPTT differs from the traditional approach as it sums errors at each time step, while feedforward networks do not need to sum errors as they do not share parameters across each layer.

Following the training procedure above, RNNs tend to run into two problems known as *exploding gradients* and *vanishing gradients*. These issues are related to the size of the gradient, which is the slope of the loss function along the error curve. On the one hand, when the gradient is too small, it recursively continues to become smaller, updating the weight parameters until they become near 0. When that occurs, the algorithm can no longer learn. On the other hand, exploding gradients occur when the gradient is too large, thereby creating an unstable model. In this case, the weights will grow too large, becoming computationally intractable (represented as NaN). A possible solution to those issues is to reduce the number of hidden layers within the neural network, hence reducing the complexity in the RNN model.

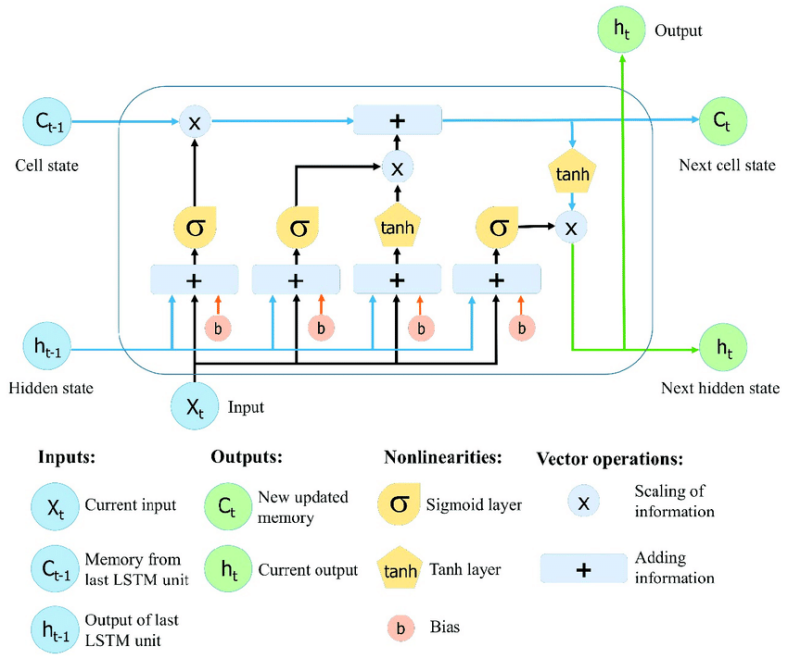


Figure 2.5: Long-short term memory unit.

2.5.1 Long short term memory neural networks

The LSTM model [16] is a powerful recurrent neural system specially designed to overcome the exploding/vanishing gradient problems that typically arise when learning long-term dependencies, even when the minimal time lags are very long [17]. Overall, this problem can be prevented by using a constant error carousel (CEC), which maintains the error signal within each unit's cell. In practice, such cells are also recurrent networks with an interesting architecture. More specifically, in the way that the CEC is extended with additional features, namely the input gate and output gate, forming the memory cell. The self-recurrent connections indicate feedback with a lag of one time step. In Fig. 2.5, the basic structure of an LSTM unit is illustrated.

2.6 Particle swarm optimization

PSO is a population-based search algorithm inspired by the swarming behavior of particles, which is also met in different hierarchically organized populations. It was initially introduced in the pioneering works of Eberhart and Kennedy in 1995 [18]. In PSO, individuals referred to as *particles* are "flown" through the search space. Each

particle determines its move by stochasticaly combining some aspect of the history of its own current and best locations, with those of a group of other members of the swarm. The next iteration takes place after all particles have moved to a new position. The search behavior of a particle is hence affected by that of other particles within the swarm.

Let x_i denote the i-th particle, $S = (x_1, \ldots, x_n)$ be the swarm consisting of n particles, and $\mathcal{X} \in \mathbb{R}^n$ be the search space in which the particles' position change. Also, let the index set $I = \{1, 2, \ldots, n\}$, and d be the optimization problem's dimension. The position of each particle is an d-dimensional vector:

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{id})^\top \in \mathcal{X}, \forall i \in I \tag{2.7}$$

The velocity of the particle is also a d-dimensional vector:

$$v_i = (v_{i1}, v_{i2}, \ldots, v_{id})^\top, \forall i \in I \tag{2.8}$$

The best previous position encountered by the i-th particle is denoted as:

$$p_i = (p_{i1}, p_{i2}, \ldots, p_{id})^\top \in \mathcal{X}, \forall i \in I \tag{2.9}$$

Then, the swarm is manipulated as follows:

$$v_{ij}(t+1) = \chi[v_{ij}(t) + c_1 r_1(p_{ij}(t) - x_{ij}(t)) + c_2 r_2(p_{g_i j}(t) - x_{ij}(t))] \tag{2.10}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \tag{2.11}$$

where $i \in I, j = 1, 2, \ldots, d$, g_i is the index of the particle that attained the best previous position among all the particles in the (local or global) neighborhood of x_i, χ is a parameter called *constriction factor*, c_1 and c_2 are positive acceleration constants used to scale the contribution of the cognitive and social components, respectively, and $r_1, r_2 \sim$ U(0,1) are uniform random variables that introduce stochasticity to the algorithm.

The constriction factor is a mechanism for controlling the magnitude of the velocities. The stability analysis of Clerk and Kennedy [19] suggests that:

$$\chi = \frac{2k}{\left|2 - \phi - \sqrt{\phi^2 - 4\phi}\right|} \tag{2.12}$$

where $\phi = c_1 + c_2$. The values received for $\phi > 4$ and $k = 1$ are considered the default setting due to their good average performance.

For minimization problems, the best positions $p_i(t)$ is updated as :

$$p_i(t+1) = \begin{cases} x_i(t+1), & \text{if } f(x_i(t+1)) \leq f(p_i(t)) \\ p_i(t), & \text{otherwise.} \end{cases} \tag{2.13}$$

Chapter 3

Proposed Approach

3.1 Proposed forecasting model

Forecasting financial data (e.g., exchange rates, stocks) is a challenging task that requires advanced models in order to attain high accuracy. In the present thesis, a new forecasting model was developed and applied to the FOREX market, although its applicability can be extended to any other financial instrument (such as stocks, cryptocurrencies, etc).

The proposed forecasting model comprises a number of discrete steps. Firstly, the closing prices of a currency are received from the MetaTrader platform. Secondly, the EMD decomposition is applied on the data and the IMF components are produced. As mentioned before, the use of the EMD algorithm aims at receiving smoothed components, which contain only the important information of the signal and remove much of the noise. Thus, the produced components are easier to predict and render the trading outcome more robust to noise. Next, each IMF is preprocessed to scale and split the data into train, validation, and test sets and, then, it is ''fed" into an LSTM model. Then, we start the LSTM training phase and, after its completion, each LSTM predicts the future value of the corresponding IMF. In the next step, the applied

transform of each IMF is inverted. Finally, the forecasted value is given as follows:

$$\hat{y}_{N+1} = \sum_{j=1}^{k} w_j \hat{y}_{N+1}^{(j)} \tag{3.1}$$

where w_j is the weight of the j-th IMF, and $\hat{y}_{N+1}^{(j)}$ is the corresponding unscaled prediction at time $N+1$. Moreover, in order to increase accuracy, the PSO algorithm is used, in order to detect a (sub-) optimal w that minimizes the prediction error. In Fig. 3.1 a visual representation of all the above steps is presented.

Putting it formally, let m be the number of past observations that will be used to predict y_{N+1}. Let $A = \{y_{N-m+1}, y_{N-m+2}, \ldots, y_N\}$ be the set of these observations, i.e., the time-window under consideration. Then, for each $y_i \in A$ the forecasted values $\hat{y}_i^{(j)}, j = 1, 2, \ldots, k$, are calculated by aggregating the forecasts of each LSTM. These values are combined as in Eq. (3.1) using an arbitrary initial weight vector, thereby producing an aggregate forecast $\hat{y}_i$ of y_i. Thus, for the whole set A, a set of aggregate forecasts $F = \{\hat{y}_{N-m+1}, \ldots, \hat{y}_N\}$ is received. These values correspond to the specific weight vector W that was used in Eq. (3.1). Assuming that the selected optimization criterion is the minimization of the mean absolute error between the forecasted values and the real ones, an optimization algorithm 3.1 is employed to minimize the objective function:

$$E(W) = \frac{\sum_{l=N-m+1}^{N} \{e_l\}}{m}, \tag{3.2}$$

where:

$$e_l = |\hat{y}_l - y_l|, \qquad l = N - m + 1, \ldots, N \tag{3.3}$$

are the absolute errors. In other words, the optimization algorithm tries to find the specific weight vector that minimizes the selected criterion (i.e., the mean absolute error in our example) for the whole window of past observations.

Let $W^* = (w_1^*, \ldots, w_k^*)$ be the best weight vector detected by the optimization algorithm. Then, the aggregate forecast for the y_{N+1} observation is eventually computed as:

$$\hat{y}_{N+1} = \sum_{j=1}^{k} w_j^* \hat{y}_{N+1}^{(j)} \tag{3.4}$$

where $\hat{y}_{N+1}^{(j)}$ is the forecasted value of y_{N+1} by the j-th LSTM. Thus, the proposed model aggregates the k LSTMs using weights that provide the best possible aggregate predictions in the past m moves defined by the sliding window. When the actual value y_{N+1} is available, the whole process is repeated anew, because a new data value

modifies the corresponding IMFs. Moreover, the sliding window may have dynamic size. In the most trivial case, the user may use all past observations as the employed window. In the conducted experiments the PSO algorithm was the selected optimizer, because it is a simple and efficient algorithm requiring only minor implementation effort. Also, it can be applied on both differentiable and non-differentiable error functions.

Algorithm 3.1 Particle Optimization Algorithm.

Require: $Max\_It \leftarrow$ Maximum number of iterations

Require: $N \leftarrow$ Swarm size

$\tau \leftarrow 0$

for $i = 1$ **to** N **do**

  $W_i^{(\tau)} \leftarrow \mathcal{U}\left([1, 1.5]^k\right)$

  $f_i^{(\tau)} \leftarrow E\left(W_i^{(\tau)}\right)$ `[ use Algorithm 3.2]`

  return w

end for

$S^{(\tau)} \leftarrow \left\{W_1^{(\tau)}, \ldots, W_N^{(\tau)}\right\}$ `// swarm`

$P^{(\tau)} \leftarrow S^{(\tau)}$ `// best positions`

$g^* \leftarrow \arg\min_{i=1\ldots N} \left\{f\left(P_i^{(\tau)}\right)\right\}$

`/* evolve swarm */`

while $\tau < \tau_{\max}$ **do**

  $\tau \leftarrow \tau + 1$

  $S^{(\tau)} \leftarrow$ **update-swarm** $\left(S^{(\tau-1)}\right)$

  for $i = 1$ **to** N **do**

    $f_i^{(\tau)} \leftarrow E\left(W_i^{(\tau)}\right)$ `[ use Algorithm 3.2]`

  end for

  $P^{(\tau)} \leftarrow$ **update-best-positions** $\left(S^{(\tau)}, P^{(\tau-1)}\right)$

  $g^* \leftarrow \arg\min_{i=1\ldots N} \left\{f\left(P_i^{(\tau)}\right)\right\}$

end while

$W^* \leftarrow P_{g^*}^{(\tau)}$

return W^* `// Optimal weight vector`

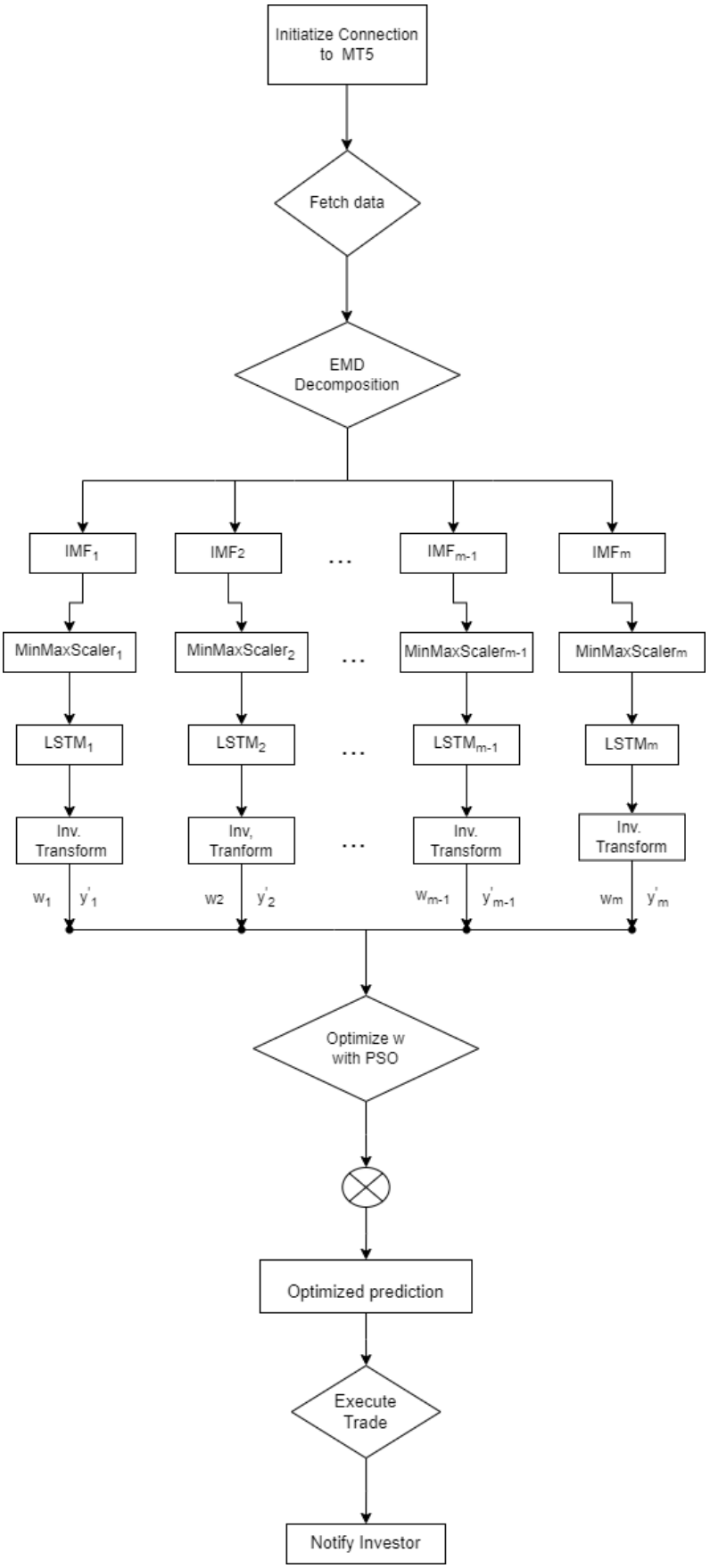


Figure 3.1: Flowchart of the proposed model

Algorithm 3.2 Pseusocode for the evaluation of $E(W)$ for a given weight vector W.

Require: $W = (w_1, \ldots, w_k)$, // `weight vector`

Require: $Y = \{y_1, \ldots, y_t\}$ // `time series`

Require: $I = \{I_1, \ldots, I_k\}$ // `set of IMFs`

Require: m // *window size*

for $i = t - m + 1$ **to** t **do**
  for $j = 1$ **to** k **do**
    $\hat{y}_i^{(j)} \leftarrow I_j\left(y_1, y_2, \ldots, y_{i-1}\right)$
  end for
end for
for $i = t - m + 1$ **to** t **do**
  $\hat{y}_i \leftarrow 0$
  for $j = 1$ **to** k **do**
    $\hat{y}_i \leftarrow \hat{y}_i + w_j\, \hat{y}_i^{(j)}$
  end for
  $\varepsilon_i \leftarrow |\hat{y}_i - y_i|$
end for
`/* objective value */`
$E(W) \leftarrow \frac{\varepsilon_{t-m+1} + \varepsilon_{t-m+2} + \ldots + \varepsilon_t}{t}$
return $E(W)$

3.2 Trading strategy

The primary contribution of the proposed model is the sophisticated aggregation forecasting scheme presented in the previous section. Nevertheless, such a scheme is only a part of a complete trading system. Technical analysis always offers critical information that can be used to the benefit of the trader along with the accurately forecasted values. For this reason, a strategy for the proposed FOREX trading model was developed based on indices used in technical analysis.

In a currency pair such as EUR/USD, the enumerator (EUR) and the denominator (USD) are called the base and the quote currency, respectively. Also, the exchange rate represents how much of the quote currency is needed in order to get one unit of the base currency.

At each step, the proposed model produces one of the following three trading signals:

1. *Buy signal*

   The "Buy signal" occurs when the model forecasts that, in the next step, the base currency will increase in value against the quote currency. Thus, the system shall open a "Buy position" by buying and selling the base and quote currency at time t, respectively. For instance, suppose that the exchange rate of EUR/USD is equal to 1.5, which means that traders need \$1.5 US to buy 1 Euro. If the exchange rate increases from 1.5 to 1.7, and the traders have bought Euro at the price of \$1.5, they can Sell them back at the price of \$1.7. As a result, they won that trade and made a profit.

2. *Sell signal*

   The "Sell signal" is the exact opposite of a Buy signal. In this case, the model predicts that, in the next step, the base currency will decrease in value against the quote currency. Therefore, a Sell position is opened by selling and buying the base and quote currency at time t, respectively.

A trading signal indicates the position a trader should open. There are three position types:

1. *Buy position*

   A "Buy position" is opened when the model produces a Buy signal.

2. *Sell position*

   A "Sell position" is opened when the model produces a Sell signal.

3. *Hold position*

   A "Hold position" is opened when the same signal as the one in the previous step appears. For instance, if the signal at time $t-1$ was "Buy", and the same signal is received at time t, then the already opened Buy position is retained without opening a second one.

Based on the different trading signals, two trading rules are created. On the one hand, if the actual value of the exchange rate at time t is lower than the forecasted value at

$t+1$, a "Buy" signal is created. On the other hand, if the actual value of the exchange rate at time t is greater than the forecasted value at $t+1$, we have a "Sell" signal is created. The trading rules can be summarized as follows:

1. **If** $Actual_t < Forecast_{t+1}$ **then** $Signal_t \leftarrow$ *Buy*
2. **If** $Actual_t > Forecast_{t+1}$ **then** $Signal_t \leftarrow$ *Sell*

Also, before opening a new position, it shall be checked if it is a "Hold" position:

$$\textbf{If } Signal_{t-1} == Signal_t \textbf{ then } Position_t \leftarrow \textit{Hold}$$

Finally, our model selects one of the above rules and, if there is no "Hold" position, it opens the corresponding position.

3.3 Profit calculation

A position opens when a "Buy" ("Sell") signal is encountered, and closes when a "Sell" ("Buy") signal occurs. When we have a "Hold" position the profit remains constant. The total profit of the model is calculated as follows:

1. Position was opened as "Buy"at time t_{open} but now the new signal is "Sell":

$$\texttt{profit}_{t_{now}} = 10000(Actual_{t_{now}} - Actual_{t_{open}}) \tag{3.5}$$

2. Position was opened as "Buy" at time t_{open} but now the new signal is "Buy":

$$\texttt{profit}_{now} = profit_{t_{open}} \tag{3.6}$$

3. Position was opened as "Sell" at time t_{open} but now the new signal is "Buy":

$$\texttt{profit}_{now} = 10000(Actual_{t_{open}} - Actual_{t_{now}}) \tag{3.7}$$

4. Position was opened as "Sell" at time t_{open} but now the new signal is "Sell":

$$\texttt{profit}_{now} = profit_{t_{open}} \tag{3.8}$$

In the relations above, the profit calculation requires multiplication with 10000. The reason is that the profit is calculated in pips. A pip expresses the smallest change in value between two currencies and is defined as the fourth decimal point in most currencies (1 pip = 0.0001 of a cent), thus the result shall be multiplied by 10000.

Chapter 4

Experimental Analysis

4.1 MetaTrader platform

MT5 is a free application for traders, allowing them to perform technical analysis and trading operations in FOREX and exchange markets. All of the trades in our experiments are executed through the MT5 platform. Figure 4.1 shows a glimpse of the platform.

