KTH Computer Science
and Communication

Forecasting on-demand video viewership ratings using neural networks

JENS ARVIDSSON

Master's Thesis at CSC
Supervisor: Olov Engwall
Examiner: Olle Bälter

Abstract

Forecasting short-term viewership ratings for on-demand video is crucial for the online advertisement market because advertisement sales is done ahead of time, and errors in forecasting means either loss of profit opportunities or having to compensate advertisers for not upholding agreements. These forecasts can be made using an uncomplicated Seasonal Averaging method, which produces forecasts for the coming weeks using averaged hourly values from previous weeks (where the forecast for next Sunday is the average of the actual value from the last three Sundays). In this thesis, an alternative approach using a neural network is implemented and benchmarked against the Seasonal Averaging method, using data from December–February from a major online video site. The network utilizes a Multilayer Perceptron design with inputs corresponding to the seasonal patterns of the ratings data. It finds that while good forecasting performance can be reached even over very long horizons, weekly averages wins out when comparing standard forecasting error metrics, likely owing to the strong seasonal pattern.

Referat

Att förutsäga tittarsiffror för on-demand video

Att förutsäga tittarsiffror för strömmande video är viktigt för reklamindustrin då försäljning av reklam sker innan den visats. Alltför stora fel i dessa förutsägelser leder till att annonsörer måste kompenseras för ej visade reklamsnuttar, alternativt att möjligheter till att sälja mer reklam går förlorade. Dessa förutsägelser kan göras genom att ta genomsnittet av tidigare veckors tittarsiffror och använda detta som förutsägelse för påföljande veckor (där tittarsiffran för söndag nästa vecka är lika med genomsnittet av de senaste tre söndagarna). I det här exjobbet undersöks möjligheten att använda ett neuronnätverk för att göra dessa förutsägelser istället, genom att jämföra resultaten från detta mot den nuvarande metoden på data från December till Februari. Neuronnätet är av typen Multilayer Perceptron och använder en design som är anpassat till de veckovisa mönster som data uppvisar. Undersökningen finner att trots goda förutsägelser från neuronnätverket når det inte samma träffsäkerhet (mätt med standardmått på förutsägelser) som den nu använda metoden, troligtvis på grund av det starka veckovisa mönstret som data uppvisar.

Preface

This thesis was written at the KTH School of Computer Science and Communication (CSC), on request from Videoplaza Ltd. The author would like to thank his thesis counselor at CSC Olov Engwall, his mentors at Videoplaza Henry Rodrick and Joachim Hedenius as well as all Plazaits who beat him up in foosball, and lastly his thesis counseling group, consisting of Olle Hassel, Anton Lindström, Markus Felldin and David Nilsson. Tack!

Contents

Chapter 1

Introduction

Forecasting on-demand video ratings is still a relevant challenge for advertisers, because the business model is built around traditional television advertising where an amount of impressions is sold in bulk over a limited amount of time. On-demand video does not have the same temporal restrictions as traditional television (a show only running at 18:00 on wednesdays, for example), but the way advertising campaigns are sold has not changed much since the advent of the on-demand market.

On-demand video ratings poses a different set of challenges than those of traditional television, because of the unscheduled nature of television programming. While measuring the actual ratings is much more precise since every view (known as an impression) is logged, forecasting those ratings can be tricky - even if broadcasters have mastered forecasting ratings for a certain show, one cannot know at what time or weekday those impressions will be. However, since impressions are spread out over time, one can imagine using time series forecasting techniques to forecast the total amount of impressions each hour for any given online television channel.

Forecasting on time series is a well studied problem of statistics, one which has always had strong relevance for a range of industries. Forecasting has historically been (and still is) carried out using statistical methods such as regression analysis. Beginning with the 1990's, the body of research investigating the performance of machine learning methods has grown considerably, predominantly focusing on neural networks. There are conflicting finds on both sides of the forecasting fence, with no real consensus or best practices emerging for all time series problems.

This thesis explores the possibility of forecasting on-demand video viewership using a Multilayer Perceptron, which has been shown to exhibit good performance in forecasting all kinds of time series, including highly seasonal time series [1, 2].

1.1 Research question and scope

The question this thesis aims to explore is: can current forecasting methods for on-demand video viewership ratings be improved using neural networks? And, as a natural followup question, is the current method of Seasonal Averaging used by Videoplaza a sound approach?

The motivation for this question comes from the online video and advertising industry, for whom accurate forecasts are in high demand. This thesis aims to specifically explore hourly forecasts, since hourly targeting is a current requirement for some publishers. For instance, in order to reach a certain audience, a marketer might want to show advertisements from a certain campaign only between midnight and 3 am. This requirement depends greatly on the way that individual publishers conduct advertisement sales, but since Videoplaza currently offers hourly precision to its customers, hourly forecasts are the scope of this thesis.

1.2 Overview

This thesis begins with a background in Chapter 2 and an overview of previous work in the neural network forecasting field including implementation variants and comparisons to statistical methods in Chapter 3. Chapter 4 explains the characteristics of the data used in this thesis, and Chapter 5 explains the test setup used to judge forecasting methods. Chapter 6 contains a thorough explanation of the design and testing phase used to develop the neural network based on the literature, and is followed by benchmark tests on ratings data using widely accepted error measurements in Chapter 5 and finally an evaluation of the results in Chapter 7 - 8.

1.3 Word list

Time series

A time series is a series of data points measured at equally spaced time intervals.

Forecasting horizon

The number of future data points to be forecast is defined as the forecasting horizon. For example, if forecasting on a time series with monthly sales data, the forecasting horizon is the number of months to forecast into.

Ad Impression

An ad impression is a measure of how many times a piece of advertising is seen by a user of online media. In the context of online video, an impression is equivalent to an ad being shown to a viewer in conjunction with a video clip. Charging per impression is one of the main ways of monetizing online content, and different impressions are worth different amounts depending on if they were shown before a clip (preroll), in the middle of a clip (midroll) or after a clip (postroll) as well as which genre (or specific show) the clip belongs to.

Chapter 2

Forecasting

2.1 Statistical methods

Forecasting is the practice of predicting the future based on information from the past, and it relies on the basic assumption that there is a correlation between the past and the future. Research into quantitative methods of forecasting has increased considerably since the 1970's, when statistical modeling using ARIMA (Autoregressive Integrated Moving Average) methods was popularized by Box and Jenkins in 1970 [3]. Other models in use since then are Dampen Trend, Holt, Winter and a host of variants and combinations [4]. These models continued to hold prominence in the forecasting field well into the 1990's and are still used extensively today [5].