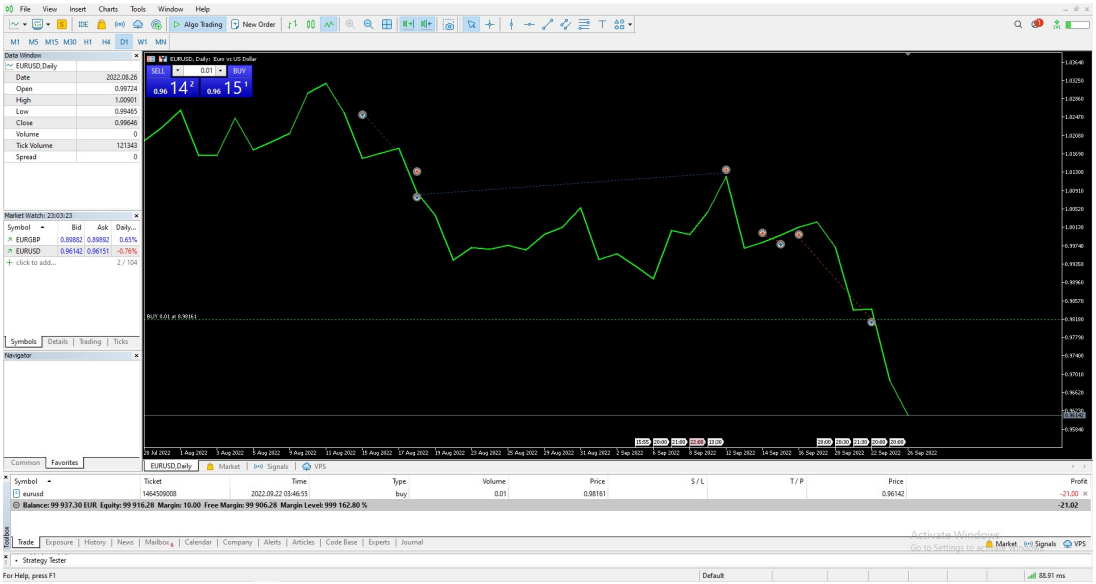


Figure 4.1: MetaTrader platform

4.2 Data preparation

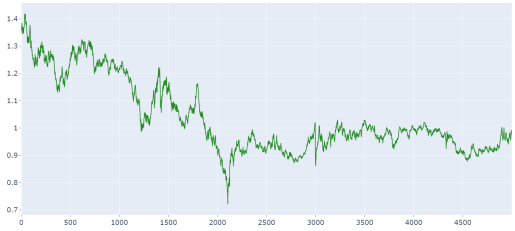
For the needs of the proposed trading system datasets consisting of 5000 daily past observations are considered. For the EUR/USD pair the dates range from 2003/07/07 to 2022/10/03, for the USD/CHF pair, the dates range from 2003/07/04 to 2022/10/03, and for the EUR/CHF pair the dates range from 2003/06/20 to 2022/10/03. In Figs. 4.2a, 4.2b, 4.2c the investigated time series are shown.

Next, for each dataset the EMD method is applied and the IMF components are being displayed in Figs. 4.3, 4.4 and 4.5 are generated. Secondly, each IMF component is normalized as follows:

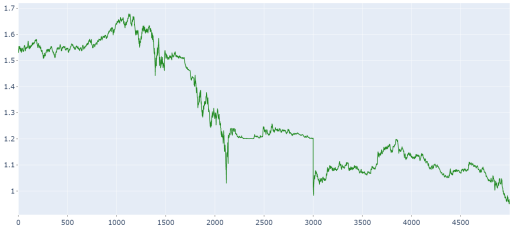
$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}, \tag{4.1}$$

where x is the corresponding IMF component. The main reason for normalization/standardization is that variables measured at different scales do not contribute equally to the model fitting and model learned function. Thus, they might end up creating a bias. This potential problem can be addressed through feature-wise normalization, such as MinMax Scaling, prior to the model fitting procedure.

(a) The EUR/USD currency pair.

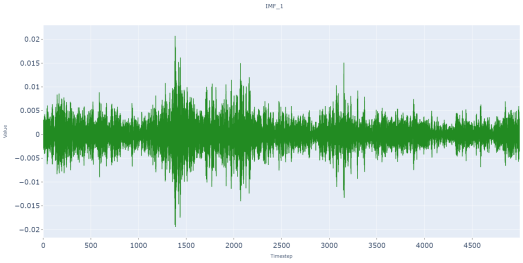


(b) The USD/CHF currency pair.

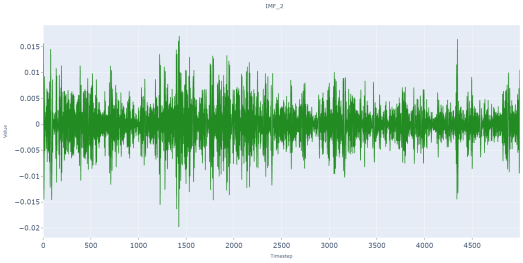


(c) The EUR/CHF currency pair.

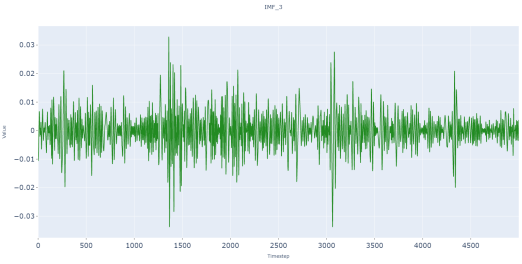
Figure 4.2: The investigated currency pairs.



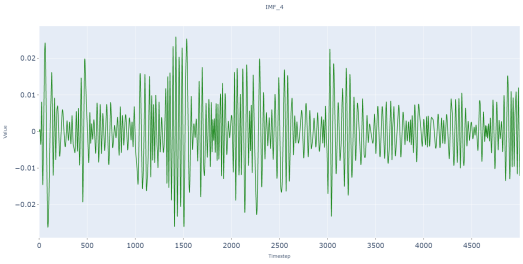
(a) IMF_1



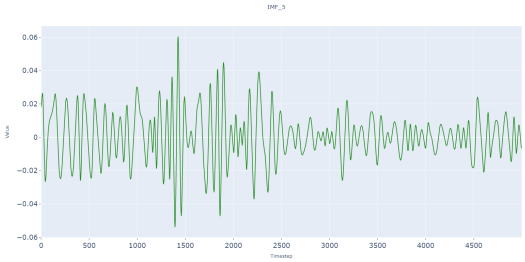
(b) IMF_2



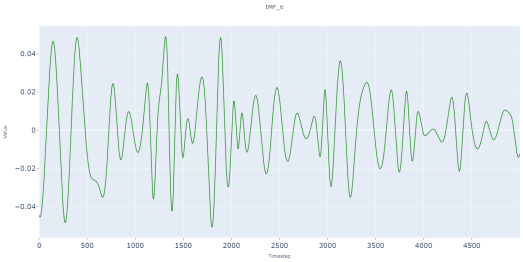
(c) IMF_3



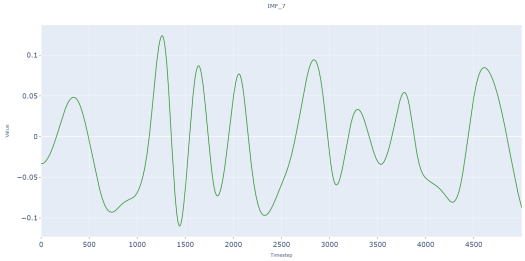
(d) IMF_4



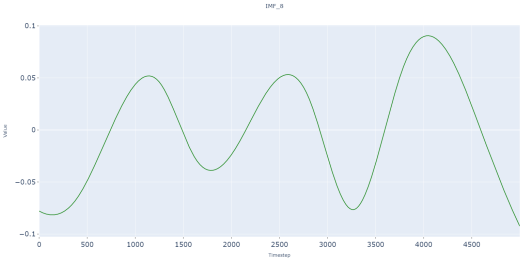
(e) IMF_5



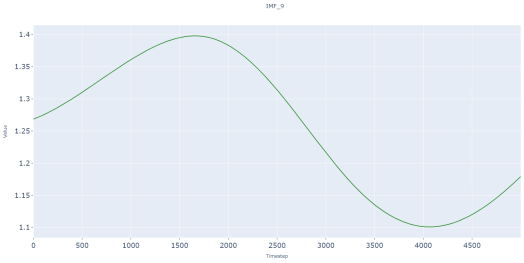
(f) IMF_6



(g) IMF_7

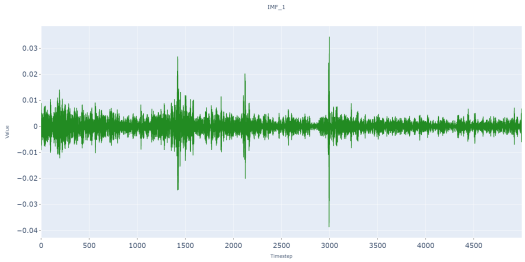


(h) IMF_8

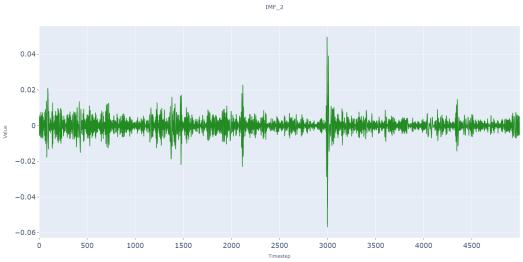


(i) IMF_9

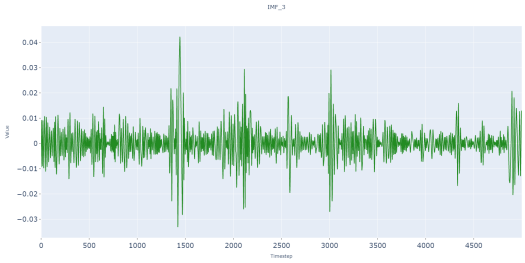
Figure 4.3: EUR/USD IMF Components.



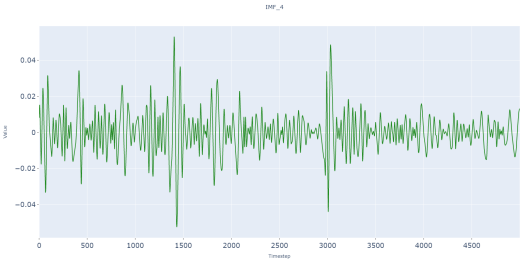
(a) IMF_1



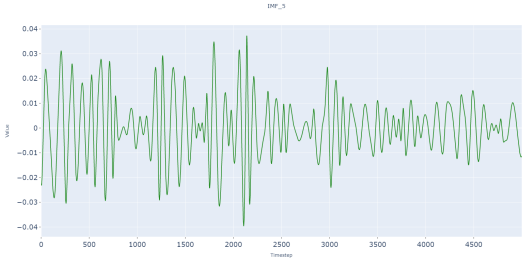
(b) IMF_2



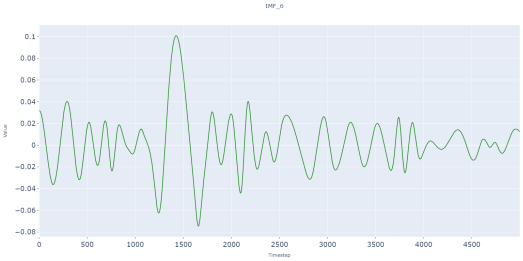
(c) IMF_3



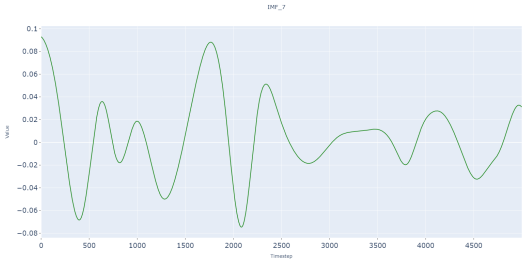
(d) IMF_4



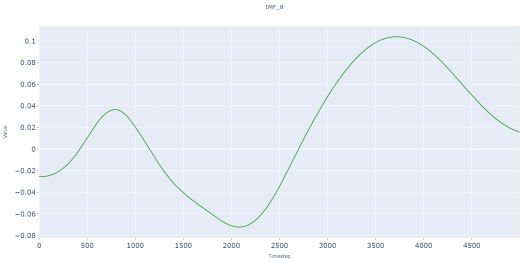
(e) IMF_5



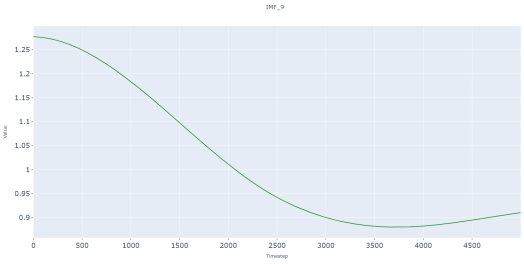
(f) IMF_6



(g) IMF_7

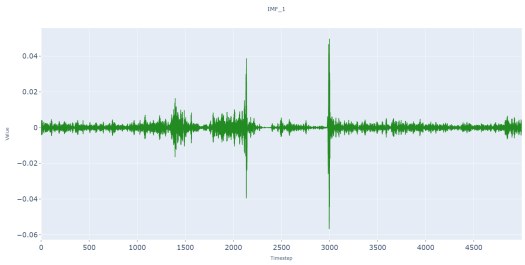


(h) IMF_8

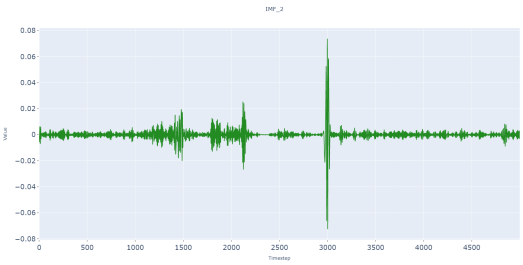


(i) IMF_9

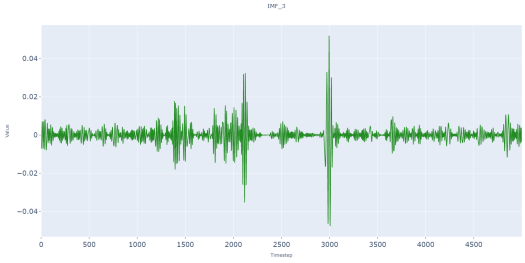
Figure 4.4: USD/CHF IMF Components.



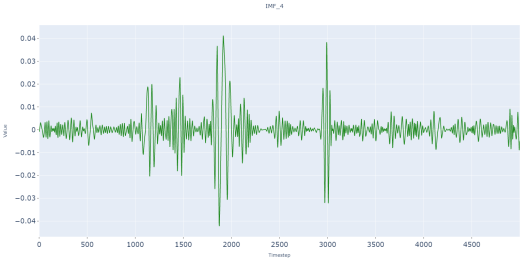
(a) IMF_1



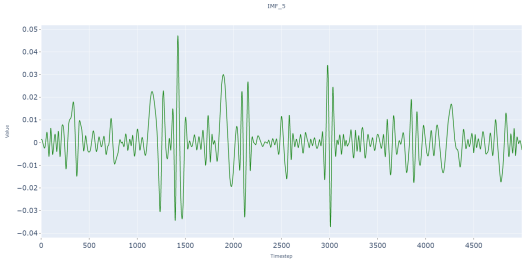
(b) IMF_2



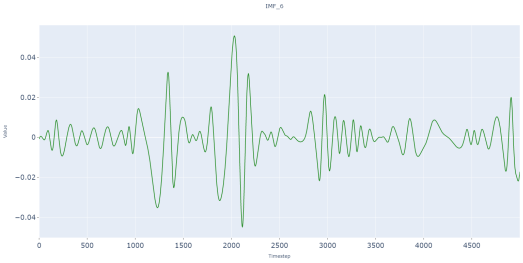
(c) IMF_3



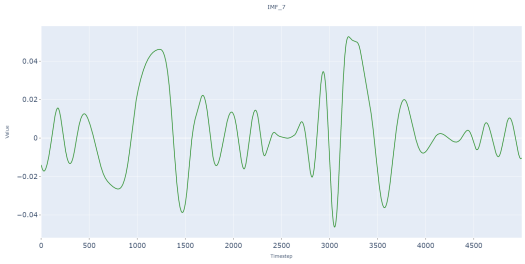
(d) IMF_4



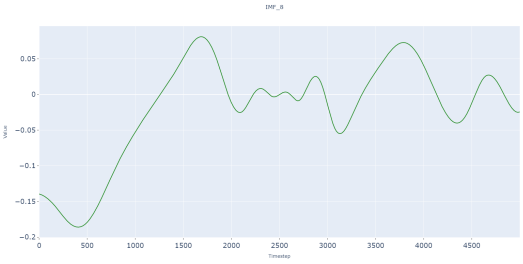
(e) IMF_5



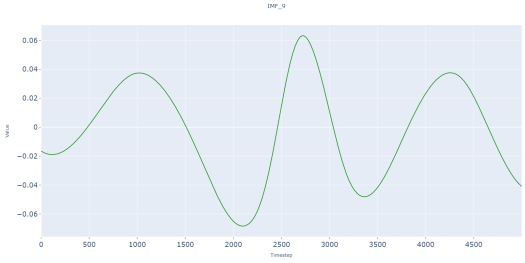
(f) IMF_6



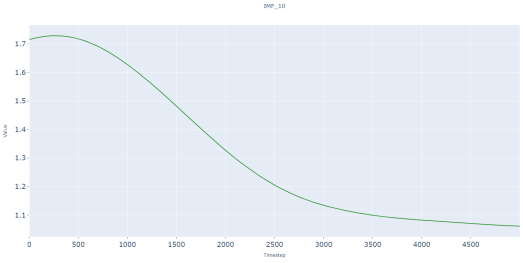
(g) IMF_7



(h) IMF_8



(i) IMF_9



(j) IMF_{10}

Figure 4.5: EUR/CHF IMF Components.

Table 4.1: Statistical analysis.

(a) EUR/USD IMF statistical analysis.

IMF	min	max	mean	median	std
IMF_1	-0.01945	0.02073	-2e-05	-6e-05	0.00363
IMF_2	-0.01981	0.01709	-2e-05	-5e-05	0.004
IMF_3	-0.03382	0.03292	-3e-05	-5e-05	0.00638
IMF_4	-0.02612	0.02591	-0.00016	-0.00014	0.00776
IMF_5	-0.05392	0.06034	0.00054	0.00057	0.01457
IMF_6	-0.05216	0.0494	0.00014	-0.00016	0.0194
IMF_7	-0.10966	0.12397	-0.00979	-0.01639	0.05699
IMF_8	-0.08729	0.08793	-0.00051	0.00041	0.04992
IMF_9	1.10191	1.398	1.25612	1.28024	0.10682

(b) EUR/CHF IMF statistical analysis.

IMF	min	max	mean	median	std
IMF_1	-0.05683	0.04977	-0.0	0.0	0.00335
IMF_2	-0.07253	0.07378	3e-05	-1e-05	0.0049
IMF_3	-0.04769	0.05204	-3e-05	-1e-05	0.00521
IMF_4	-0.04221	0.0412	0.00024	2e-05	0.00666
IMF_5	-0.03722	0.04707	0.00029	-4e-05	0.00915
IMF_6	-0.04488	0.05088	-0.00059	-0.0002	0.01151
IMF_7	-0.04633	0.05275	0.00258	0.00085	0.01992
IMF_8	-0.18591	0.08098	-0.02033	-0.00332	0.07062
IMF_9	-0.06837	0.06322	-0.00243	-0.00156	0.03408
IMF_{10}	1.06115	1.72894	1.31394	1.20539	0.24687

(c) USD/CHF IMF statistical analysis.

IMF	min	max	mean	median	std
IMF_1	-0.03875	0.03447	1e-05	0.0	0.00356
IMF_2	-0.05685	0.04981	-1e-05	-2e-05	0.00451
IMF_3	-0.03312	0.04227	8e-05	-0.0	0.00631
IMF_4	-0.05253	0.05314	-6e-05	-0.00019	0.01091
IMF_5	-0.03963	0.03729	0.0002	0.00013	0.01209
IMF_6	-0.07453	0.1008	0.0016	0.00083	0.02517
IMF_7	-0.07461	0.09302	0.00368	0.00391	0.03429
IMF_8	-0.07211	0.10405	0.01752	0.02023	0.0545
IMF_9	0.88022	1.27725	1.01366	0.94226	0.14087

Table 4.2: Augmented Dickey-Fuller test

(a) EUR/USD IMFs Augmented Dickey-Fuller test.

IMF	ADF Statistic	p-value	1%	5%	10%
IMF_1	-23.0731	0.0	-3.4317	-2.8621	-2.5671
IMF_2	-22.6184	0.0	-3.4317	-2.8621	-2.5671
IMF_3	-15.6429	0.0	-3.4317	-2.8621	-2.5671
IMF_4	-16.7469	0.0	-3.4317	-2.8621	-2.5671
IMF_5	-12.2946	0.0	-3.4317	-2.8621	-2.5671
IMF_6	-10.3696	0.0	-3.4317	-2.8621	-2.5671
IMF_7	-4.387	0.0003	-3.4317	-2.8621	-2.5671
IMF_8	-0.3379	0.92	-3.4317	-2.8621	-2.5671
IMF_9	-6.0722	0.0	-3.4317	-2.8621	-2.5671

(b) EUR/CHF IMFs Augmented Dickey-Fuller test.

IMF	ADF Statistic	p-value	1%	5%	10%
IMF_1	-16.6737	0.0	-3.4317	-2.8621	-2.5671
IMF_2	-12.4855	0.0	-3.4317	-2.8621	-2.5671
IMF_3	-24.8993	0.0	-3.4317	-2.8621	-2.5671
IMF_4	-14.5824	0.0	-3.4317	-2.8621	-2.5671
IMF_5	-10.0124	0.0	-3.4317	-2.8621	-2.5671
IMF_6	-8.811	0.0	-3.4317	-2.8621	-2.5671
IMF_7	-4.3768	0.0003	-3.4317	-2.8621	-2.5671
IMF_8	-2.3904	0.1445	-3.4317	-2.8621	-2.5671
IMF_9	-1.0023	0.7524	-3.4317	-2.8621	-2.5671
IMF_{10}	-5.1755	0.0	-3.4317	-2.8621	-2.5671

(c) USD/CHF IMFs Augmented Dickey-Fuller test.