2.1.1 Naive methods

Naive methods for forecasting can be used as a baseline for evaluating the performance of forecasting models. The simplest one is straightforward – the forecast for the next period is equal to the actual value of the last period. Formally, we say that

$$F_{t+1} = Y_t \tag{2.1}$$

where F is the forecast at time period $t+1$, and Y_t is the actual observation at time period t. For any forecast horizon longer than 1 this method is obviously flawed, but the idea is to only use this method as a mean of benchmarking a real forecasting method. A slightly more advanced method involves removing the seasonality from the time series, and then doing the same as above, i.e. use the last observation as the forecast [5].

Seasonal Average

Another possibility for strongly seasonal data is using the average of the last n seasons as the future value for any given period. In this method, if the data contains a strong weekly cycle, the forecast for next Wednesday is the average of the last

n Wednesdays. Increasing the number of seasons used for the average creates a smoother forecast, but one that will be slow in picking up trends, while using fewer weeks will be susceptible to spikes and noise in the data.

2.2 Machine learning

2.2.1 Motivation

In recent times, great strides have been made in the forecasting field by using machine learning and data-driven approaches – a study by Zhang et al. reveals the uncertainty surrounding the performance of neural networks for forecasting, while more recent studies show an advantage in forecasting performance by machine learning approaches, and neural networks in particular [6, 1, 7]. Meanwhile, other recent studies show that statistical methods still hold their own in forecasting competitions, and the only real consensus in the literature seems to be that models need to be evaluated on a per-dataset basis [8]. This thesis therefore explores the possibilities of a neural network approach for forecasting on-demand online video ratings.

2.2.2 Artificial neural networks

An artificial neural network (ANN) is a nonlinear mapping system built up by small processing units, called neurons. The neurons are linked to each other via weighted connections, and receive input from other neurons through these connections which is processed by the neurons activation function. The output of the activation function is then sent on to other neurons in the network to be used as input.

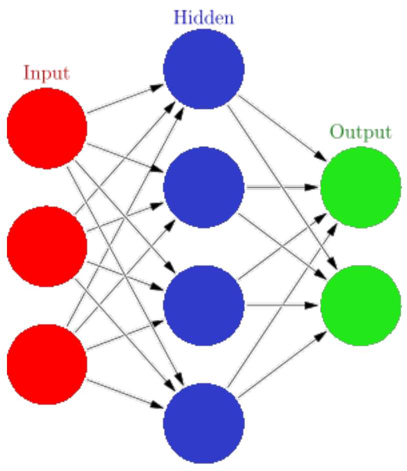

Figure 2.1: Basic multilayer perceptron layout

The input to a neural network is a vector of normalized numerical values in the range (1,0) which is fed to an input layer of neurons, that then pass it through to the hidden layers. These hidden layers ultimately pass their output on to the output layer, whose values is then interpreted as the output of the network [9].

In this way, a neural network functions as an approximator of a function from the input space to the desired output space. The weights of the connections between neurons are adjusted via supervised training, where the desired outputs given particular inputs are specified using one partition of the total amount of data known as the training set. The other partition of the data is the test set, which is used to evaluate the performance of the network. The most popular method of training a neural network is the backward propagation algorithm (also known as backpropagation or backprop, a network trained with this algorithm is denoted backpropagation network), and a detailed description of this algorithm can be found in multiple places [9, 10]. This approach requires the network designer to provide at least one parameter called the learning rate. This parameter balances training speed and avoidance of local error minimums. A similar algorithm called Resilient backpropagation needs no such tuning, since the same standard constants can be used for almost any neural networking problem. Resilient backpropagation has been shown to be a highly effective training algorithm in comparisons [9, 11].

2.2.3 Multilayer Perceptron

Motivation

The choice of a standard feed-forward Multilayer Perceptron as the neural network used in this thesis is based on the work of Ahmed et al. who found that they generally perform better on time series data than a range of different popular machine learning methods [1].

Description

A Multilayer Perceptron is constructed out of cascading layers of neurons. Every neuron in one layer is connected to all neurons in the next layer. It is feed-forward, that is, the connection only goes one way, from left to right as in Figure 1.

The input layer does no processing, but instead represents the input data, and thus it is the entry point of the system. Each input neuron passes on their value to every neuron in the first hidden layer through their weighted connections. The neurons in the hidden layer adds this weight input, runs it through its activation function, and passes on the output to every node in the next layer (which can be the output layer or another hidden layer).

Formally, what happens in each neuron in the hidden layer is as follows. The neuron receives input x_i from each neuron i in the preceding layer, associated with a weight w_i. The product of the input and the weight is summed:

$$p = \sum_{i=0}^{n} w_i x_i \tag{2.2}$$

and that sum is run through the neurons activation function to produce the output u:

$$u = f(p) \tag{2.3}$$

The sigmoid

$$f(u) = \frac{1}{1 + e^{-u}} \tag{2.4}$$

is a common choice of activation function because it limits very large or small values, and has a simple derivative which is important for performance when training the network. Other choices include tanh, step and radial basis functions, though some research indicates that the sigmoid and tanh functions are superior for most applications [12].

The output u is then sent forward in the network, to all neurons in the next layer, to be used in the same fashion. The output from the neurons in the output layer is finally interpreted as the output of the system.

The number of layers in the system can theoretically be any value, but Hornik showed that a MLP with as little as one hidden layer is a universal approximator, given enough neurons in that hidden layer [13]. A summary of proofs of this can be found in [9]. In practice, the number of hidden nodes required for an accurate result may be too high in a single hidden layer, and some research indicates that using more than one layer can improve accuracy [6].

Choosing the number of neurons in each layer is however still considered somewhat of an art and is also dependent on the data, but should generally lie somewhere between the number of inputs and the number of outputs. There exists a few rules of thumb for this, but ultimately a test-and-evaluate approach is necessary [14].

Time series forecasting

Forecasting on a time series with a neural network can be done in two ways, with a one-step ahead or multi-step ahead method. In both methods, the input vector consists of the normalized time series values for some fixed amount of time in the past leading up to the forecast horizon. In the one-step ahead method, there is only one output neuron, and it represents the forecast for the next time period. This forecast value is then used iteratively for forecasting the next period, until values for the entire forecasting horizon is found. In the multi-step ahead method, the output layer has one neuron for each time period in the forecasting horizon, whose values represent the total forecast. Which method performs best seems to be problem-specific, but later research seems to indicate better results for the iterative one-step ahead method [6, 15, 16].

2.3 Error Measurement

In order to compare and evaluate the efficiency of a forecasting network, we need a way to measure the accuracy of any given forecast. In the M3 competition (described in Chapter 3) the main error measurement used was the Mean Absolute Percentage Error (described in 2.3.1), which has remained popular among forecasters [4, 17]. However, Hyndman and Koehler argues for the adoption of Mean Absolute Scaled Error (described in 2.3.2) as the standard when comparing the performance of time series forecasts. They conclude that the MASE is the most stable and accurate error measure for time series forecasts, although the MAPE may be preferable under certain conditions because of its simplicity and intuitive explanation [18]. As such, in this thesis, both measures will be used to determine the accuracy of forecasts.

2.3.1 Mean Absolute Percentage Error

The Mean Absolute Percentage Error (MAPE) is the mean of the absolute Percentage Error (PE) of a forecast period. The PE for one forecast period t is calculated as follows:

$$PE_t = 100(Y_t - \frac{F_t}{Y_t}) \tag{2.5}$$

where, as above, F_t is the forecast at time period t, and Y_t is the actual observation at time period t. The MAPE is then defined as

$$\frac{1}{n}\sum_{t=1}^{n}|PE_t| \tag{2.6}$$

Where n is the forecast horizon. The MAPE is useful since it gives an error measure that is independent of the scale of the data, and allows the quality of forecasts on datasets with different time intervals and different sizes to be compared. It also gives an intuitive understanding of the forecast performance without comparisons – an error of 10% is easier to understand than an arbitrary error amount [5].

2.3.2 Mean Absolute Scaled Error

The Mean Absolute Scaled Error (MASE) is defined using the scaled error q_t, defined by

$$q_t = \frac{Y_t - F - t}{\frac{1}{n-1}\sum_{i=2}^{n}|Y_i - Y_i - 1|} \tag{2.7}$$

The MASE is then defined as

$$mean(|q_t|) \tag{2.8}$$

When the MASE for any method is less than 1, it is an indication that the method performs better than the naive one-step ahead method (and greater than 1 means it performs worse).The closer to 0 the MASE is, the better the method performs. The greatest advantage of MASE over simpler measurements like the mean absolute error is that it is independent of scale, which makes it better suited for comparisons across different datasets [18].

Chapter 3

Previous work

3.1 Comparisons of neural networks to statistical methods

Comparisons between statistical and neural network approaches to forecasting are not straightforward, and there is currently no consensus on which method is the "best", not even for specific datasets. The reasons outlined for this is variations and different properties of datasets involved, specifics of implementation and choice of error measurement [8]. Attempts to find the best forecasting method have been made, most prominently in the M3 competition held in 1999, where a multitude of statistical methods were benchmarked against each other on 3003 different time series datasets. Among these were only one neural network, which did not perform particularly well [4]. More recently in the NN3 competition which replicated the M3 competition circumstances with the addition of a much larger amount of neural network and other machine learning approaches, neural networks fared better but did not dominate the top list [8]. One of the most prominent results of both competitions was that methods which combined several other methods in some weighted average fashion generally fared much better than those that consisted of only one single model. The datasets included varied greatly, but forecasting horizons were limited to 18 data points, which makes the conclusions reached hard to apply to this thesis, where much longer horizons are required.

Other studies have found neural networks to forecast on average as well or better than common statistical methods, with Box-Jenkins ARIMA being a popular comparison method [19, 20]. The problem with these studies, however, is the uncertainty involved with general statements of fact regarding forecasting performance across all types of time series. Even when researchers produce good results forecasting with neural networks, it is hard to say whether they are better or worse than all statistical methods, since there are so many methods around. Generally, a highly specialized or improved technique is benchmarked against a basic, common technique because of the huge burden of testing every available one. A study done in 2012 exemplifies the problem of comparing forecasting methods: among forecasts on 28 different time series representing inflation in 28 different countries, which technique performed better varied between the time series. This is even though the time series in theory should have similar properties, since they model the same phenomenon (albeit in different countries) [21]. Thus, the results are inconclusive, despite neural networks performing quite well.

3.2 Neural network variants and implementation considerations

Forecasting with neural networks comprises a host of different implementations and designs. A comparison between multiple approaches done in 2010 using the M3 competition data concluded that a fairly standard MLP implementation outperformed all other tested implementations, including Bayesian neural networks, Radial Basis Function (RBF) networks, support vector regression, K-nearest neighbor regression and more [1]. Other studies finds that MLPs is the dominant type of neural network used in forecasting, and even though other designs show good performance, standard feed-forward MLPs on average perform as good or better [22].

Design issues for implementing MLPs include the number of nodes in each layer of the network, which most studies involving neural networks agree is something that need to be found through experimentation with the data at hand, since the optimal number of nodes to use for a particular task varies greatly with the problem to solve [6]. Another problem plaguing neural networks is overfitting, where the network is trained to be so good at predicting the training sample that it cannot generalize to new data. A study conducted 2010 proposes an alternative model to combat this, using a dual-network design where one network outputs a forecast and one a relative forecasting error, and their outputs are weighted to create an improved forecast which is less prone to error when the input changes over time [23]. Unfortunately, the work focuses on forecasts with a horizon of 1 data point, and using the method in a longer horizon context is complex.

3.3 Seasonal artificial neural networks

When dealing with strongly seasonal time series, one can exploit the repeating properties of the data in forecasts. One proposed model is the Seasonal artificial neural network (SANN), which uses the seasonality of the data in network design. The idea is to use the same amount of input nodes as there are data points in one seasonal cycle, meaning that for monthly data one uses 12 input nodes, for hourly data 24 nodes and so on. If the data is multi-seasonal, that is, more than one seasonal cycle is present, input nodes corresponding to the largest reasonable cycle is used. Applying this method to the dataset used in this thesis, with both daily and weekly cycles, one would use the amount of hours in a week as input nodes, e.g. 168 input nodes. The SANN approach makes no suggestions about the amount of hidden layers or nodes [24].

In contrast to traditional statistical methods, which emphasize removing the seasonal component of the data before forecasting, neural networks seem not to need this filtering, but rather detect the pattern if left in the data. A study carried out in 2012 showed an MLP outperforming traditional methods without any seasonal filtering, when that filtering was being done for the statistical methods tested. This result is highly relevant to this thesis, as it marks a real difference in approaches to seasonal data between statistical and neural network strategies, and the data used in this thesis is strongly seasonal.