IMF	ADF Statistic	p-value	1%	5%	10%
IMF_1	-36.511	0.0	-3.4317	-2.8621	-2.5671
IMF_2	-20.6102	0.0	-3.4317	-2.8621	-2.5671
IMF_3	-15.6381	0.0	-3.4317	-2.8621	-2.5671
IMF_4	-15.2507	0.0	-3.4317	-2.8621	-2.5671
IMF_5	-15.4237	0.0	-3.4317	-2.8621	-2.5671
IMF_6	-7.2072	0.0	-3.4317	-2.8621	-2.5671
IMF_7	-5.5054	0.0	-3.4317	-2.8621	-2.5671
IMF_8	-0.8995	0.7881	-3.4317	-2.8621	-2.5671
IMF_9	-15.7164	0.0	-3.4317	-2.8621	-2.5671

Furthermore, Tables 4.1a, 4.1b, and 4.1c report the statistical properties of the corresponding IMFs. In addition, in order to verify that the generated IMF components are stationary, the ADF (Augmented Dickey-Fuller) test [20] was applied. This is a statistical significance test, regarding the stationarity of the time series, which provides a p-value that allows relevant inferences.

The obtained p-values are reported in Tables 4.2a, 4.2b, and 4.2c, which reveal that all IMF components except IMF_8 of each currency pair, are stationary (p-value ≤ 0.05). On the other hand, the initial time series (Figs 4.2a, 4.2c, 4.2b) are clearly non- stationary, as there are different trend levels and big reversals throughout the dataset.

4.3 Parameter configuration

The parameter configuration of the proposed system is reported in Tables 4.3 - 4.4. Regarding the *train_test split*, the common approach of using 70% for training, 10% for validation, and the remaining 20% for testing was adopted. Regarding the sliding window size, the value of 30 was selected, because daily trading requires a time interval is needed that is neither too big nor too small. Note that the proposed system differs from those trading stocks, where the market moves much slower, but the system shall take into consideration possible huge changes that may have happened in the short term. Therefore, a window of size 30 is a wise choice as we show later in the results. Regarding the LSTM architecture, different set-ups were tested before deciding the one that was eventually preferred.

Table 4.3: Parameter details.

Scaler	Scale range	train_set (%)	test_set (%)	validation_set (%)	Sliding window size		
MinMaxScaler	(-1, 1)	70	20	10	30		
Swarm size	Neighborhood radius	Maximum iterations	wlb	wrb	χ	c_1	c_2
50	10	30000	1.0	1.5	0.729	2.05	2.05
Epochs	Optimizer	loss	batch_size				
300	Adam	MAE	10				

Table 4.4: LSTM architecture.

LSTM (units=128)
LSTM (units=256)
Dropout (rate = 0.1)
LSTM (units=128)
Dense (units=1, activation = linear)

4.4 Performance metrics

The accuracy of the proposed system was assessed according to three accuracy metrics, namely the Mean Absolute Error (MAE), the Mean Squared Error (MSE), and the Mean Absolute Percentage Error (MAPE).

$$MAE = \sum_{i=1}^{n} \frac{|\hat{y}_i - y_i|}{n} \tag{4.2}$$

$$MSE = \sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n} \tag{4.3}$$

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} |\frac{\hat{y}_i - y_i}{y_i}| \tag{4.4}$$

In addition, the following four trading metrics were used in order to asses the system's trading performance. First of all, the success rate which is also denoted as *winning rate* (%), was considered. Secondly, there is the *Profit/Loss ratio*, which measures how well a trading system is performing. Obviously, the higher the ratio the better the better the system is. For example, if a system achieved a winning average of $750 per trade and an average loss (over the same time) of $250 per trade, then the profit/loss ratio would be 3:1. Next, the system takes into account the *profit*, which refers to gains, and losses. These quantities are measured in pips. The pips are then converted to the base currency's value. For instance, let the EUR/USD exchange rate be 1.1130. Suppose, a buy position is opened at 1.1130 and, after a day, the price goes to 1.1160. Let a sell signal occurs, and the trade is closed. Then, the profit is 30 pips (=1.1160 - 1.1130). Let the trader's account balance be 350000 EUR. Then, the conversion of profit from pips to EUR is as follows:

Value of pip in US Dollars: $350000 * 0.0001 = \$35$ USD

Value of pip in Euro: $\frac{35}{1.1130} = 31.45$ EUR

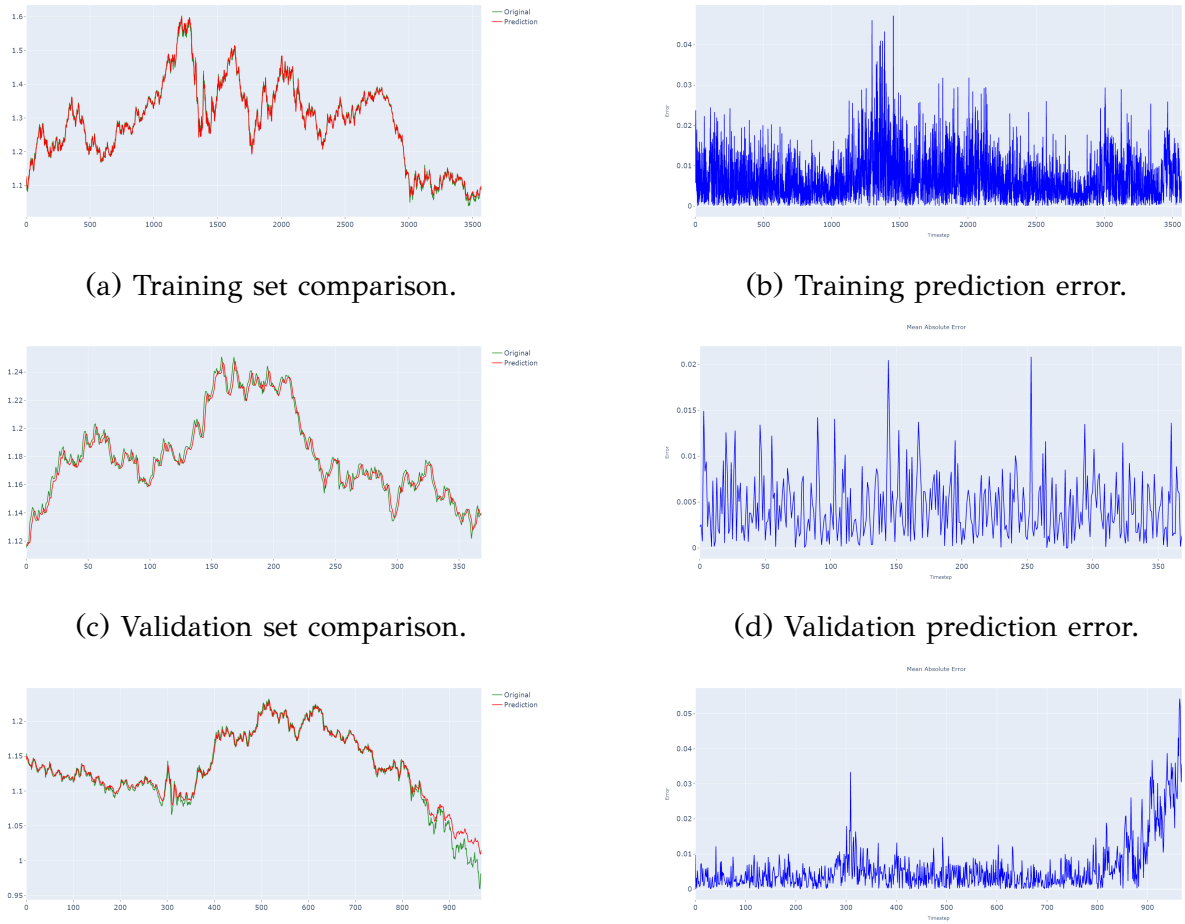
Trade Profit: $30 * 31.45 = 943.4$ EUR

Lastly, the *Maximum DrawDown* (MDD) is a specific measure that looks for the greatest movement from a high point to a low point, before a new peak is achieved. It is measured in pips and takes only negative values. Evidently, this measure shows the maximum loss the trade has reached, while it was open.

4.5 Experimental results

In this section the obtained results from the experimental analysis are presented. The amount of the obtained results was quite large. For this reason, only the EUR/USD

results are presented in the present section with the other two pairs. EUR/CHF and USD/CHF, reported in Appendix A.2.1, A.2.2 respectively.



(a) Training set comparison.

(b) Training prediction error.

(c) Validation set comparison.

(d) Validation prediction error.

(e) Test set comparison.

(f) Test prediction error.

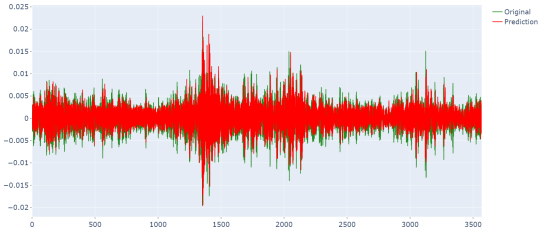
Figure 4.6: EUR/USD, LSTM comparison between real (green) and forecasted(red) prices and prediction errors.

The reported results include comparison of our findings against LSTM, EMD-LSTM, as well as a previously published work based on moving average aggregation and metaheuristic optimization (MA-PSO) [21]. It shall be mentioned that the proposed system's trading performance is compared against MA-PSO, despite the fact that they refer to different time periods. The reported results clearly show that the proposed approach outperforms the LSTM and EMD-LSTM implementations.

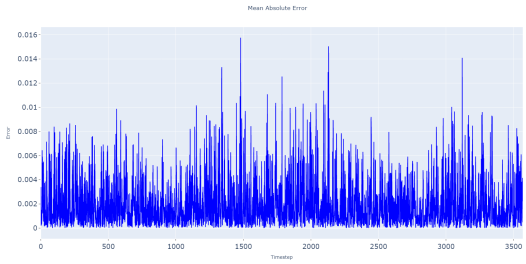
Table 4.5: EUR/USD, LSTM prediction errors

Set	MAE	MSE	MAPE
Train	6.72e-03	3.13e-05	5.24
Validation	4.39e-03	3.13e-05	3.71e-03
Test	5.97e-03	8.84e-05	5.51e-03

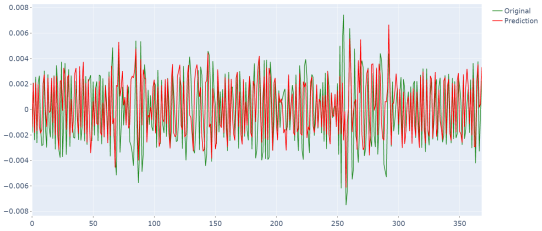
As it can be seen from Fig. 4.6 during the training and validation process the LSTM model seems to perform well. On the other hand, that does not seem to be the case for the test set. In the test set, while the model seemed to perform well throughout the dataset, in the last 100 steps its forecasting ability was drastically decreased. This can also be observed in Table 4.5, where the LSTM's accuracy is being shown. As a result, the system cannot rely on the LSTM model, since its performance is not stable throughout the testing period.



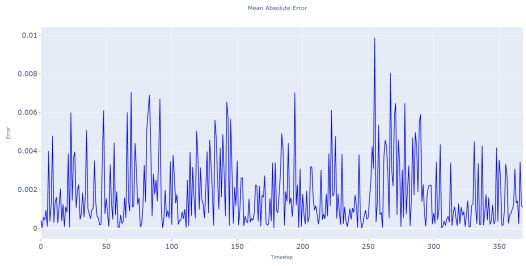
(a) Training set comparison.



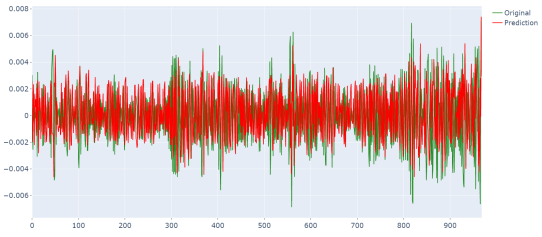
(b) Training set prediction error.



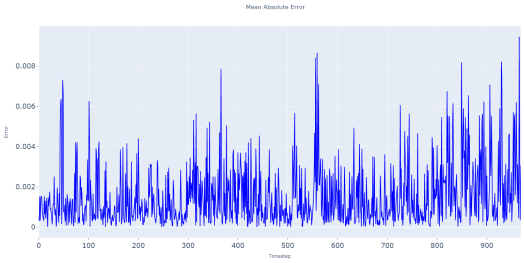
(c) Validation set comparison.



(d) Validation set prediction error.

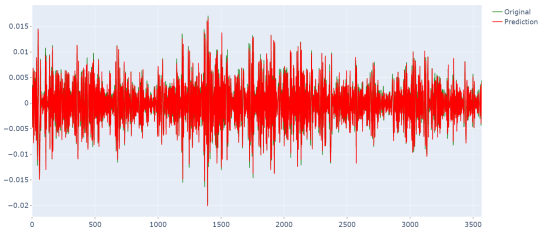


(e) Test set comparison.

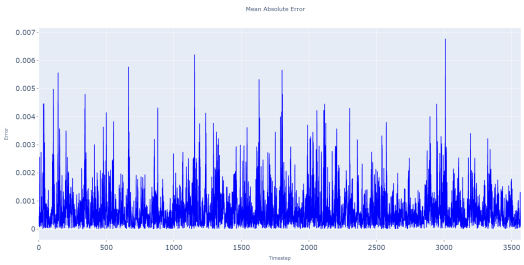


(f) Test set prediction error.

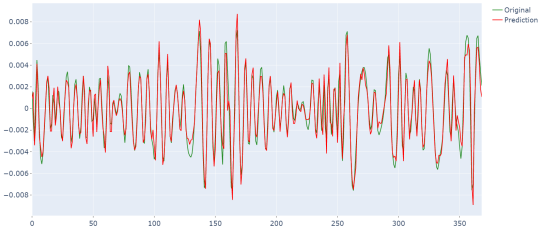
Figure 4.7: EURUSD - IMF_1 real (green) vs predicted (red) and prediction errors.



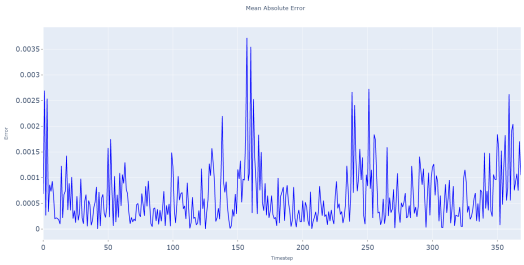
(a) Training set comparison.



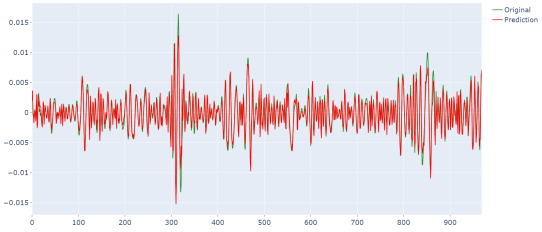
(b) Training set prediction error.



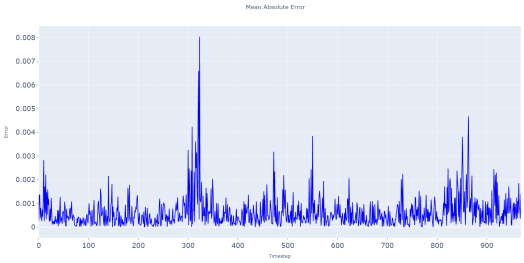
(c) Validation set comparison.



(d) Validation set prediction error.

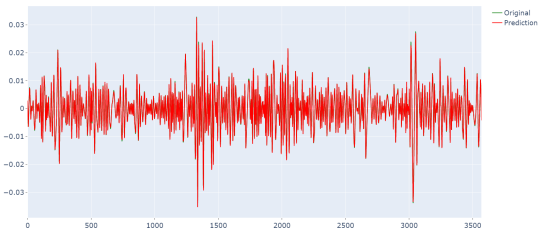


(e) Test set comparison.

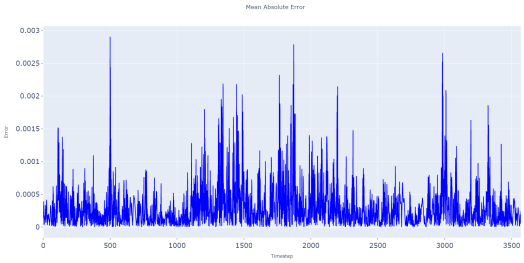


(f) Test set prediction error.

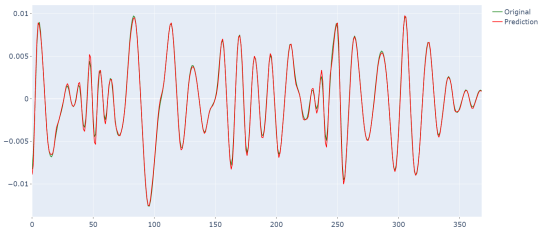
Figure 4.8: EURUSD - IMF_2 real (green) vs predicted (red) and prediction errors.



(a) Training set comparison.

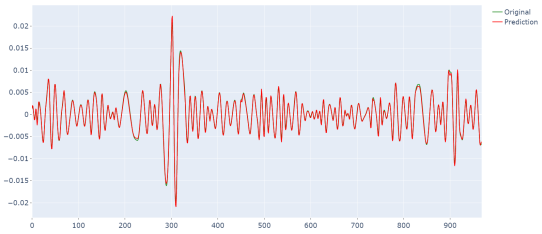


(b) Training set prediction error.

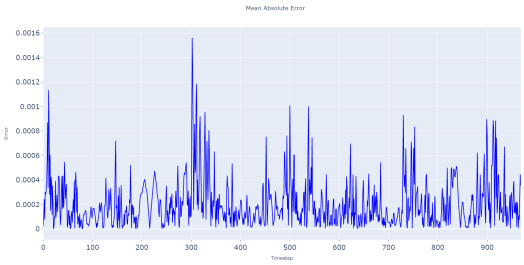


(c) Validation set comparison.

(d) Validation set prediction error.

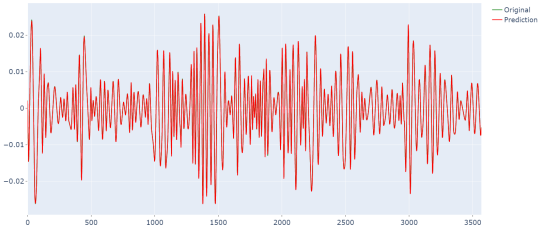


(e) Test set comparison.

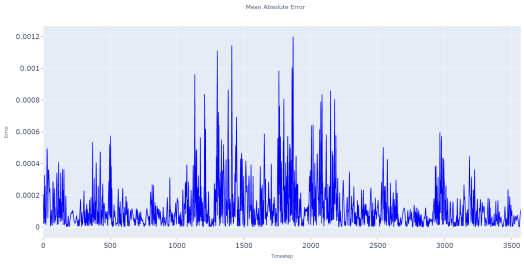


(f) Test set prediction error.

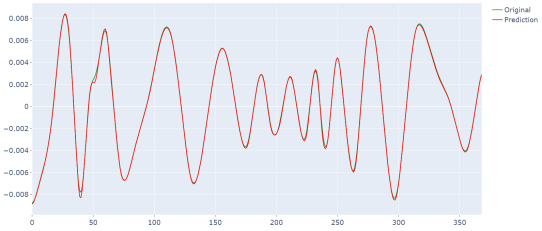
Figure 4.9: EURUSD - IMF_3 real (green) vs predicted (red) and prediction errors.



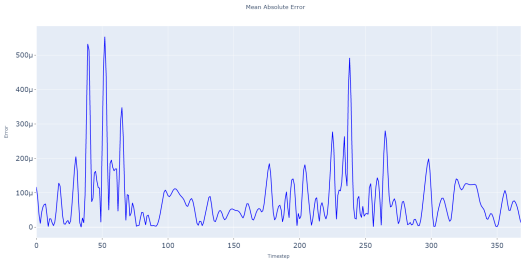
(a) Training set comparison.



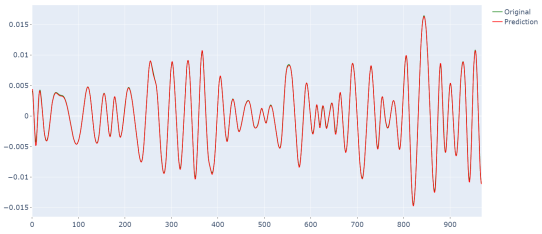
(b) Training set prediction error.



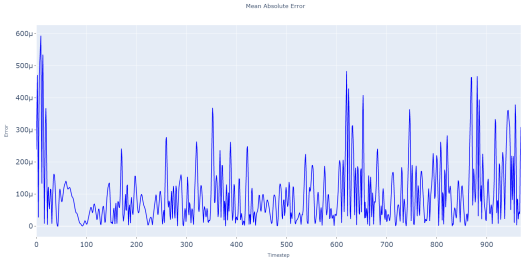
(c) Validation set comparison.



(d) Validation set prediction error.