3.4 Forecasting television ratings

Very little public research has been done in the area of on-demand video advertising, which can probably be attributed to the highly competitive nature of the market. An analysis of forecasting methods for traditional (scheduled) television ratings was done in 2011, concluding that research in the area is made difficult by the secrecy which surrounds the field. Although using ratings data from traditional television, the study indicates the same pattern found in on-demand video ratings: viewership is highly seasonal, with time of day, day of week and month of year playing a big part in determining viewership. This pattern is the motivation for using straight historical values from the same period the year before when forecasting television viewership, similar to the Seasonal Average in use for on-demand video forecasting [25].

Chapter 4

The data

The data used in this study encompasses a sample of nine weeks of viewer logs from three online video sites with different traffic volumes, from December 9th 2013 to February 18th 2014. The dataset is sampled at a rate of 1/100th of actual impressions.

The sites sampled were chosen because they represent the spectrum of online video sites – high, medium and low volume. The time period was chosen to capture irregularities found over holiday periods, so that tests could be run on both irregular and regular, stable data. This irregularity can be easily spotted in Figure 4.3, where the impression counts takes a big dive around Christmas and keeps exhibiting an irregular pattern which does not stabilize again until well into January. This period is interesting because it is prone to forecasting errors, stemming from the fact that the data before Christmas is regular and stable, forming an inaccurate basis for forecasts over the Christmas season.

The sample data exhibits strong seasonality on a daily and weekly basis. On a typical day, impression count progresses from around 6.00 every day, platforms around lunchtime and continues to climb until the peak at 21.00 after which the impressions rapidly decrease. Weekly, impressions peak on Wednesdays, and Sundays are marked by their typical "hump", where the daily peak is less pronounced as total impressions are spread more evenly throughout the day. A typical week can be seen in Figure 4.1 and a typical day in Figure 4.2. Figure 4.3 shows the full impression data from the largest dataset, where the disruption of the weekly pattern, as well as the dip in ratings produced by the holiday season, is clearly visible.

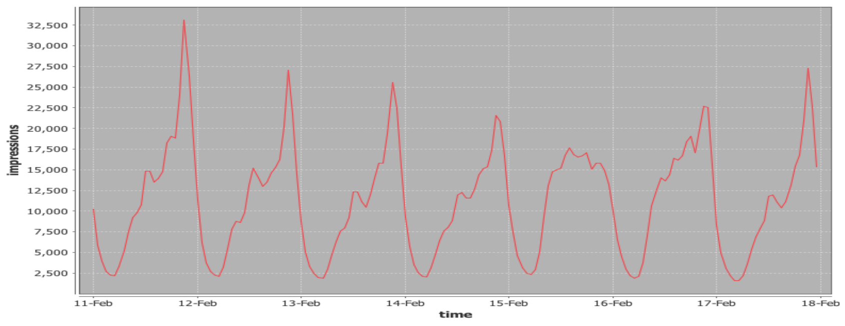

Figure 4.1: Typical week

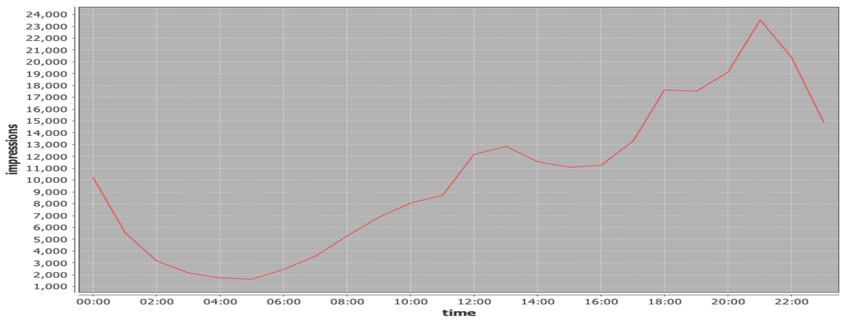

Figure 4.2: Typical day

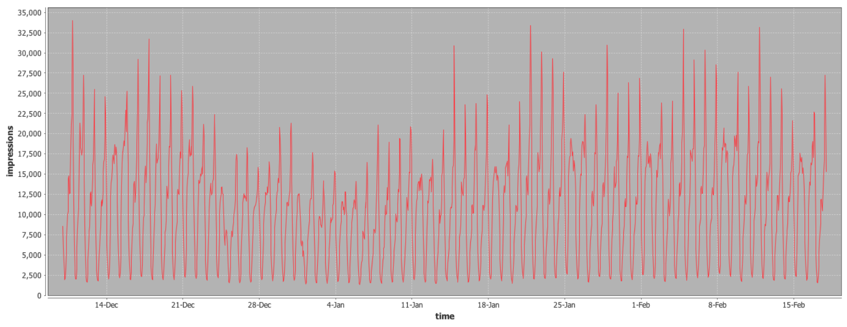

Figure 4.3: The full sample

Chapter 5

Test setup

In order to test whether a neural network can create forecasts with improved accuracy over current methods, a benchmarking framework was built so that tests could be run with varying amounts of data and over varying time periods. Reference implementations of the Seasonal Averaging method were created to represent the currently used method at Videoplaza, both to benchmark a neural network implementation against, and to investigate the sub-question of this thesis: is the current method of Seasonal Averaging used by Videoplaza a sound approach? The Seasonal Average methods were implemented using 3, 2 and 1 week of historical data, to discover whether using more or less data makes a significant difference to the results. Motivations and descriptions of how the neural network was designed and implemented can be found in Chapter 6.

The benchmark was designed as follows. Two different horizons were defined, which roughly corresponds to the normal usage seen by Videoplaza's user base: 1 and 3 weeks ahead (short and long term for ad campaigns). The Seasonal Averages used corresponding amounts of data for their forecasts, while the neural network used three weeks historical data for training, and an amount of hours leading up the forecasting period corresponding to the amount of input nodes as the first input. For example, when using 24 input nodes and forecasting from midnight the 30th of december, the 24 hours between midnight the 29th and midnight the 30th would be used as starting input. Comparison tests were then ran in a sliding window the size of training data + forecast horizon, sliding the window one day at each step, and recording the results. The purpose of this design was to gather statistical evidence of the performance of the tested methods over a long period, and to capture the behavior of methods when faced with both unpredictable (Christmas & New year celebrations) and predictable (late January and onward) data.

The results acquired from the testing program are in the form of result graphs created using the JFreeChart library showing MAPE errors at each step of the test.

Chapter 6

Neural network design

In order to evaluate and benchmark a neural network approach to forecasting, an implementation was created through experimentation outlined in the following chapter.

6.1 Experimentation parameters

The basis for the neural network used was a basic feed-forward MLP implementation, as discussed in the majority of forecasting literature on neural networks. The standard design parameters of a feed-forward MLP are: number of hidden layers, number of hidden nodes in each layer, training algorithm, and finally input and output vector sizes. In addition, the decision of whether to always train the network on data that lies just before the test period, or train it only once and use that network over all of the test periods had to be made.

The test setup described in Chapter 5 was used to evaluate network designs during experimentation. Literature ([6, 8, 22]) emphasizes the role of experimentation when finding an optimal network design for a certain dataset, and therefore final design decisions were decided based on results from these evaluations. The evaluation was straightforward – the lower MAPE and MASE sum over the test period, the better.

6.2 Hidden layers and number of hidden nodes

MLP designs commonly use one or at most two layers of hidden nodes, with more layers increasing the risk of overtraining. Running the evaluation on networks with one hidden layer yielded the lowest errors, regardless of the number of neurons in each layer.

How to choose the number of hidden neurons is less well understood, and most literature defer to experimentation. In order to find the optimal number, evaluation is necessary, and so the method of finding the optimal number of hidden nodes was conducted in the following way. A network with one hidden layer was constructed, using some basic assumptions for input and output nodes (later, at the end of the experimentation phase, the test was rerun using the final number of nodes to confirm the results). Starting at a small number of hidden nodes (10), a full evaluation was run and the MAPE sum recorded. The number of nodes was then increased by 1, the evaluation run again and the MAPE recorded. Continuing this evaluation until MAPE values begin to destabilize, the recorded MAPE values were then graphed.

The results of this method can be seen in Figure 6.1, where it is clear that the amount of hidden neurons, when chosen between 40 and 70 neurons, does not impact results to a significant degree. Exact MAPE values naturally varied between runs, but the pattern of results stabilizing after 40 and destabilizing after 70 was consistent. Since increasing the amount of neurons in the hidden layer also increases running time, it is preferable to choose the smallest amount which retains good performance, and thus, 40 neurons was chosen for the final design.

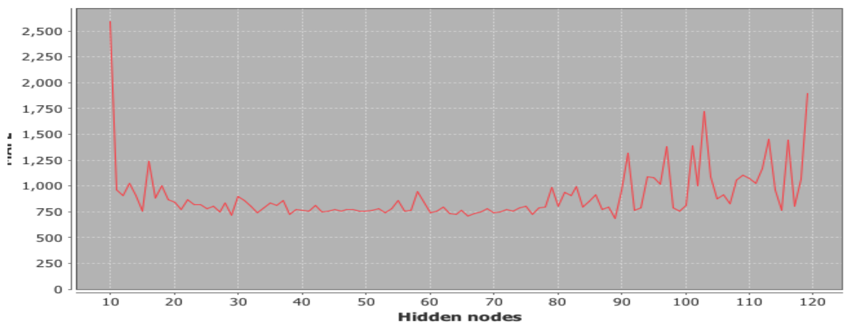

Figure 6.1: MAPE sum as a function of the number of hidden nodes in MLP

6.3 Output vector size

The size of the output vector depends on the approach taken regarding forecasting – multi-step or one-step ahead forecasting. For one-step ahead forecasting, only one output node is used, while multi-step ahead forecasting uses multiple output nodes, up to (at most) the whole forecast horizon. There is some evidence for good performance in both methods, and thus, both were evaluated using the benchmarking environment [6, 15, 16].

The tested variants were one-step ahead, 24-hour (24 neurons) and 1 week ahead (168 neurons) approaches. The number of neurons for the multi-step ahead variants were motivated by the SANN model proposed by Hamzaçebi [26], which produced improved results using input and output vectors corresponding to seasonal cycles in the data. The main seasonal cycles in the data sample were daily and weekly cycles, which motivates the choices of 24 hours and 1 week as output vectors to test.

The results of the 24-hour ahead method can be seen in Figure 6.2 . The 1-week ahead forecast is not included, since it the forecasts it produced were on average three times worse than the other two methods. Although the 24-hour ahead method appears to produce adequate forecasts for the most part, the one-step ahead is clearly superior, having a lower MAPE overall.

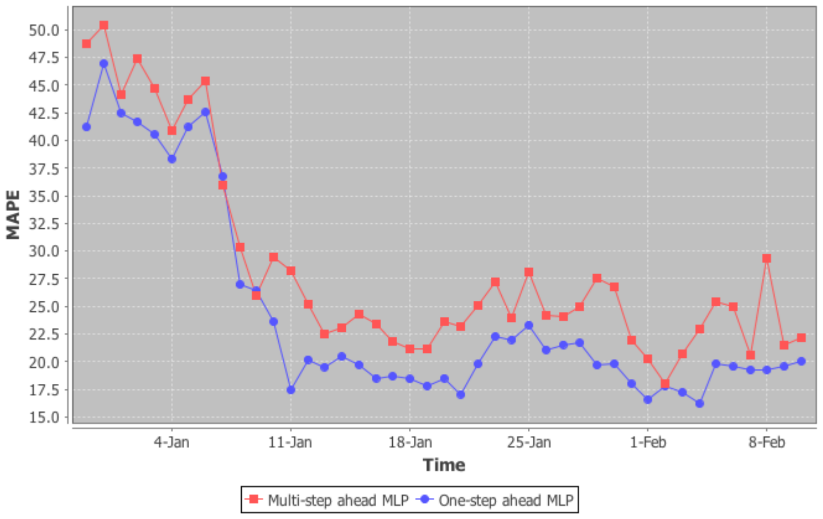

Figure 6.2: MAPE over evaluation period for 24-hour ahead and one-step ahead methods

6.4 Input vector size

The optimal size of the input vector can be determined experimentally in the same fashion as other parameters. The SANN model can, again be used as a starting point, proposing an input vector corresponding to the seasonal cycles in the data. This means an input vector of 24 or 168 neurons based on the daily and weekly seasonality in the data. An evaluation of networks with increasing amounts of input neurons (starting at 7) was carried out, the result of which can be seen in Figure 6.3. The results of this evaluation agrees with the theory of the SANN model: MAPE values stabilize around 24 input neurons, to later climb and descend again at exactly 168 neurons. Increasing the number of input neurons after 168 offers no benefit, and the error rises again if using more neurons past 230. Based on the SANN model and the evaluation carried out, 168 input neurons was chosen for the final design.