(e) Test set comparison.

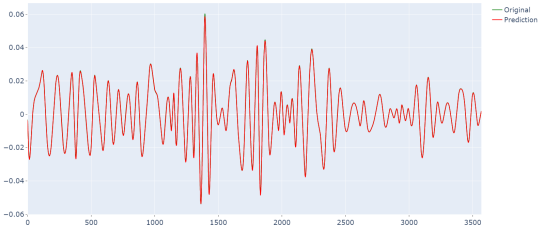


(f) Test set prediction error.

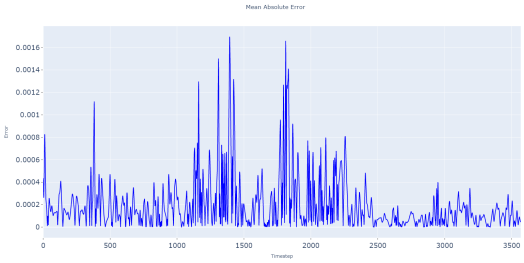
Figure 4.10: EURUSD - IMF_4 real (green) vs predicted (red) and prediction errors.

Next, in Figs. 4.7 - 4.15 and in Tables 4.6a - 4.6c the obtained results for all the generated IMFs are shown. We can clearly see that the EMD-LSTM model predicts the respective IMFs with high accuracy such that one cannot easily distinguish the real (green) from the predicted (red). In addition, the model's performance remains constant throughout the sets (training, validation and test), which is of great significance.

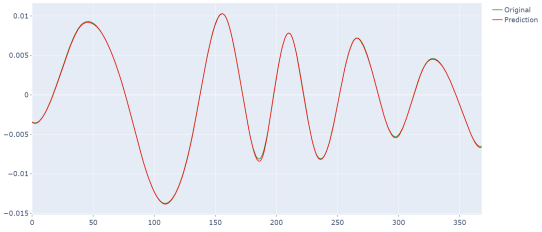
Furthermore, the obtained forecasted prices of the EUR/USD currency pair are computed by adding the IMF components. As it can be seen in Table 4.7 and in Fig. 4.16, the EMD-LSTM model is able to accurately predict the EUR/USD currency pair. The difficult part when predicting exchange rates is to identify the big reversals. These reversals can act like entry or exit points from the market. By accurately identifying these points, the investor expects two things, i.e. (a) enter the market when the trend has changed, which can yield high profits, and (b) exit the market when the trend has changed to the opposite direction. In this situation, the investor's capital is protected and minimizes the risk of loss.



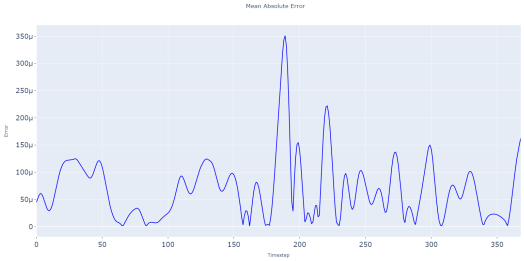
(a) Training set comparison.



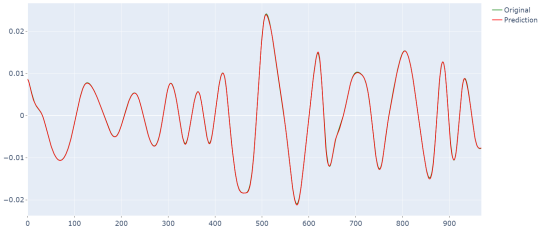
(b) Training set prediction error.



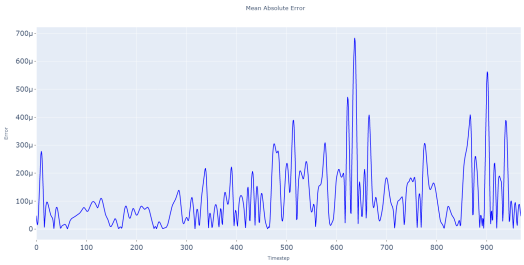
(c) Validation set comparison.



(d) Validation set prediction error.

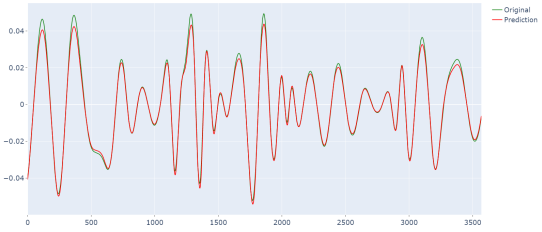


(e) Test set comparison.



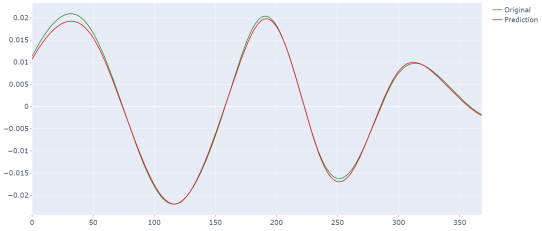
(f) Test set prediction error.

Figure 4.11: EURUSD - IMF_5 real (green) vs predicted (red) and prediction errors.

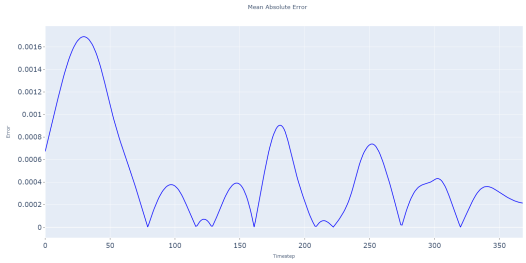


(a) Training set comparison.

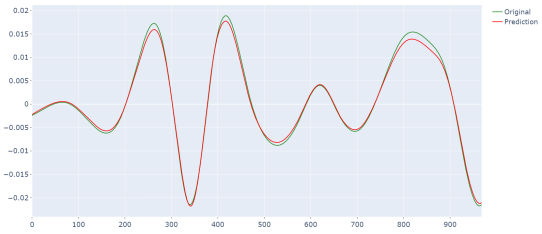
(b) Training set prediction error.



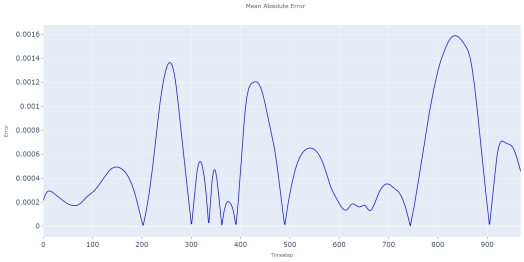
(c) Validation set comparison.



(d) Validation set prediction error.

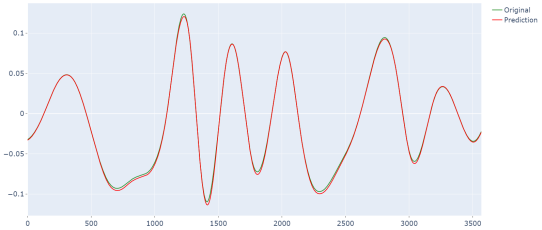


(e) Test set comparison.

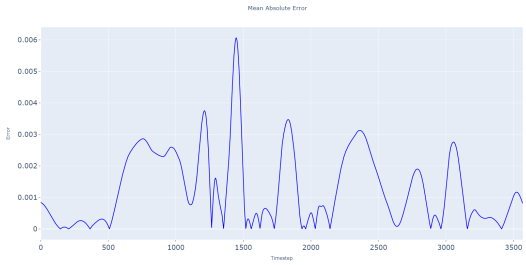


(f) Test set prediction error.

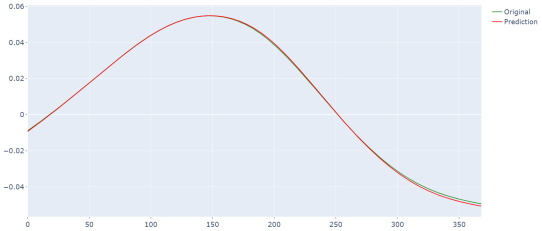
Figure 4.12: EURUSD - IMF_6 real (green) vs predicted (red) and prediction errors.



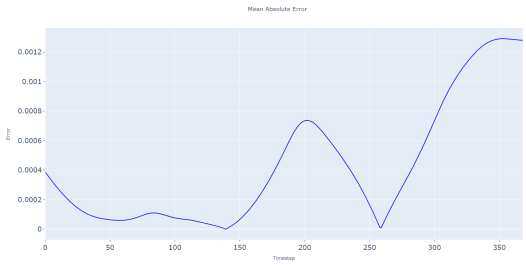
(a) Training set comparison.



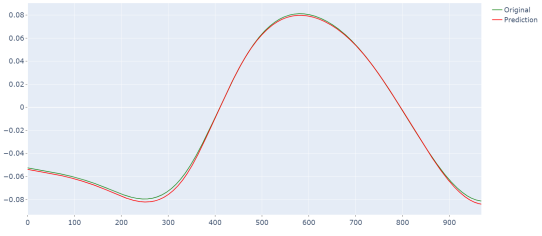
(b) Training set prediction error.



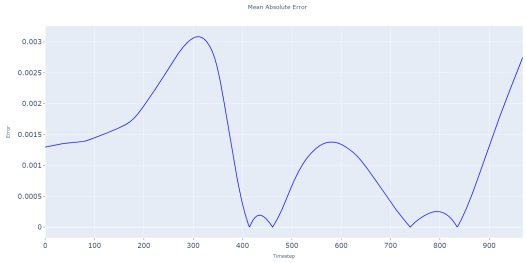
(c) Validation set comparison.



(d) Validation set prediction error.

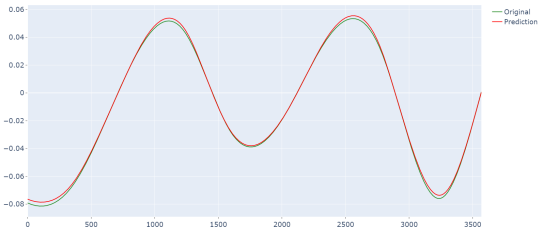


(e) Test set comparison.

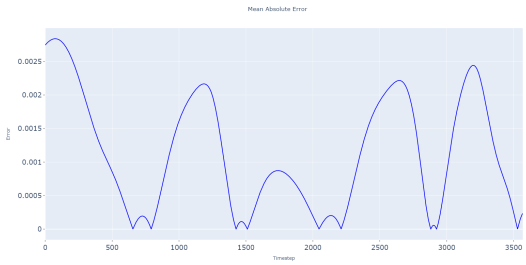


(f) Test set prediction error.

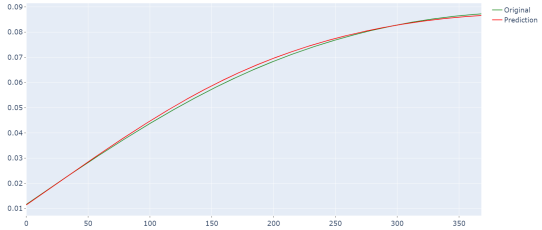
Figure 4.13: EURUSD - IMF_7 real (green) vs predicted (red) and prediction errors.



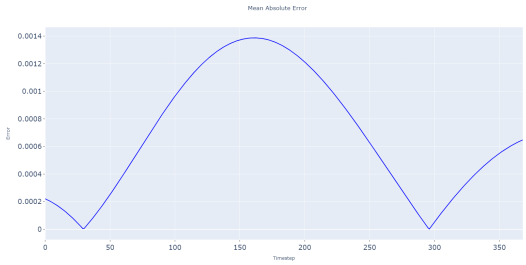
(a) Training set comparison.



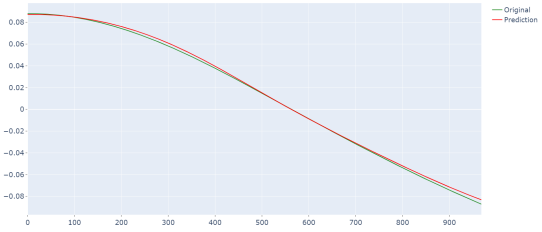
(b) Training set prediction error.



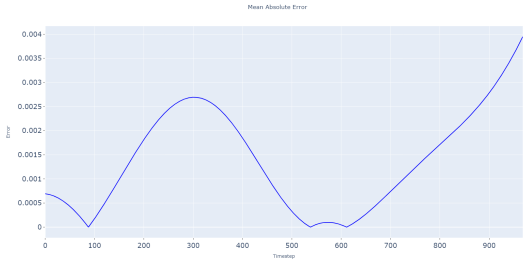
(c) Validation set comparison.



(d) Validation set prediction error.

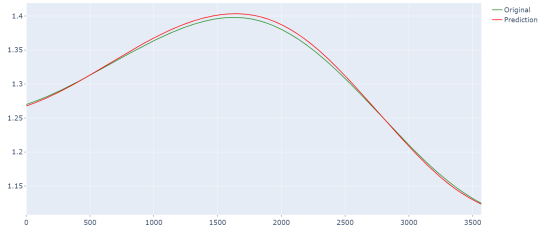


(e) Test set comparison.

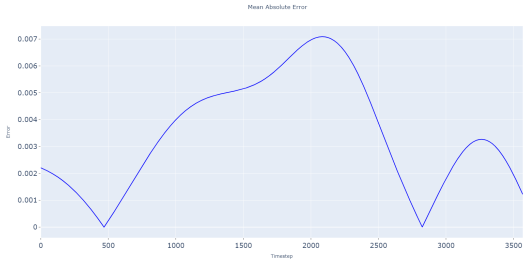


(f) Test set prediction error.

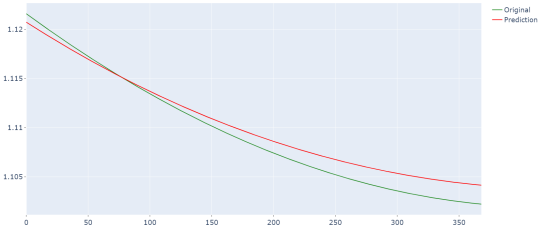
Figure 4.14: EURUSD - IMF_8 real (green) vs predicted (red) and prediction errors.



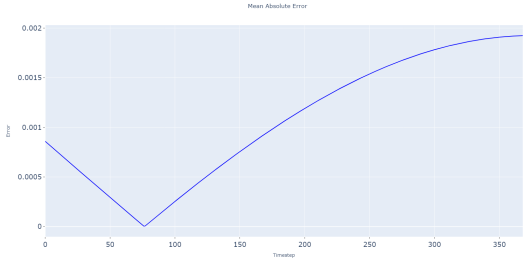
(a) Training set comparison.



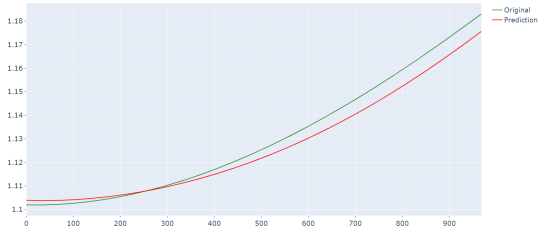
(b) Training set prediction error.



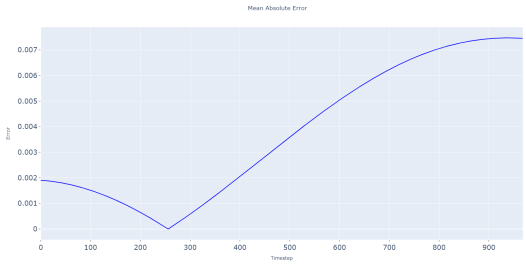
(c) Validation set comparison.



(d) Validation set prediction error.



(e) Test set comparison.



(f) Test set prediction error.

Figure 4.15: EURUSD - IMF_9 real (green) vs predicted (red) and prediction errors.

Table 4.6: EUR/USD, IMF results

(a) Train set prediction error.

IMF	MAE	MSE	MAPE
IMF_1	1.92e-03	7.75e-06	1.74
IMF_2	6.7e-04	9.69e-07	4593192111.74
IMF_3	2.96e-04	1.89e-07	4593192111.74
IMF_4	1.26e-04	3.59e-08	4593192111.74
IMF_5	1.99e-04	9.19e-08	4593192111.74
IMF_6	1.23e-03	3.33e-06	100949277.32
IMF_7	1.28e-03	3.04e-06	3.57e-02
IMF_8	1.13e-03	2.02e-06	3.57e-02
IMF_9	3.48e-03	1.64e-05	3.57e-02

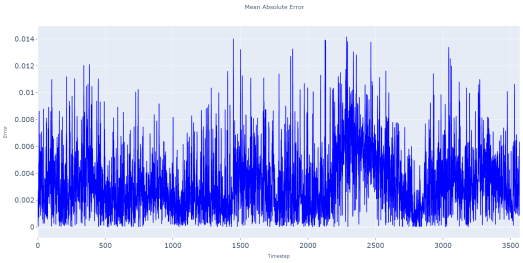
(b) Validation set prediction error.

IMF	MAE	MSE	MAPE
IMF_1	1.79e-03	6.30e-06	1.3221
IMF_2	6.33e-04	7.20e-07	1.3221
IMF_3	2.04e-04	8.55e-08	1.3221
IMF_4	8.26e-05	1.38e-08	1.3221
IMF_5	6.86e-05	7.96e-09	1.3221
IMF_6	4.67e-04	4.06e-07	1.3221
IMF_7	4.17e-04	3.40e-07	1.3221
IMF_8	6.87e-04	6.80e-07	1.3221
IMF_9	1.05e-03	1.50e-06	1.3221

(c) Test set prediction error.

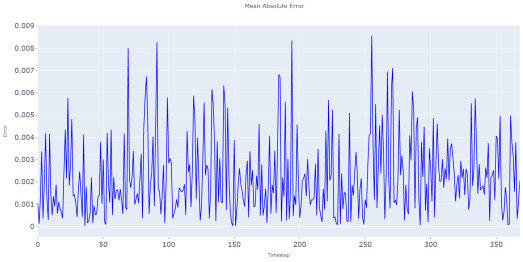
IMF	MAE	MSE	MAPE
IMF_1	1.51e-03	4.63e-06	1.3221
IMF_2	6.22e-04	8.47e-07	1.3221
IMF_3	2.06e-04	8.03e-08	1.3221
IMF_4	9.80e-05	1.79e-08	1.3221
IMF_5	1.12e-04	2.33e-08	1.3221
IMF_6	5.36e-04	4.63e-07	1.3221
IMF_7	1.25e-03	2.34e-06	1.3221
IMF_8	1.37e-03	2.96e-06	1.3221
IMF_9	3.75e-03	2.08e-05	1.3221

(a) Training set comparison.



(b) Training prediction error.

(c) Validation set comparison.



(d) Validation prediction error.

(e) Test set comparison.

(f) Ttest prediction error.

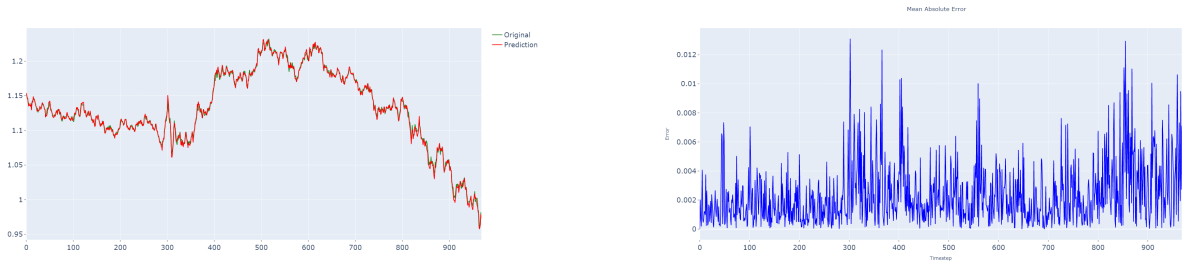
Figure 4.16: EUR/USD, EMD-LSTM comparison between real (green) and forecasted(red) prices and prediction errors.

Table 4.7: EUR/USD, EMD-LSTM prediction errors

Set	MAE	MSE	MAPE
Train	3.77e-03	2.25e-05	2.92e-03
Validation	2.35e-03	9.42e-06	1.19e-03
Test	3.89e-03	2.32e-05	3.45e-03

Table 4.8: EUR/USD, EMD-LSTM-PSO prediction errors

Set	MAE	MSE	MAPE
Test	**2.34e-03**	**1.00e-05**	**2.10e-03**



(a) Test set comparison.

(b) Test set prediction error.

Figure 4.17: EMD-LSTM-PSO, EUR/USD comparison

Next, it was investigated if it was possible to further improve the system's forecasting ability. For that reason, as mentioned in Chapter 3 the PSO algorithm was used. As seen in Fig. 4.17 and Table 4.8, the implementation of the PSO algorithm had successfully minimized the prediction error. In addition, as described in Section 4.6, the EMD-LSTM-PSO model is characterized for its increased profits.