6.5 Training algorithm

Forecasting literature indicates that Backpropagation is the standard training algorithm for neural networks used for time series forecasting, while some literature on neural networks indicate that Resilient Backpropagation is faster and less prone

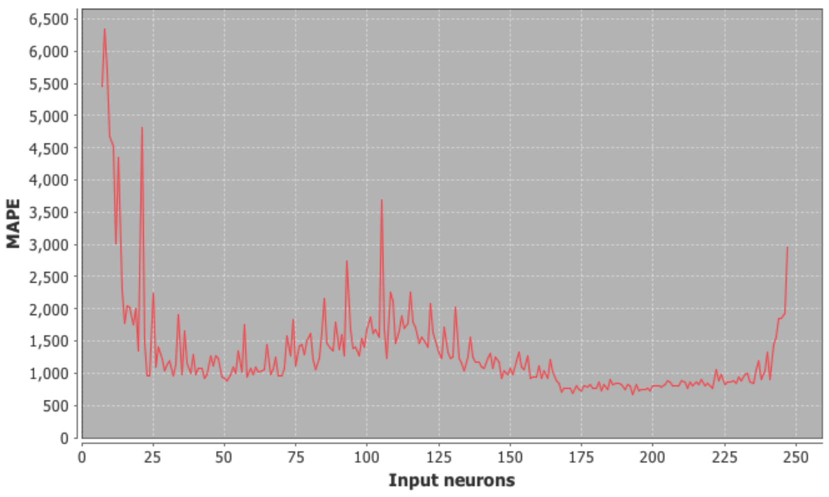

Figure 6.3: MAPE sum as a function of the number of input neurons in the network

to getting stuck in local error minimums for most problems [9, 11]. As such, both were evaluated during experimentation, which showed that Resilient Backpropagation indeed produced much better results in the form of lower MAPE values than regular Backpropagation. Resilient Backpropagation was therefore chosen as the training algorithm.

6.6 Dynamic retraining vs Static network

The question of whether to dynamically retrain the network before each forecast in the sliding window test also had to be answered, since the evaluation is done on multiple continuous time series. Finding answers to this question in literature proved difficult, since time series forecasting research generally focus on forecasting one dataset at a time - which could be interpreted as an indication that retraining the network dynamically is the correct approach. Intuitively, one might think that a continuously retrained network will perform better than one trained on only one set of data since it has access to the latest data, but the evaluation of the two approaches proved differently. The network trained once on data between 9th och December - 30th of December produced much more stable and more accurate forecasts than the method of updating the network at each step. A comparison between the two approaches can be seen in Figure 6.4, where very large error rates can be seen in the beginning of January. This is likely because of the increasingly irregular patterns being found in the training data as the evaluation window moves forward through January. Although there are irregularities in the training data for the static approach, it seems like this is offset by the presence of enough stable and regular data from the beginning of December.

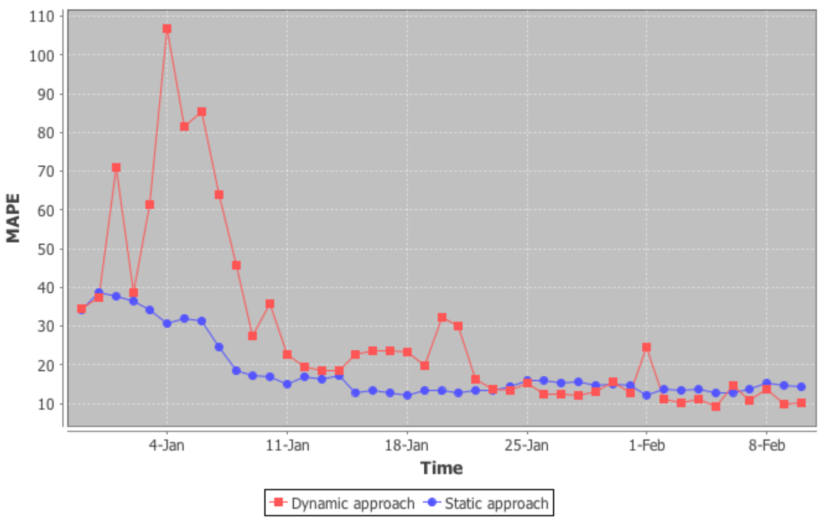

Figure 6.4: MAPE over evaluation period of a dynamically retrained network versus a network trained once

6.7 Final MLP implementation

The final implementation of the MLP used for benchmarking against the Seasonal Average method, motivated by experiments based on literature, used the following design: 168 input neurons, 1 hidden layer with 40 hidden neurons, and 1 output neuron. The training algorithm used was Resilient Backpropagation, and the network was trained only once, on the first three weeks of data in the sample.

The network was implemented in the Java programming language using the well-tested Encog neural network library [14]. The benchmarking environment was also implemented in Java, using the JFreeChart library to create graphs.

Chapter 7

Benchmark results

The results below are generated from one of the video sites in the dataset – the largest one. Very similar results were found while benchmarking using data from the other two video sites in the complete dataset, and therefore those results are omitted.

Running the sliding window benchmark test starting at the 30th of December 2013, Figure 7.1 and 7.2 shows the MAPE of the MLP compared to the MAPE of the 3 different Seasonal Averages at each sliding iteration (for 1 and 3 week horizons, respectively). The value at the 6th of January indicates the forecast error from forecasting 6th Jan - 13th Jan for the one week horizon tests, and 6th Jan - 27th Jan for the three week horizon tests.

The MAPE and MASE sums for all four methods are tabulated in Table 7.1 for one-week horizons and Table 7.2 for three-week horizons.

The huge difference in error from the beginning of the evaluation period (January) compared to the end (mid February) stems from the huge irregularity in the data caused by the holiday season, which brings with it a big decrease in viewership as well as an irregular pattern. Typical weekend patterns are placed in the middle of the week, and ratings volume is nearly cut in half.