4.6 Simulation

In order to measure the system's actual performance, a simulation environment was implemented, in which the proposed system was tested for three months (64 actual trading days), starting from 06/07/2022 to 03/10/2022. As mentioned before, each IMF has equal length with the initial time series, equal to 5000. The last ws (window size) observations are used to predict the next closing price. The reason for this approach was to verify two things, i.e., (a) how the change of IMFs at every step (day) affects the system and (b) the different vector of weights generated by the PSO algorithm in every run. Thus, the goal was to verify if such a technique is profitable. Let $D = \{d_1, d_2, \ldots, d_{64}\}$ be the set of the aforementioned dates. For each $d_i \in D, i = 1, 2, \ldots, 64$, a set consisting of the past 5000 observations of day i is considered. The created set has the following form:

$$
\begin{aligned}
s_1 &= \{cp_1, cp_2, \ldots, cp_{5000}\} \\
s_2 &= \{cp_2, cp_3, \ldots, cp_{5001}\} \\
s_3 &= \{cp_3, cp_4, \ldots, cp_{5002}\} \\
&\vdots \\
s_{64} &= \{cp_{64}, cp_{65}, \ldots, cp_{5064}\},
\end{aligned}
$$

where cp is the close price of the day i. Next, each s_i is given as input to the system, which produces a forecast $\hat{f}_i$ for the closing price of day $i+1$. The produced vector containing the forecasts has the following form:

$$f = \left(\hat{f}_{5001}, \hat{f}_{5002}, \ldots, \hat{f}_{5065}\right). \tag{4.5}$$

Furthermore, the simulation for the EMD-LSTM-PSO was applied, as it is the model with the best forecasting accuracy. For the analysis, the EUR/USD currency pair was selected, since is the most traded currency pair in the market. As shown in Figs. 4.18a and 4.18b, the proposed approach accurately follows the trend of the real prices even in the big reversals. As a result, the system yields high profits as reported in Table 4.10. In addition, it is obvious that the proposed model outperforms by far the MA-PSO model.

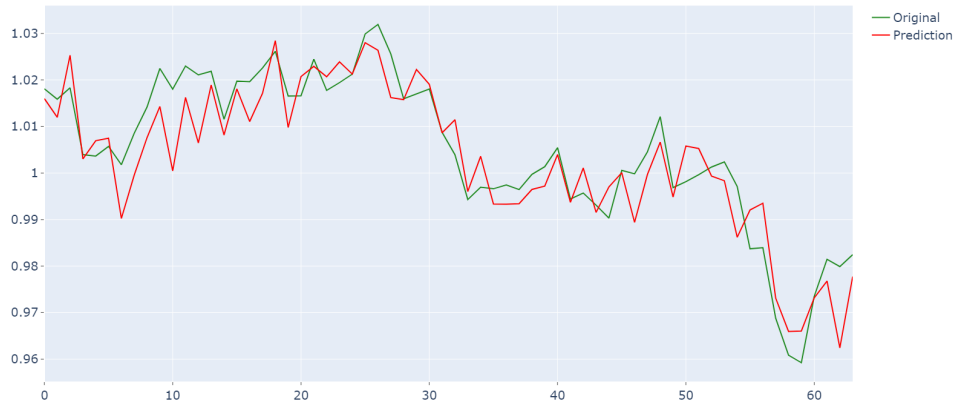
Table 4.9: EUR/USD, EMD-LSTM-PSO, MA-PSO simulation error

Model	MAE	MSE	MAPE
MA-PSO	6.24e-03	5.03e-03	4.36e-03
EMD-LSTM-PSO	**5.07e-03**	**4.05e-05**	**5.24e-03**

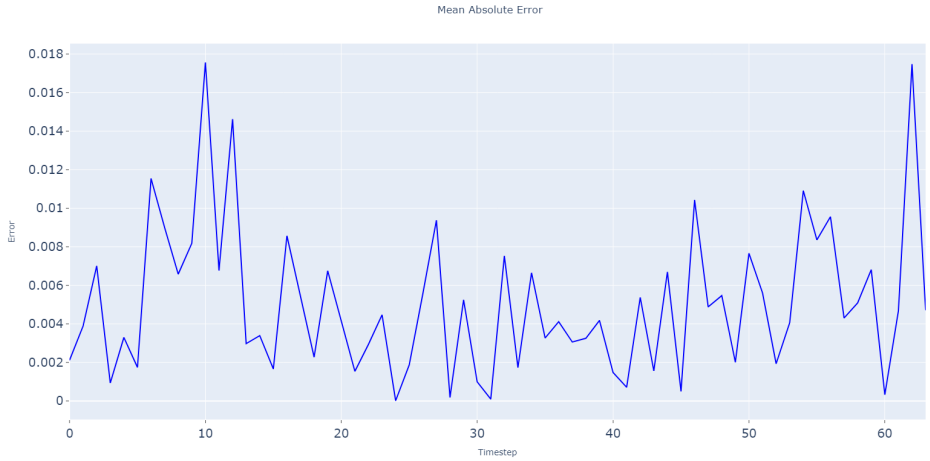
Table 4.10: Trading performance of the proposed and competing models.

Model	Trades	Winning Trades	Winning Rate (%)	P/L Ratio	Profit(in pips)	MDD(in pips)
MA-PSO	22	16	72.8	2.469614	737.3	-183.5
EMD-LSTM-PSO	**36**	**31**	**86.11**	**3.35**	**2197.7**	**-95.9**

As being shown in Table 4.9 during that period the system achieved a $MAE = 5e-3$, which can be translated to average deviation of 50 pips. Taking into account that EUR/USD for the last several years moves between 70 to 100 pips daily [22], the proposed model exhibits limited risk in addition to its great trading performance. We need to clarify in this point that the considered work was tested during the second term of $2018(06/2018 - 12/2018)$.



(a) Simulation comparison between real (green) and predicted (red).



(b) Mean absolute error during the simulation period.

Figure 4.18: EMD-LSTM-PSO EUR/USD simulation results.

Chapter 5

Epilogue

5.1 Conclusions

Nowadays, the necessity to perform human tasks with the minimum cost and at higher speed, along with the need to process voluminous data, justifies the expansion of computational intelligent models in various scientific fields such as finance. In addition, financial forecasting is inherently connected with the high degree of uncertainty ruling the modern world, thereby computational intelligent models can be efficient alternatives to traditional models.

The present thesis introduced an automated trading system based on signal decomposition and deep learning. The aggregate forecast was optimized by the particle swarm optimization algorithm. Proper parameters were identified and reported. The proposed trading system led to high profit on the tested cased, with the minimum risk of loss. In terms of statistical performance, the proposed model outperforms the LSTM, EMD-LSTM, and also relevant previous work with moving averages aggregation.

In addition, in order to test the developed system, three different FOREX currency pairs were selected, EUR/USD, EUR/CHF and USD/CHF. The proposed system is capable of accurately predicting the time series. Even for the simulation performed

for the EUR/USD currency pair it was found that the system yielded increased profits, with minimum loss (5 lost trades).

To conclude, traders should experiment beyond the boundaries of traditional models. Their trading decisions should be based on forward-looking expectations from models and strategies optimized withing the framework of a hybrid trading and statistical approach. Nonetheless, there are still many paths to be taken in the search of efficient calibration of computational intelligent models for financial and economic forecasting tasks.

5.2 Future work

As a follow-up to the presented work, a series of distinct directions can be explored in order to try to improve the developed system. Some of the most relevant are the following:

- Improve system's execution time, in order to experiment with smaller timeframe.
- Take into account technical analysis indicators.
- Introduction of a leverage mechanism. This would be interesting in order to evaluate how the system addresses inherent potential risks.
- Explore the usage of NLP methods, in order to process all the latest political and economical factors, which can greatly affect the market.

Bibliography

[1] BIS. Foreign exchange turnover in april 2019. [Online]. Available: https://www.bis.org/statistics/rpfx19_fx.htm

[2] IBM. Machine learning. [Online]. Available: https://www.ibm.com/cloud/learn/machine-learning

[3] I. C. Education. Neural networks. [Online]. Available: https://www.ibm.com/cloud/learn/neural-networks

[4] CFI. Machine learning (in finance). [Online]. Available: https://corporatefinanceinstitute.com/resources/knowledge/other/machine-learning-in-finance/

[5] Metatrader5. [Online]. Available: https://www.metatrader5.com/

[6] J. C. Bretton woods agreement. [Online]. Available: https://www.investopedia.com/terms/b/brettonwoodsagreement.asp

[7] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, N.-C. Yen, C. C. Tung, and H. H. Liu, "The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis," *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 454, no. 1971, pp. 903–995, 1998.

[8] N. E. Huang, M.-L. C. Wu, S. R. Long, S. S. P. Shen, W. Qu, P. Gloersen, and K. L. Fan, "A confidence limit for the empirical mode decomposition and hilbert spectral analysis," *Proceedings: Mathematical, Physical and Engineering Sciences*, vol. 459, no. 2037, pp. 2317–2345, 2003.

[9] G. Rilling, P. Flandrin, and P. Gonçalves, "On empirical mode decomposition and its algorithms," *Proceedings of IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing NSIP-03*, vol. 3, 2003.

[10] R. Rato, M. Ortigueira, and A. Batista, "On the hht, its problems, and some solutions," *Mechanical Systems and Signal Processing*, vol. 22, no. 6, pp. 1374–1394, 2008.

[11] W. Mcculloch and W. Pitts, "A logical calculus of ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 127–147, 1943.

[12] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain." *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[13] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[14] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[15] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] H. S and S. J, "Lstm can solve hard long time lag problems," 1997, pp. 473–479.

[18] E. R.C. and K. J., "A new optimizer using particles swarm theory," in *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*. IEEE.

[19] C. M. and K. J., "The particle swarm: Explosion, stability and convergence in a multi-dimensional complex space," *IEEE Trans. on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[20] Y.-W. Cheung and K. S. Lai, "Lag order and critical values of the augmented dickey–fuller test," *Journal of Business & Economic Statistics*, vol. 13, no. 3, pp. 277–280, 1995.

[21] M. G. Papatsimpas, I. Lykogiorgos, and K. E. Parsopoulos, "Forex trading model based on forecast aggregation and metaheuristic optimization," ser. SETN 2020. Association for Computing Machinery, 2020, p. 215–223.

[22] David. Average daily range. [Online]. Available: https://offbeatforex.com/forex-average-daily-range-table/

[23] B. W. Kernighan and D. M. Ritchie, *The C programming language*, 2006.

[24] F. Chollet *et al*. (2015) Keras.

[25] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al*., "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, pp. 2825–2830, 2011.

[27] W. McKinney *et al*., "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, vol. 445. Austin, TX, 2010, pp. 51–56.

[28] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in science & engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[29] K. Reitz. Requests. [Online]. Available: https://requests.readthedocs.io/en/latest/

[30] H. Eijs. Pycryptodome. [Online]. Available: https://github.com/Legrandin/pycryptodome

[31] D. Laszuk, "Python implementation of empirical mode decomposition algorithm," https://github.com/laszukdawid/PyEMD, 2017.

[32] Telegram FZ LLC and Telegram Messenger Inc., "Telegram." [Online]. Available: https://telegram.org

[33] P. T. Inc. (2015) Collaborative data science. Montreal, QC. [Online]. Available: https://plot.ly

Appendix A

Appendix

A.1 Software requirements

The thesis code was written in the Python (3.7.9) [63] and the C [23] programming languages. Regarding the Python code the following libraries have been used to implement the program:

- MetaTrader5 [24] is a package for Python is designed for convenient and fast obtaining of exchange data via interprocessor communication directly from the MetaTrader 5 terminal.
- Keras [24] is an open-source machine learning framework. Keras is an exceptionally useful and flexible libraries for the constructing and training of artificial neural network architecture.
- Numpy [25] library is fundamental package for scientific computing with Python. Numpy is essential for scientific computations and matrices' creation and manipulation.
- Scikit-learn [26] is a simple and efficient library that contains tools for machine learning, data analysis, and scientific computations.

- Pandas [27] is a fast, powerful, flexible, and easy to use open-source data analysis and manipulation tool.
- Matplotlib [28] is a comprehensive library for creating static, animated, and interactive visualizations.
- Requests [29] is an elegant and simple HTTP library for Python.
- PyCryptodome [30] is a self-contained Python package of low-level cryptographic primitives.
- PyEMD [31] is a Python implementation of Empirical Mode Decomposition (EMD) and its variations.
- Telegram [32] is a messaging app with a focus on speed and security, it's superfast, simple and free.
- Plotly [33] is graph library which makes interactive, publication-quality graphs.

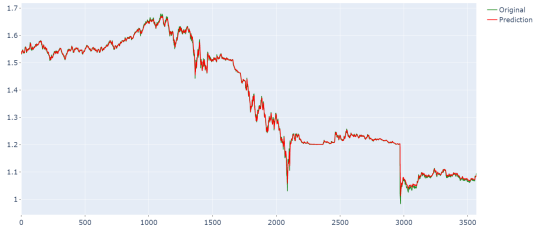
A.2 Detailed results

In this section and more specifically in subsections A.2.1, A.2.2 the detailed results for both EUR/CHF and USD/CHF currency pairs are shown. As it can be seen below, the proposed system accurately predicts currency pairs other than the EUR/USD currency pair. As a result, it can be used as a general trading system for any currency pair.

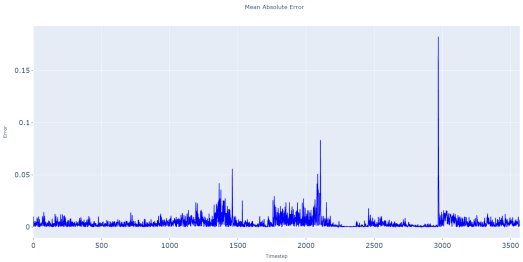
A.2.1 EUR/CHF

In this section the results for the EUR/CHF currency pair are presented. In this case too, the proposed approach is compared against the standard LSTM, the EMD-LSTM implementations.

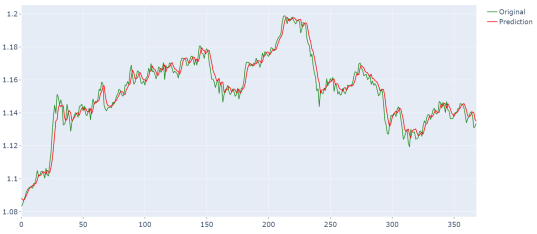
To begin with, the LSTM model behaves as in the EUR/USD case, where it performed well in the training and the validation set. However, during the testing period its generalization ability was unstable. In Fig. A.1 and in Table A.1 it can be observed what was just described.



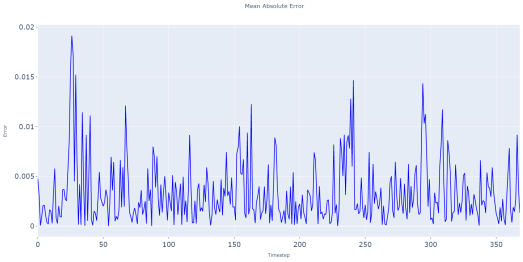
(a) Training set comparison.



(b) LSTM, training prediction error.

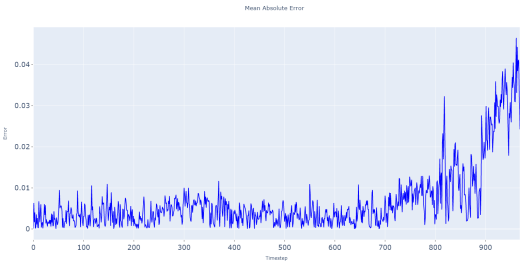


(c) Validation set comparison.



(d) Validation prediction error.

(e) Test set comparison.



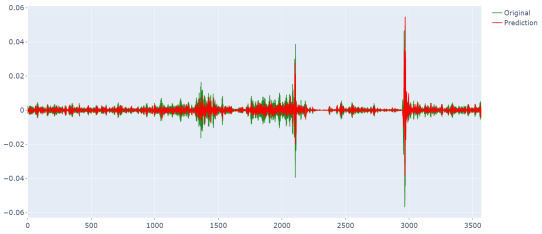
(f) test prediction error.

Figure A.1: EUR/CHF, LSTM comparison between real (green) and forecasted(red) prices and prediction errors.

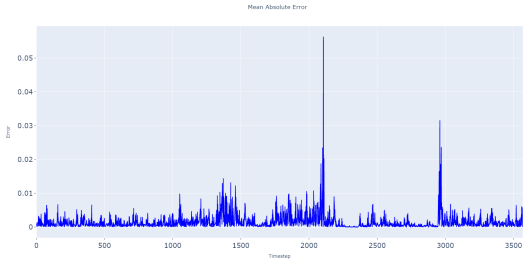
Table A.1: EUR/CHF, LSTM prediction errors

Set	MAE	MSE	MAPE
Train	4.10e-3	5.37e-05	3.11e-03
Validation	3.27e-03	2.03e-05	2.84e-03
Test	6.66e-03	1.05e-04	6.46e-03

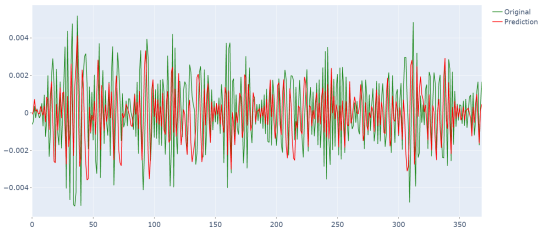
Next, in Figs. A.2 - A.11 and in Tables A.2a - A.2c the proposed model's results for each IMF are shown. We can clearly see that the EMD-LSTM approach accurately predicts the IMF components. In addition, for the case of EUR/CHF the number of the generated IMFs differs from the one of the EUR/USD currency pair. As mentioned in Section 2.2, the number of IMFs depends on the input signal.



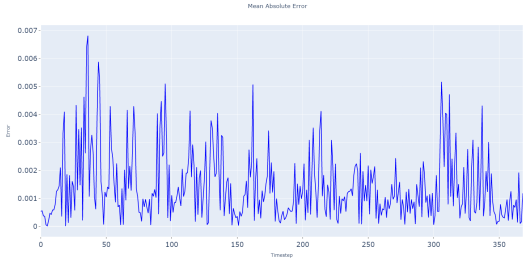
(a) Training set comparison.



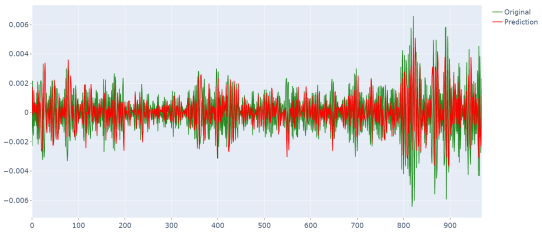
(b) Training set prediction error.



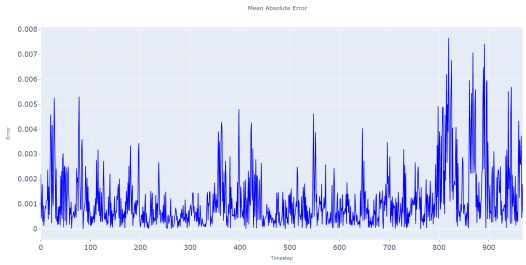
(c) Validation set comparison.



(d) Validation set prediction error.



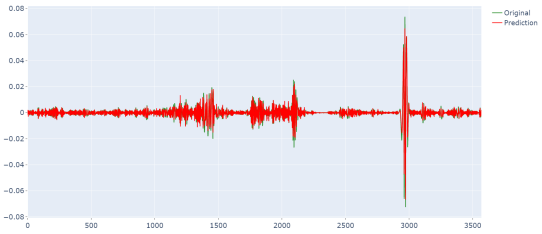
(e) Test set comparison.



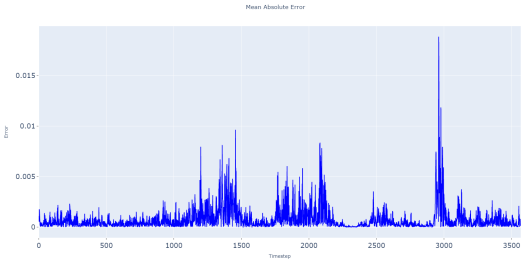
(f) Test set prediction error.

Figure A.2: EURCHF - IMF_1 real (green) vs predicted (red) and prediction errors.

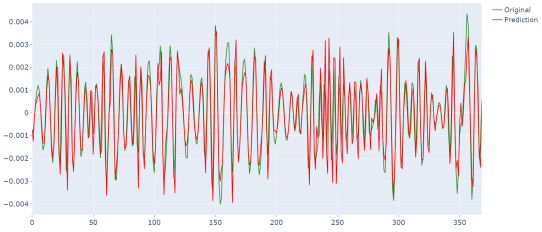
Furthermore, although that the currency pair has changed, the model's behavior remains the same. In the first IMF components, which are the high frequency components, the prediction error can easily be seen. On the other hand, in the low frequency components we can see that someone cannot easily distinguish the difference between the forecasted (red) and the real (green) values.



(a) Training set comparison.

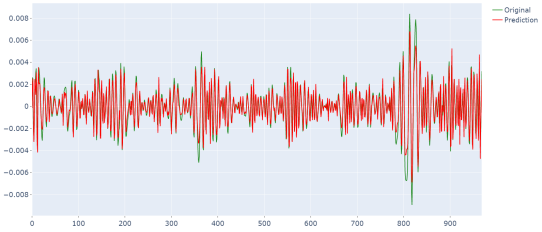


(b) Training set prediction error.

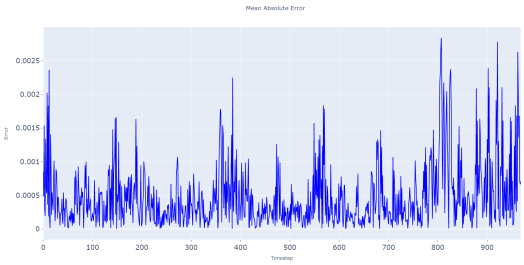


(c) Validation set comparison.

(d) Validation set prediction error.

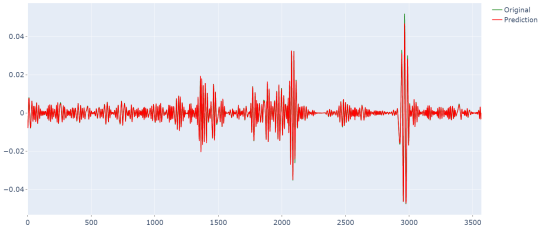


(e) Test set comparison.

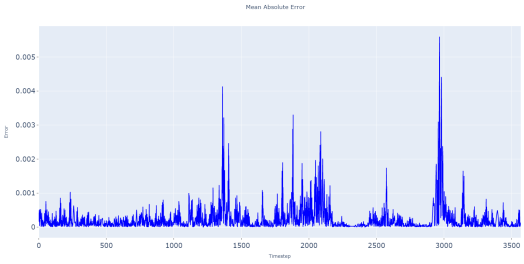


(f) Test set prediction error.

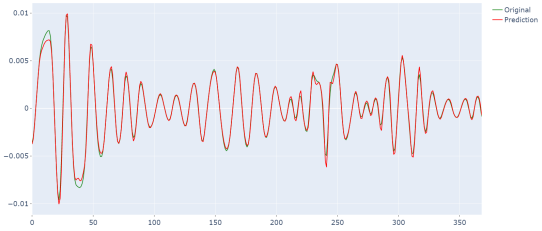
Figure A.3: EURCHF - IMF_2 real (green) vs predicted (red) and prediction errors.



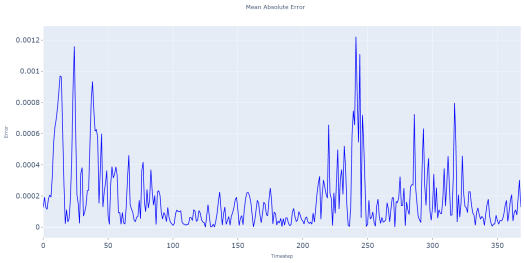
(a) Training set comparison.



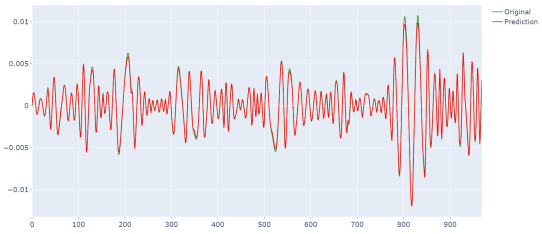
(b) Training set prediction error.



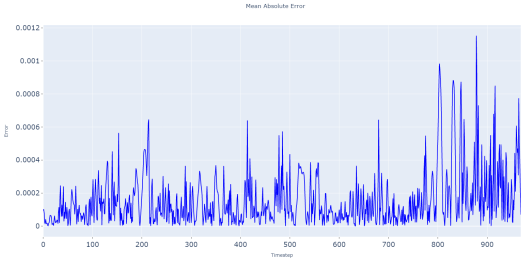
(c) Validation set comparison.



(d) Validation set prediction error.

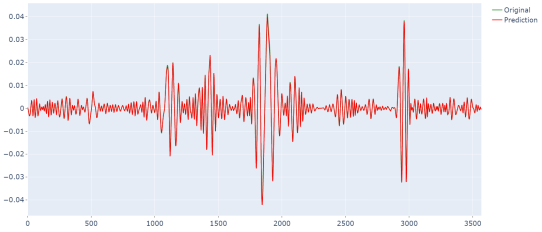


(e) Test set comparison.

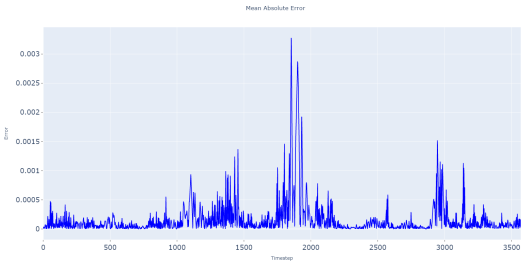


(f) Test set prediction error.

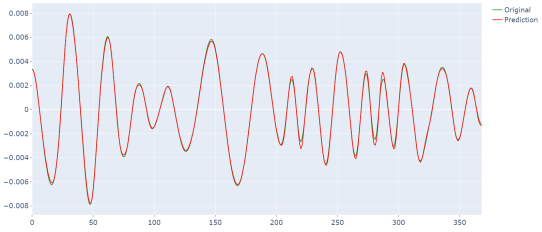
Figure A.4: EURCHF - IMF_3 real (green) vs predicted (red) and prediction errors.



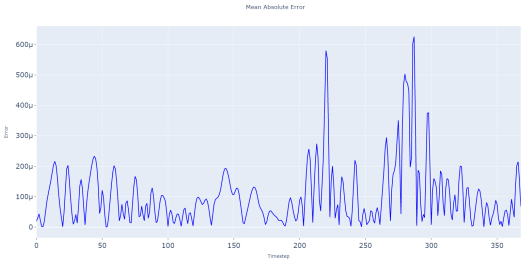
(a) Training set comparison.



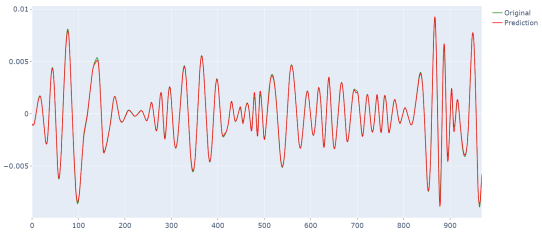
(b) Training set prediction error.



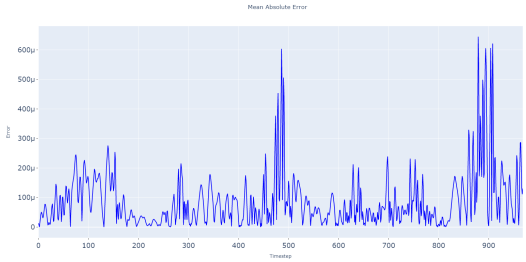
(c) Validation set comparison.



(d) Validation set prediction error.

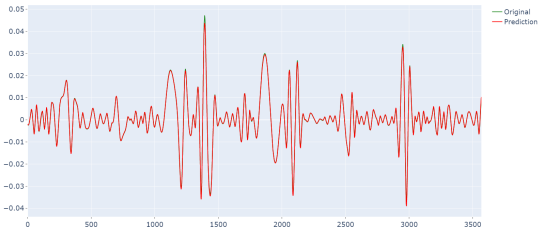


(e) Test set comparison.



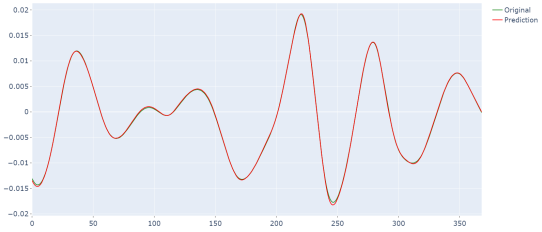
(f) Test set prediction error.

Figure A.5: EURCHF - IMF_4 real (green) vs predicted (red) and prediction errors.

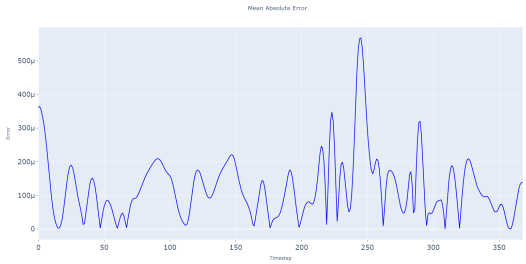


(a) Training set comparison.

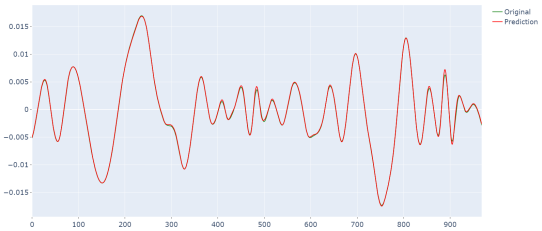
(b) Training set prediction error.



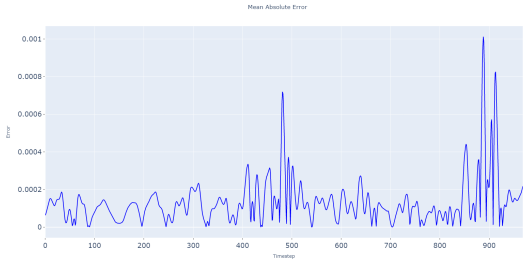
(c) Validation set comparison.



(d) Validation set prediction error.

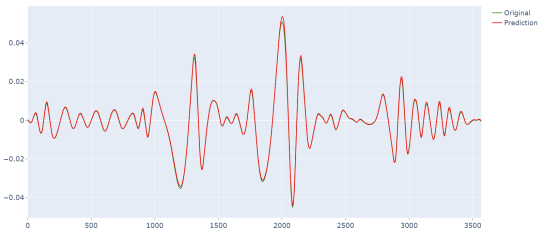


(e) Test set comparison.



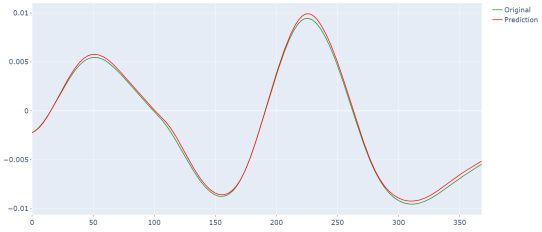
(f) Test set prediction error.

Figure A.6: EURCHF - IMF_5 real (green) vs predicted (red) and prediction errors.

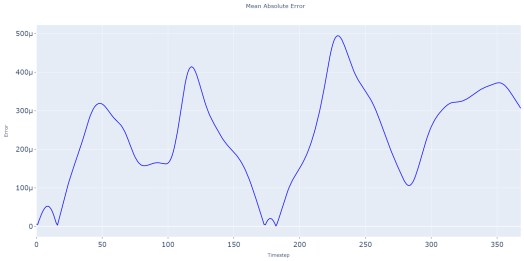


(a) Training set comparison.

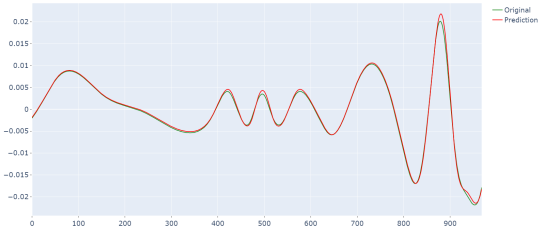
(b) Training set prediction error.



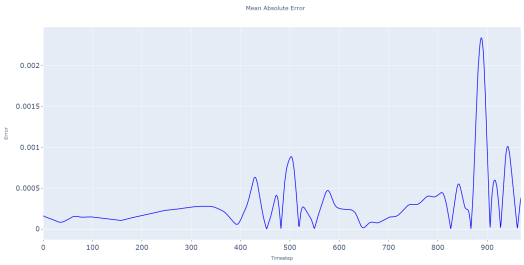
(c) Validation set comparison.



(d) Validation set prediction error.

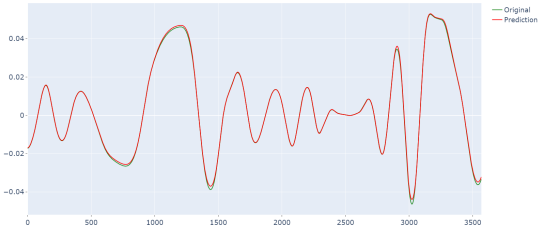


(e) Test set comparison.



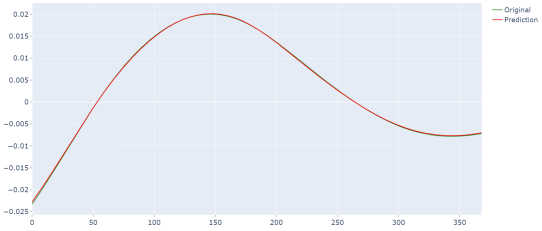
(f) Test set prediction error.

Figure A.7: EURCHF - IMF_6 real (green) vs predicted (red) and prediction errors.

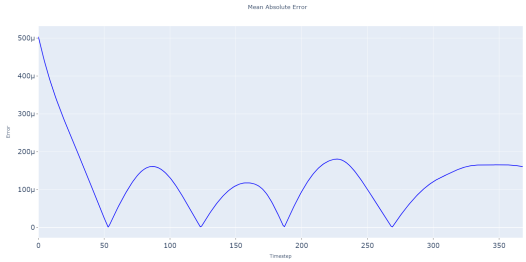


(a) Training set comparison.

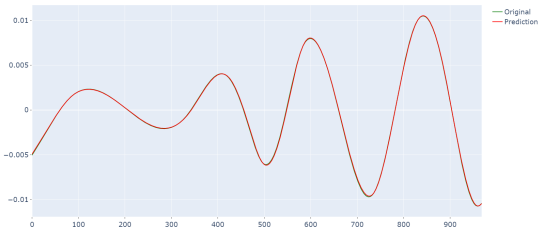
(b) Training set prediction error.



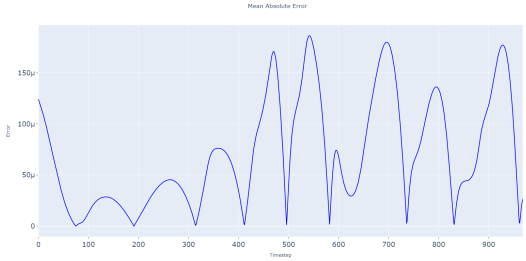
(c) Validation set comparison.



(d) Validation set prediction error.

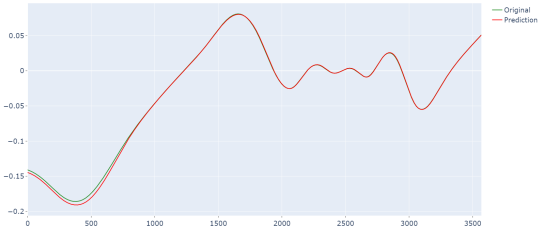


(e) Test set comparison.



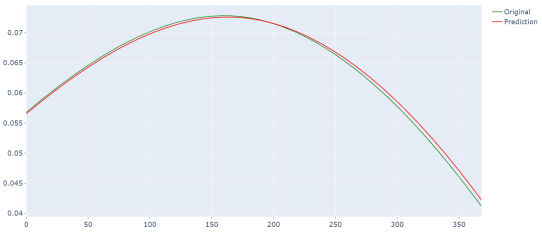
(f) Test set prediction error.

Figure A.8: EURCHF - IMF_7 real (green) vs predicted (red) and prediction errors.

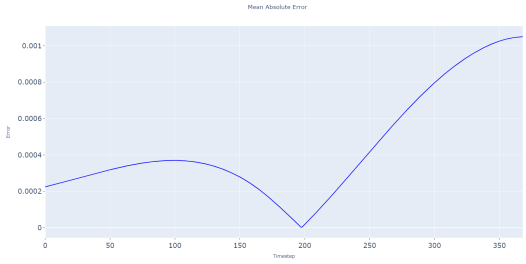


(a) Training set comparison.

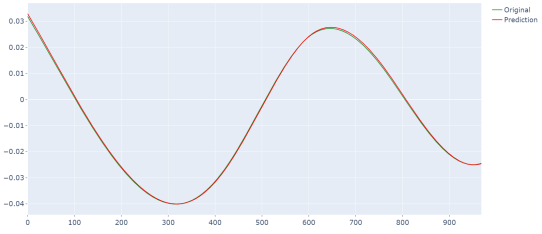
(b) Training set prediction error.



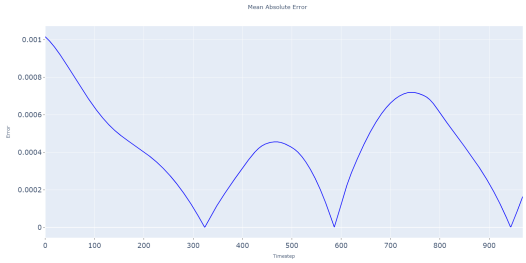
(c) Validation set comparison.



(d) Validation set prediction error.

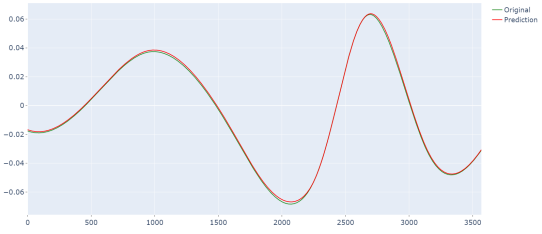


(e) Test set comparison.



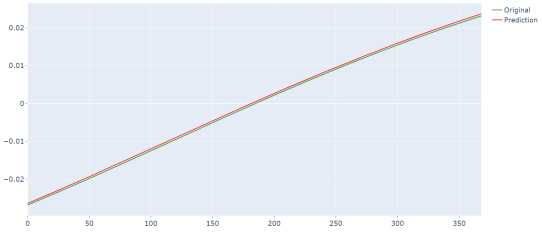
(f) Test set prediction error.

Figure A.9: EURCHF - IMF_8 real (green) vs predicted (red) and prediction errors.

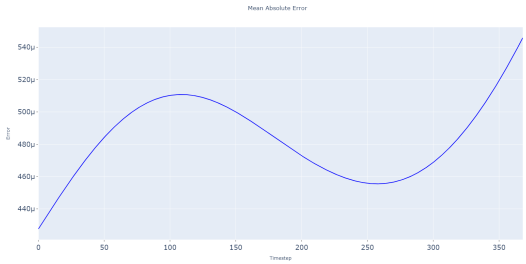


(a) Training set comparison.

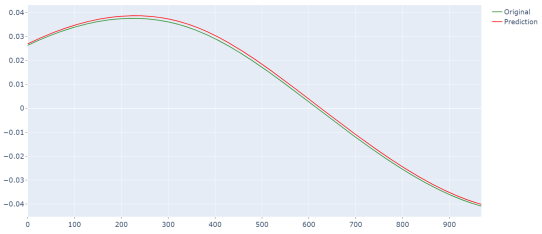
(b) Training set prediction error.



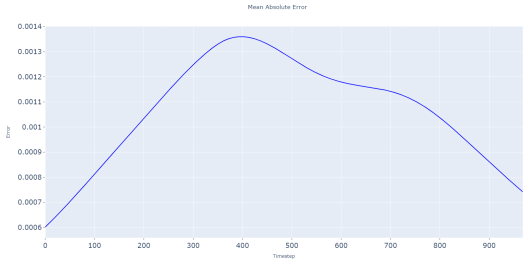
(c) Validation set comparison.



(d) Validation set prediction error.

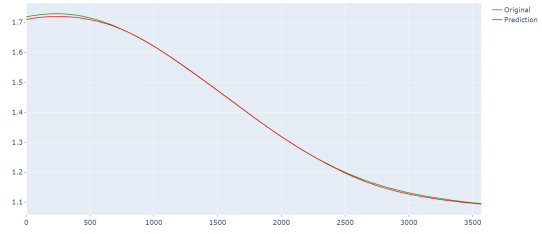


(e) Test set comparison.



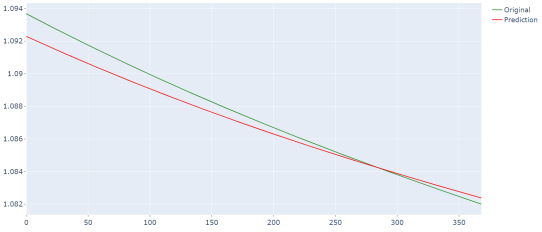
(f) Test set prediction error.

Figure A.10: EURCHF - IMF_9 real (green) vs predicted (red) and prediction errors.

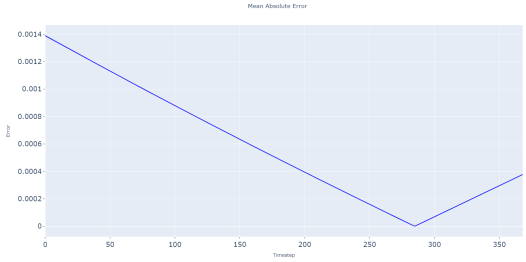


(a) Training set comparison.

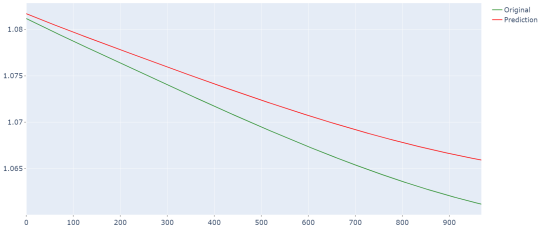
(b) Training set prediction error.