Method	MAPE	MASE
Last week average	641	35.9
Last 2 weeks average	735	40.7
Last 3 weeks average	806	45.0
MLP	742	38.9

Table 7.1: Summed error comparison of Seasonal Averages and MLP over 1 week horizon

Method	MAPE	MASE
Last week average	574	32.4
Last 2 weeks average	622	34.9
Last 3 weeks average	612	35.0
MLP	767	42.5

Table 7.2: Summed error comparison of Seasonal Averages and MLP over 3 week horizon

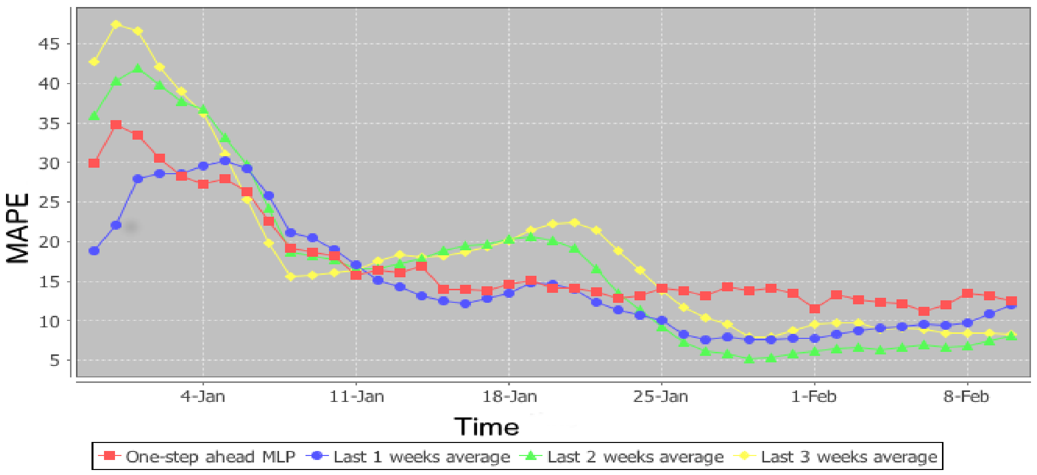

Figure 7.1: MAPE errors of MLP in benchmark period compared to weekly averages over 1 week horizons

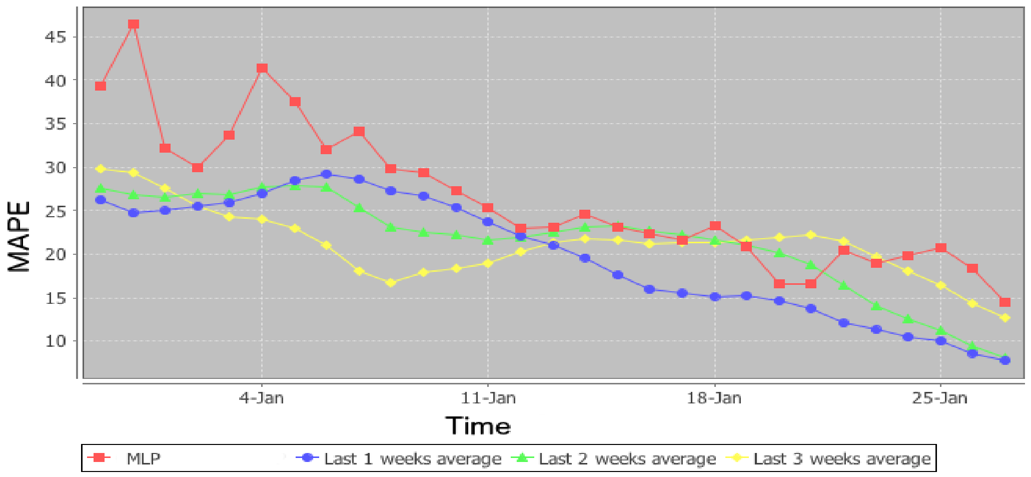

Figure 7.2: MAPE errors of MLP in benchmark period compared to weekly averages over 3 week horizons

7.1 Typical forecast results

A typical successful run with the MLP captured the repeating pattern with some success, but was usually unable to model large spikes or trends. Figure 7.3 shows a typical well–behaved forecast, where the network picks up on the weekly repeating pattern including the Sunday hump.

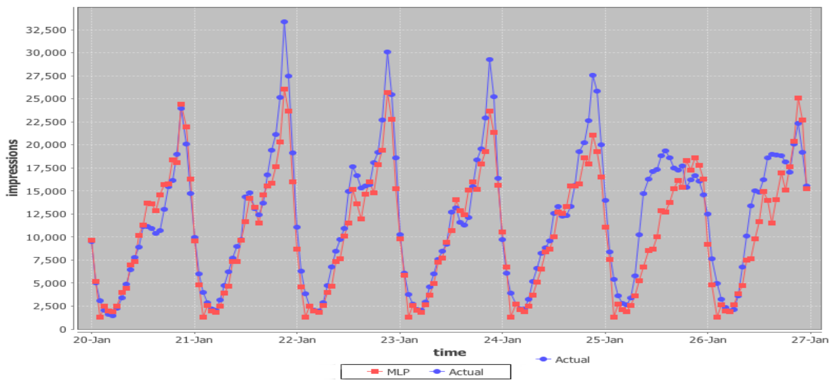

Figure 7.3: A typical successful forecast created by a MLP

Failed runs (runs with high error rates) could usually be identified by their propagation of initial forecasting errors or heavy reliance on a repeating daily pattern. Figure 7.4 shows a typical out of sync forecast run where the network has identified a repeating pattern, but failed to identify the daily variations, and over time completely degenerates as it goes out of sync. This behavior leads to big errors on a hourly level because of hourly de-synchronization, but can still produce accurate forecasts on a daily level.

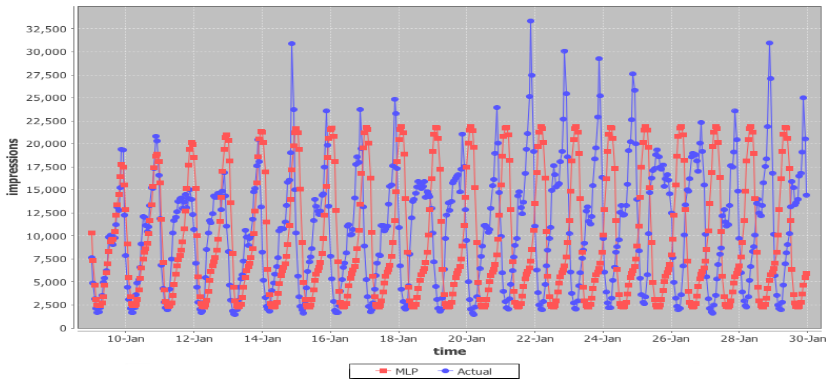

Figure 7.4: A typical degenerating forecast created by an MLP

7.2 Evaluation

The MLP did not manage to outperform the Seasonal Averages over the entire period, despite performing better than some of them intermittently. The main factor in this is the fact that the Seasonal Averages begin to outperform the MLP consistently at the end of the benchmark window, where the sample data begins to take on a predictable and unchanging pattern. It is this pattern that allows the Seasonal Averages to perform so well, and the MLP designs are not able to keep up when the Seasonal Averages are performing at their best. According to Videoplaza, this repeating pattern makes up the majority of the data throughout the year, with brief interruptions for holidays and other major events. The repeating pattern is so strong and so predictable that the Seasonal Average method really works well, which is one explanation as to why the MLP approach cannot keep up.