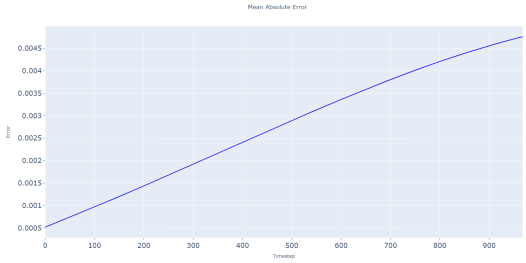
(c) Validation set comparison.



(d) Validation set prediction error.



(e) Test set comparison.



(f) Test set prediction error.

Figure A.11: EURCHF - IMF_{10} real (green) vs predicted (red) and prediction errors.

Table A.2: EUR/CHF, IMF results

(a) Train set prediction error.

IMF	MAE	MSE	MAPE
IMF_1	1.57e-03	7.49e-06	2258740078.2121
IMF_2	7.87e-04	2.01e-06	2258740078.2121
IMF_3	2.37e-04	2.13e-07	2258740078.2121
IMF_4	1.72e-04	1.26e-07	2258740078.2121
IMF_5	2.07e-04	1.13e-07	2258740078.2121
IMF_6	4.20e-04	4.45e-07	2258740078.2121
IMF_7	4.20e-04	3.80e-07	2258740078.2121
IMF_8	1.36e-03	5.36e-06	2258740078.2121
IMF_9	8.89e-04	1.01e-06	2258740078.2121
IMF_{10}	2.69e-03	1.46e-05	2258740078.2121

(b) Validation set prediction error.

IMF	MAE	MSE	MAPE
IMF_1	1.14e-03	3.50e-06	2258740078.2121
IMF_2	1.14e-03	4.96e-07	7933170076.9416
IMF_3	1.84e-04	7.82e-08	7933170076.9416
IMF_4	1.02e-04	2.05e-08	7933170076.9416
IMF_5	1.23e-04	2.40e-08	7933170076.9416
IMF_6	2.37e-04	7.10e-08	7933170076.9416
IMF_7	1.26e-04	2.30e-08	7933170076.9416
IMF_8	4.40e-04	2.79e-07	7933170076.9416
IMF_9	4.82e-04	2.33e-07	7933170076.9416
IMF_{10}	5.69e-04	4.93e-07	7933170076.9416

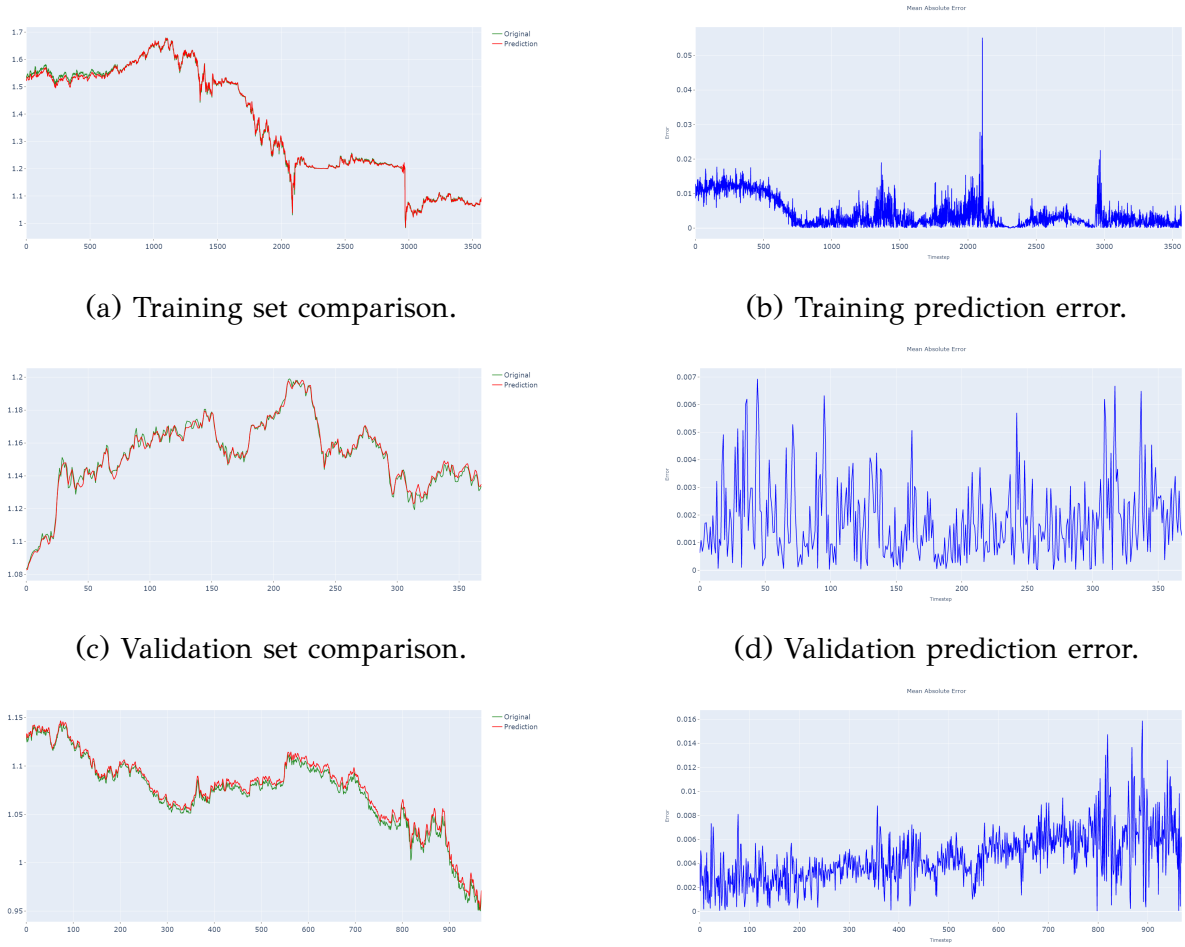
(c) Test set prediction error.

IMF	MAE	MSE	MAPE
IMF_1	1.15e-03	2.74e-06	7933170076.9416
IMF_2	4.69e-04	4.37e-07	7933170076.9416
IMF_3	8.77e-05	1.62e-08	7933170076.9416
IMF_4	1.02e-04	2.05e-08	7933170076.9416
IMF_5	1.31e-04	3.19e-08	7933170076.9416
IMF_6	2.91e-04	1.84e-07	7933170076.9416
IMF_7	7.07e-05	7.76e-09	7933170076.9416
IMF_8	4.18e-04	2.31e-07	2742129804.1685
IMF_9	1.07e-03	1.19e-06	2742129804.1685
IMF_{10}	2.76e-03	9.24e-06	2.58e-03

As it can been seen in the Tables A.2a - A.2c the MAPE metric, which was used to measure the models' performance, produces large numbers due to the near zero values that the IMFs have. As result, more attention is given in the rest of the metrics, MAE and MSE respectively.

Next, Fig. A.12 and Table A.3 show the EMD-LSTM results. In this case, the EMD-LSTM approach outperforms the standard LSTM approach, but still from the obtained results in the test set we can clearly see that there is still for improvement. Thus, the EMD-LSTM-PSO implementation was tested on the EUR/CHF currency too, in order to verify its superiority against the other implementations. Figure A.13a

and Table A.4 show that the proposed approach outperformed by far the standard LSTM and the EMD-LSTM approaches.



(a) Training set comparison.

(b) Training prediction error.

(c) Validation set comparison.

(d) Validation prediction error.

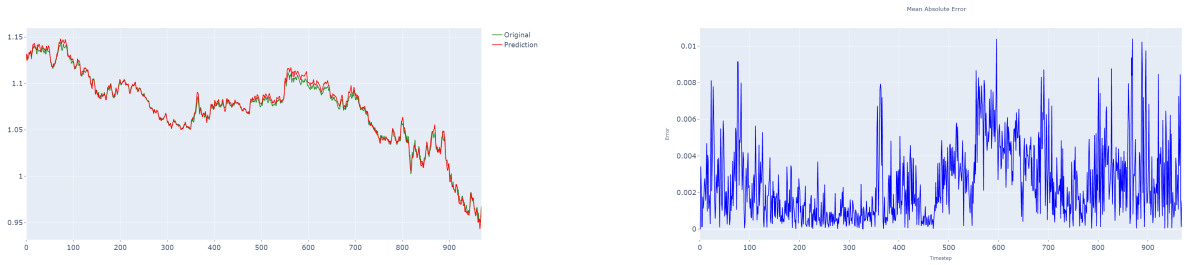
(e) Test set comparison.

(f) Test prediction error.

Figure A.12: EUR/CHF, EMD-LSTM comparison between real (green) and forecasted(red) prices and prediction errors.

Table A.3: EUR/CHF, EMD-LSTM prediction errors.

Set	MAE	MSE	MAPE
Train	4.23e-03	3.58-05	3.00e-03
Validation	1.78e-03	5.13e-06	1.55e-03
Test	4.50e-03	2.53e-05	4.24e-03



(a) Test set comparison.

(b) Test set prediction error.

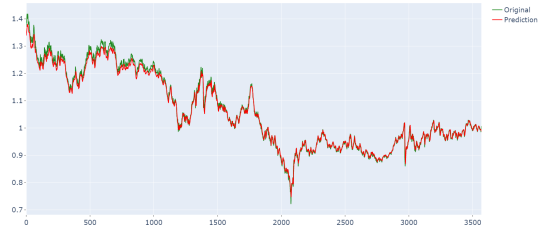
Figure A.13: EURCHF, EMD-LSTM-PSO real (green) vs predicted (red) and prediction errors.

Table A.4: EUR/CHF, EMD-LSTM-PSO prediction errors.

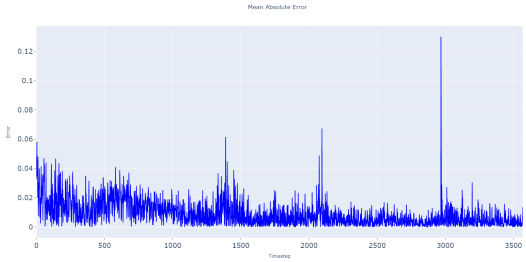
Set	MAE	MSE	MAPE
Test	**2.52e-03**	**1.06e-05**	**2.35e-03**

A.2.2 USD/CHF

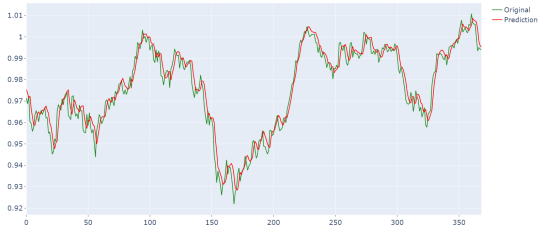
In this Section, the results for the USD/CHF currency pair are shown. As with the other two currency pairs, the proposed model behaves almost the same. Their main difference lies in the performance of the standard LSTM approach, which in this currency pair seems to perform well even in the testing period. However, that does not mean that performs better than the EMD-LSTM and the EMD-LSTM-PSO implementations. Comparing the EMD-LSTM results presented in Fig. A.24 and in Table A.7 with the respective results from the LSTM model shown in Fig. A.14 and A.5, the EMD-LSTM approach outperforms the standard LSTM approach. Moreover, the obtained results from the proposed approach shown in Fig. A.25a and in Table A.8 verify that its superiority against the other two implementations.



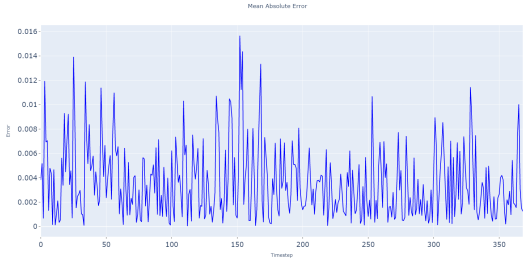
(a) Training set comparison.



(b) Training prediction error.

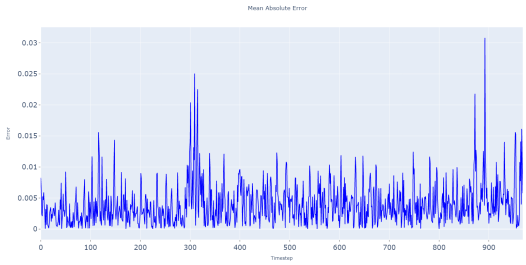


(c) Validation set comparison.



(d) Validation prediction error.

(e) Test set comparison.

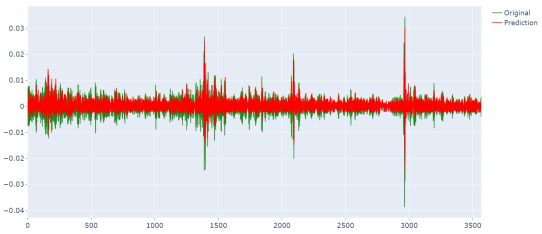


(f) Test prediction error.

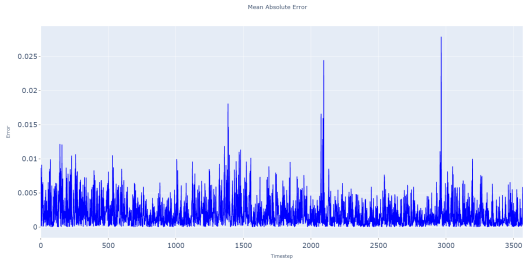
Figure A.14: LSTM, USD/CHF comparison between real (green) and forecasted(red) prices and prediction errors.

Table A.5: USD/CHF, LSTM prediction errors

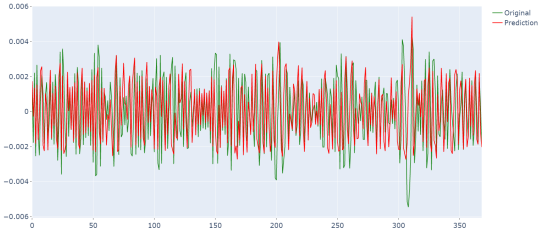
Set	MAE	MSE	MAPE
Train	8.93-03	1.50e-05	8.02e-03
Validation	3.57e-03	2.13e-05	3.68e-03
Test	3.97e-03	2.71e-05	4.20e-03



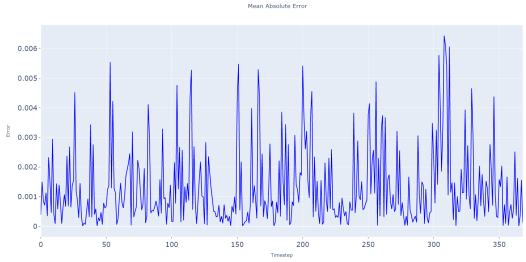
(a) Training set comparison.



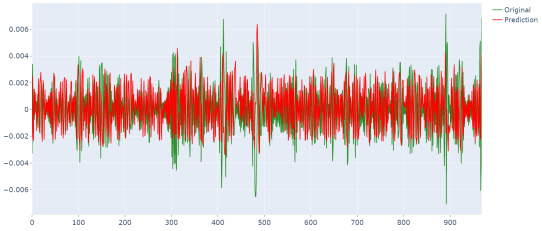
(b) Training set prediction error.



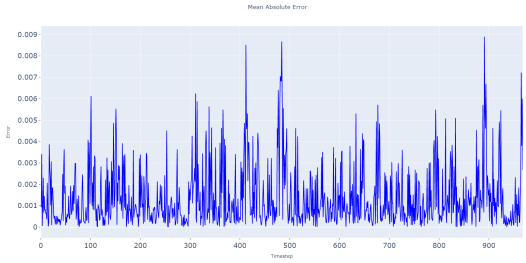
(c) Validation set comparison.



(d) Validation set prediction error.

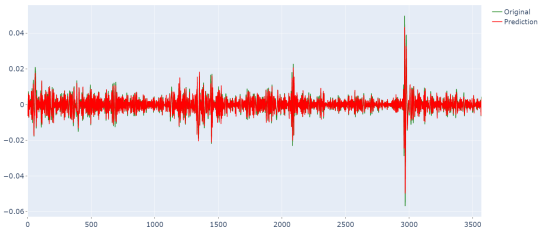


(e) Test set comparison.



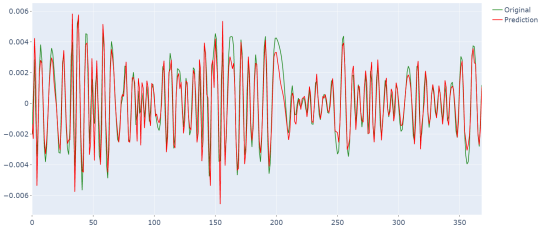
(f) Test set prediction error.

Figure A.15: USDCHF - IMF_1 real (green) vs predicted (red) and prediction errors.



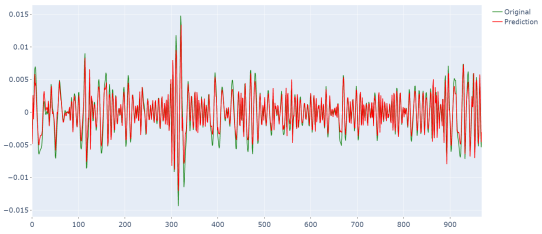
(a) Training set comparison.

(b) Training set prediction error.

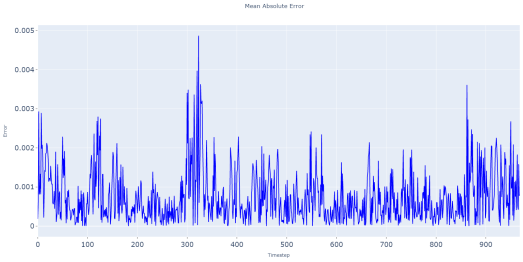


(c) Validation set comparison.

(d) Validation set prediction error.

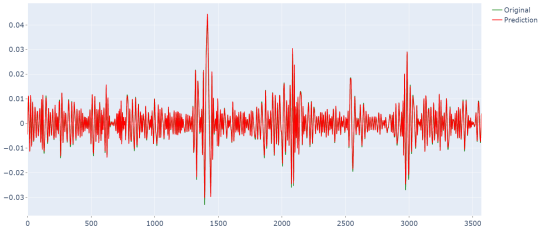


(e) Test set comparison.

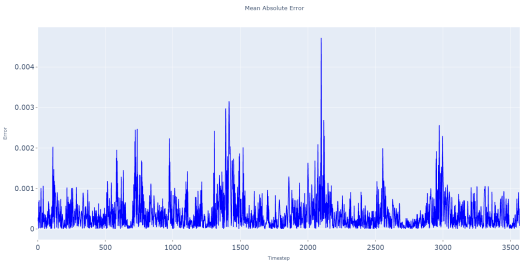


(f) Test set prediction error.

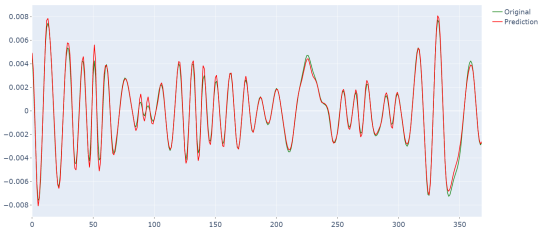
Figure A.16: USDCHF - IMF_2 real (green) vs predicted (red) and prediction errors.



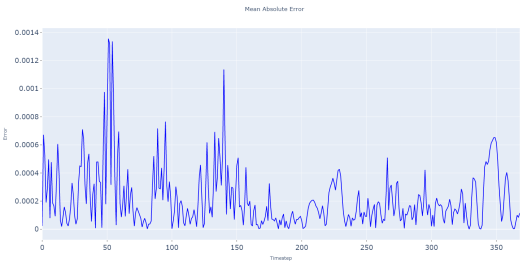
(a) Training set comparison.



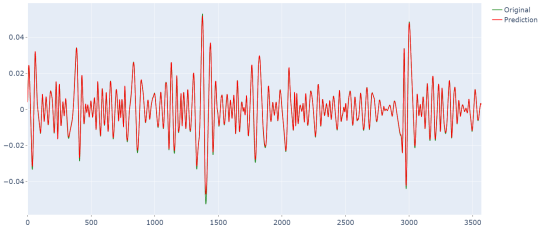
(b) Training set prediction error.



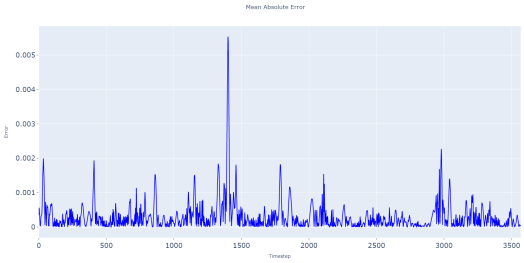
(c) Validation set comparison.



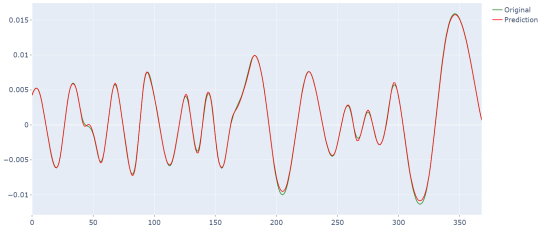
(d) Validation set prediction error.



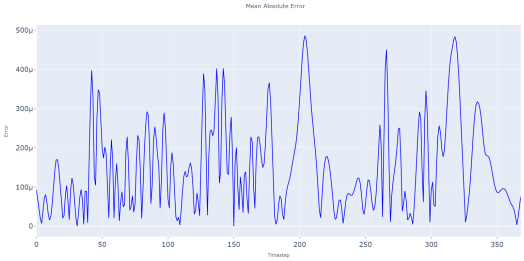
(a) Training set comparison.



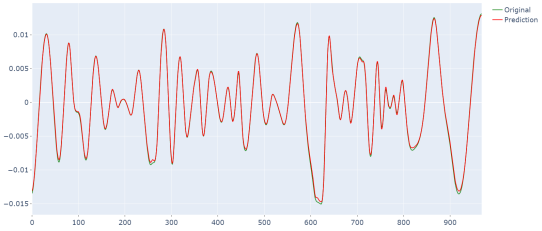
(b) Training set prediction error.



(c) Validation set comparison.



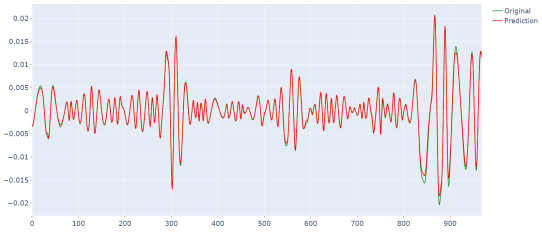
(d) Validation set prediction error.



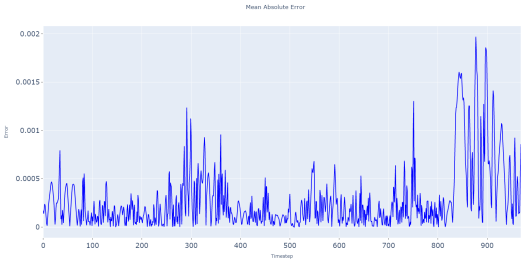
(e) Test set comparison.

(f) Test set prediction error.

Figure A.18: USDCHF - IMF_4 real (green) vs predicted (red) and prediction errors.

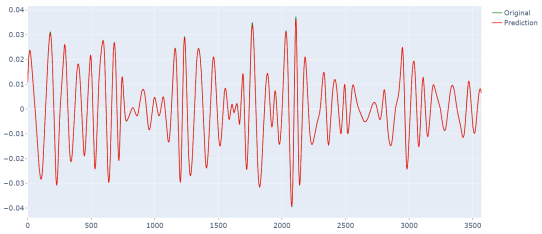


(e) Test set comparison.



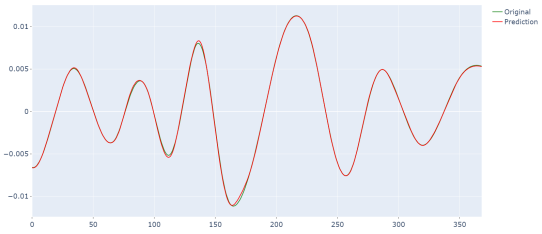
(f) Test set prediction error.

Figure A.17: USDCHF - IMF_3 real (green) vs predicted (red) and prediction errors.

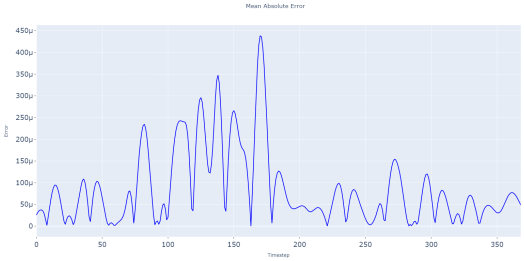


(a) Training set comparison.

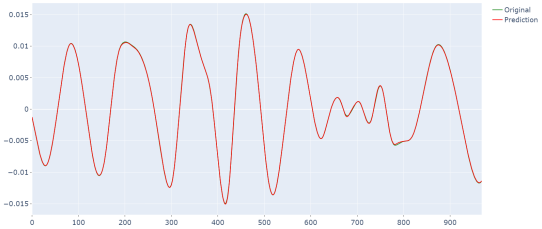
(b) Training set prediction error.



(c) Validation set comparison.



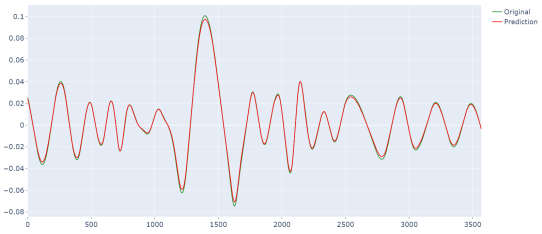
(d) Validation set prediction error.



(e) Test set comparison.

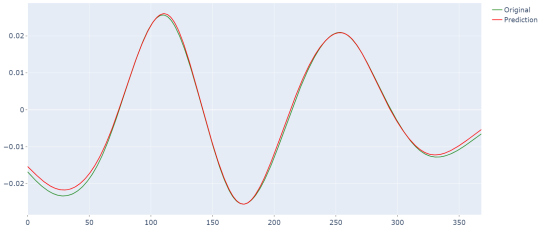
(f) Test set prediction error.

Figure A.19: USDCHF - IMF_5 real (green) vs predicted (red) and prediction errors.

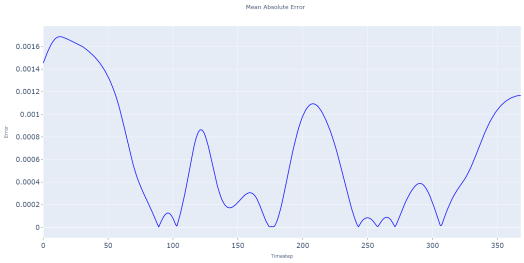


(a) Training set comparison.

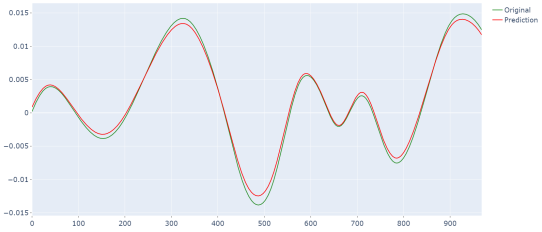
(b) Training set prediction error.



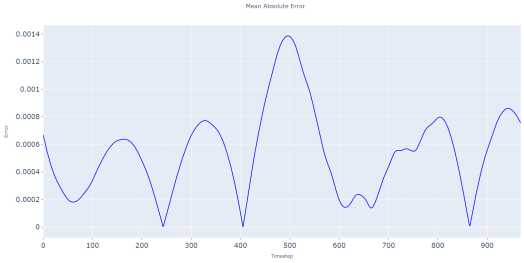
(c) Validation set comparison.



(d) Validation set prediction error.

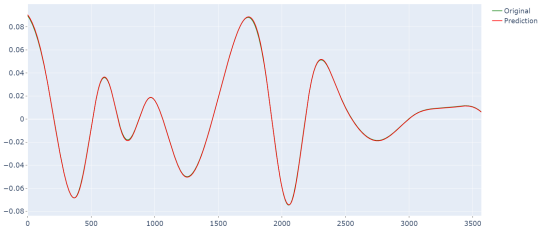


(e) Test set comparison.

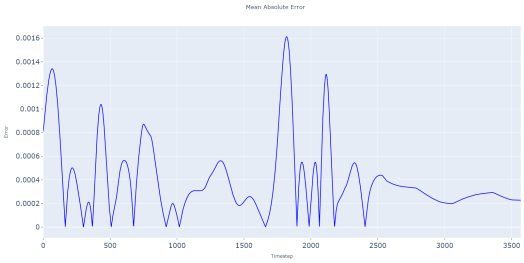


(f) Test set prediction error.

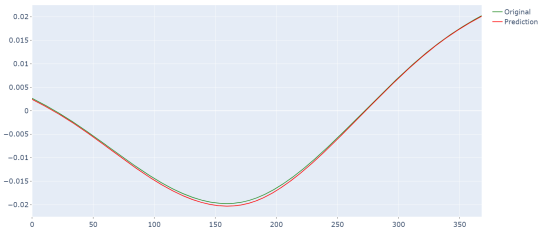
Figure A.20: USDCHF - IMF_6 real (green) vs predicted (red) and prediction errors.



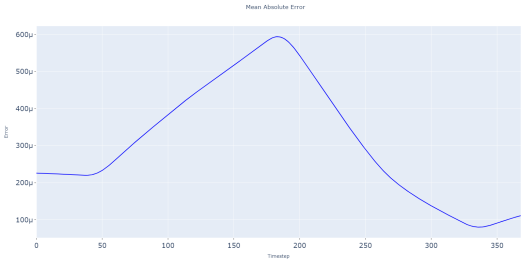
(a) Training set comparison.



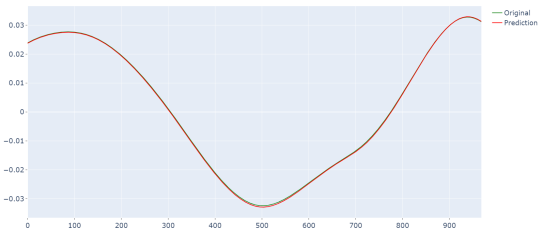
(b) Training set prediction error.



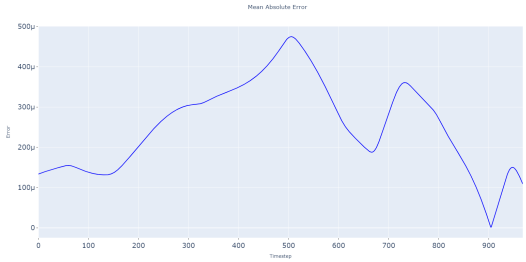
(c) Validation set comparison.



(d) Validation set prediction error.

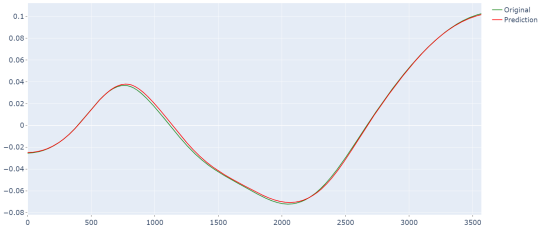


(e) Test set comparison.

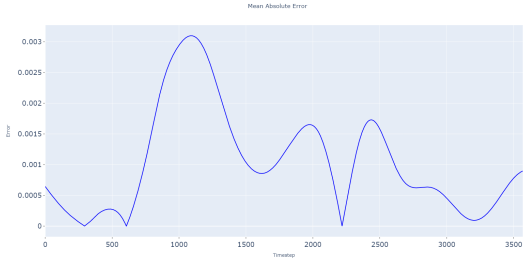


(f) Test set prediction error.

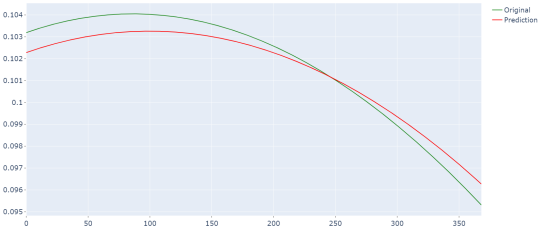
Figure A.21: USDCHF - IMF_7 real (green) vs predicted (red) and prediction errors.



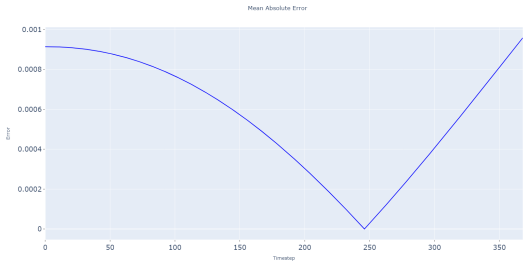
(a) Training set comparison.



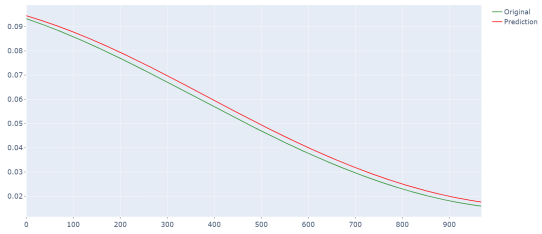
(b) Training set prediction error.



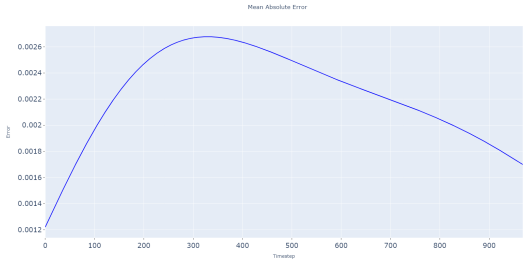
(c) Validation set comparison.



(d) Validation set prediction error.

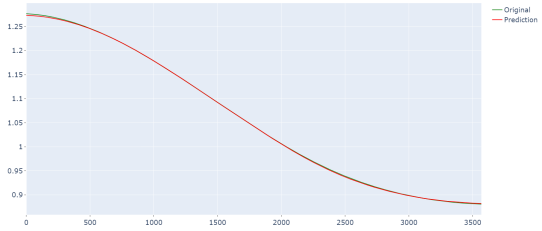


(e) Test set comparison.



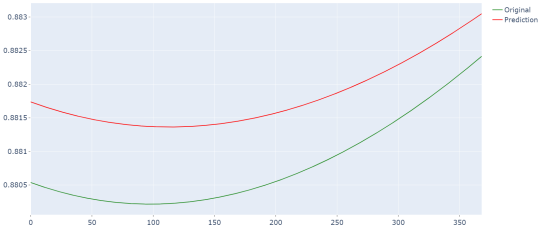
(f) Test set prediction error.

Figure A.22: USDCHF - IMF_8 real (green) vs predicted (red) and prediction errors.

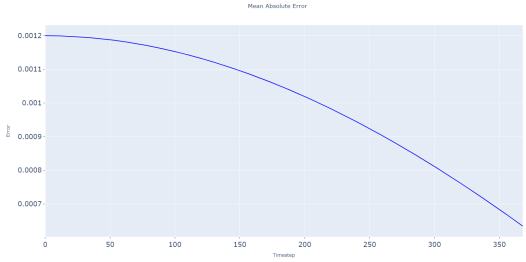


(a) Training set comparison.

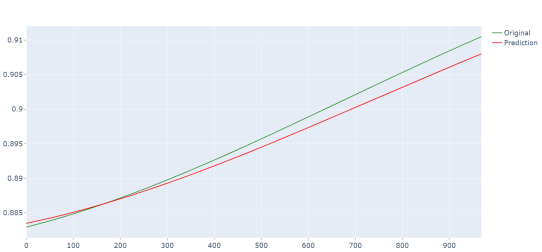
(b) Training set prediction error.



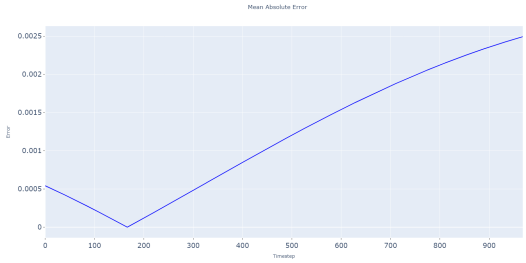
(c) Validation set comparison.



(d) Validation set prediction error.



(e) Test set comparison.



(f) Test set prediction error.

Figure A.23: USDCHF - IMF_9 real (green) vs predicted (red) and prediction errors.

Table A.6: USD/CHF, IMF results

(a) Train set prediction error.

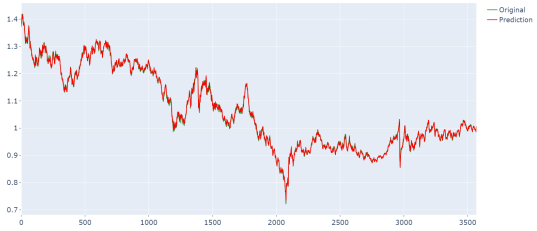
IMF	MAE	MSE	MAPE
IMF_1	1.98e-03	8.26e-06	2.58e-03
IMF_2	9.97e-04	2.04e-06	2.58e-03
IMF_3	3.76e-04	3.05e-07	2.58e-03
IMF_4	3.04e-04	2.78e-07	555221024.4754
IMF_5	1.27e-04	3.97e-08	555221024.4754
IMF_6	1.17e-03	2.24e-06	555221024.4754
IMF_7	3.99e-04	2.54e-07	4.81e-02
IMF_8	1.03e-03	1.77e-06	1.52e-01
IMF_9	7.86e-04	1.20e-06	1.52e-01

(b) Validation set prediction error.

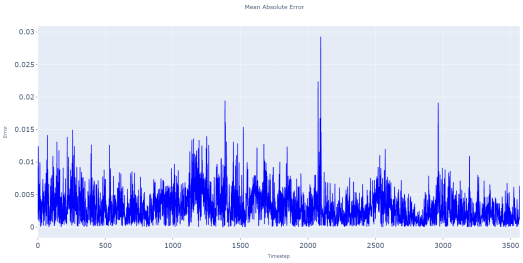
IMF	MAE	MSE	MAPE
IMF_1	1.31e-03	3.50e-06	1.52e-01
IMF_2	6.15e-04	7.13e-07	1.52e-01
IMF_3	2.13e-04	9.00e-08	9.00e-08
IMF_4	1.50e-04	3.56e-08	9.00e-08
IMF_5	8.60e-05	1.47e-08	9.00e-08
IMF_6	6.13e-04	6.46e-07	9.00e-08
IMF_7	3.12e-04	1.23e-07	9.00e-08
IMF_8	5.63e-04	3.98e-07	9.00e-08
IMF_9	1.00e-03	1.03e-06	9.00e-08

(c) Test set prediction error.

IMF	MAE	MSE	MAPE
IMF_1	1.37e-03	3.91e-06	4089956319.7622
IMF_2	7.39e-04	9.73e-07	4089956319.7622
IMF_3	2.76e-04	1.82e-07	1.91e-01
IMF_4	1.49e-04	3.82e-08	1.03e-01
IMF_5	5.56e-05	4.57e-09	3.30e-02
IMF_6	5.41e-04	3.90e-07	3.30e-02
IMF_7	2.53e-04	7.63e-08	4.55e-02
IMF_8	2.23e-03	5.11e-06	4.55e-02
IMF_9	1.18e-03	2.02e-06	1.31e-03



(a) Training set comparison.

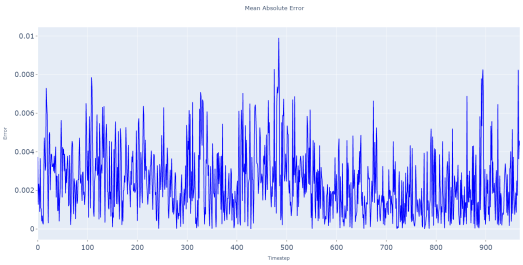


(b) Prediction error (train set).

(c) Validation set comparison.

(d) Prediction error (val set).

(e) Test set comparison.

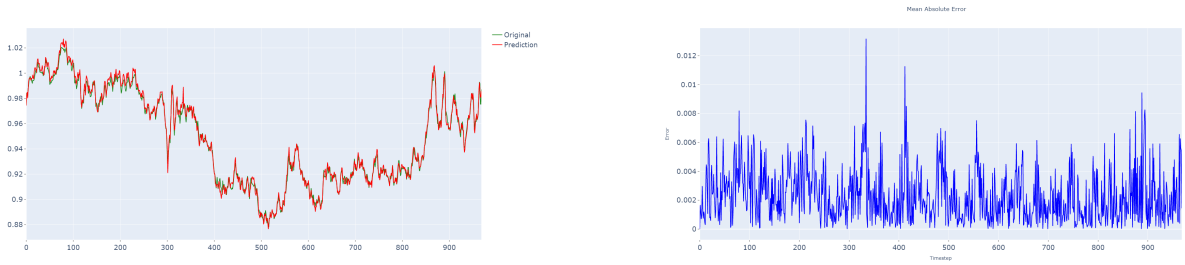


(f) Prediction error on (test set).

Figure A.24: USD/CHF, EMD-LSTM comparison between real (green) and forecasted(red) prices.

Table A.7: USD/CHF, EMD-LSTM prediction errors.

Set	MAE	MSE	MAPE
Train	3.08e-03	1.59e-05	2.91e-03
Validation	1.88e-03	5.66e-06	1.93e-03
Test	2.39e-03	8.69e-06	2.52e-03



(a) Test set comparison.

(b) Test set prediction error.

Figure A.25: USD/CHF, EMD-LSTM-PSO comparison between real (green) and forecasted(red) prices.

Table A.8: USD/CHF, EMD-LSTM-PSO prediction errors.

Set	MAE	MSE	MAPE
Test	**2.30e-03**	**8.71e-06**	**2.42e-03**

Short Biography

Michail Papatsimpas was born in Ioannina, Greece, in 1996. In 2014 he enrolled in the undergraduate program of Computer Science and Engineering of the University of Ioannina and earned his Diploma in 2020. His Diploma thesis was entitled "A FOREX Trading Model Based on Forecaster Aggregation and Metaheuristic Optimization", which was later presented in the *SETN 2020: 11th Hellenic Conference on Artificial Intelligence*. In 2020 he enrolled in the post-graduate program of the same Department. His research interests focus on Machine Learning and, more specifically, on Deep Learning methods.