Among the different weekly averages the result indicate a slight advantage for the 1 week method in all error measurements.

Chapter 8

Analysis of results

The results seem to indicate a small advantage to the weekly average methods over the MLP, and in particular the 1 week Seasonal Average, which is best situated to adapt to changes in viewership patterns. When the data falls into a repeating pattern, the differences between the different weekly method diminishes, and while the MLP also sees a stabilization over that period, it never quite catches up to the averaging method. Considering a majority of a year's data resembles that repeating period, it can be argued that MLP methods does not produce superior forecasts on online video viewership statistics.

The MLP evaluated in this thesis was implemented according to previous work in forecasting literature, as well as thoroughly experimented with in regards to parameters to produce an optimal outcome – there is therefore little reason to believe that design flaws are the cause of the MLP performing worse than the Seasonal Average method. Two factors likely play the major roles: first of all, even the 1 week forecast horizon is abnormally long. Encompassing 168 data points, the horizon is considerably longer than forecast horizons in most forecasting literature. This creates problems for any neural network approach since forecasting errors are propagated over the entire horizon, which means small error at the outset affects the whole forecast, as is demonstrated in Figure 7.4. Second, the data is strongly seasonal, a property so well modeled by the weekly averaging method that it becomes hard for the MLP to beat it. If the data repeats itself exactly every week, the Seasonal Average method will perform at 100% accuracy. The area in which the MLP approach could shine is in anticipating spikes and lows in the data, which it could not do much better than the Seasonal Averages. It could be argued that the MLP could anticipate changes in the data better with more data, perhaps with a few year's worth of data the yearly seasonal cycle could be picked up by the MLP. However, the same could be said for the Seasonal Averaging method, given data from some years back.

8.1 Daily aggregation

There is one point to make about MLP performance which is out of scope of this thesis, but nevertheless interesting: forecasting results aggregated on a daily basis. Aggregating the hourly forecasts into daily forecasts beginning at midnight each day and running the same evaluation described in Chapter 5 but instead of measuring MAPE measuring the Mean Absolute Error (MAE):

$$\frac{1}{n}\sum_{t=0}^{n}|e_t| \tag{8.1}$$

where e_t is the error at time t, a different picture emerges: the MLP actually outperforms all of the Seasonal Averages. This can be seen in Figure 8.1. The reason for using the MAE for the daily aggregates here is that it is a measure of the absolute amount of "missed impressions" rather than a measure of forecast graph accuracy – in short, it is a more relatable measure for publishers and advertisers, since it represents the amount of impressions they're actually losing.

What does this mean? It is an indication that the difference in performance between the MLP and the Seasonal Averages comes down more to small hourly sync errors rather than major erroneous calculation. This means that for publishers and advertisers more interested in forecast resolutions on a daily level, the MLP approach may be worth a second look. However, it should be noted that forecasting on an hourly level and aggregating the results to a daily level might not be the best approach if daily forecasts are sought – it is likely that forecasting directly on the daily data will create even better results.

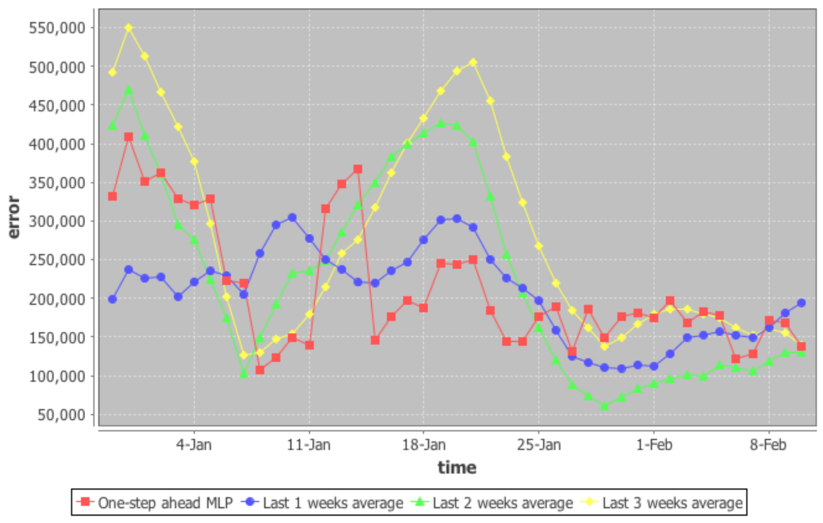

Figure 8.1: Daily MAE of MLP in benchmark period compared to weekly averages over 1 week horizons

8.2 Implications

The implications for companies like Videoplaza are clear: the currently used method of weekly averaging is a solid and reasonable approach. It would be possible to ease up on sample size, since using one or two weeks is not only as good, but in most cases gives a more accurate forecast than using three weeks as averaging data. Neural networks, of the form tested in this thesis, are both more complex to build and maintain and are less accurate than weekly averaging, and based on these tests cannot be recommended for implementation.

8.3 Possible improvements

In order to improve performance of a neural network, one could analyze each forecast produced by a neural network, and compare the error rate and look of the graph to detect strange behavior, thereby eliminating the outliers that appear in some of the tests. One can also imagine producing a daily forecast in addition to the hourly, and reconciling the two forecasts. With this approach, an advertiser who is really only interested in the daily forecast can get more accurate results, while still maintaining an improved hourly forecast.

Bibliography

[1] N. K. Ahmed, A. Atiya, N. E. Gayar, and H. El-Shishiny, “An empirical comparison of machine learning models for time series forecasting,” *Econometric Reviews*, no. 29, pp. 594–621, 2010.

[2] R. Adhikari and R. K. Agrawa, “Forecasting strong seasonal time series with artificial neural networks,” *Journal of Scientific & Industrial Research*, vol. 71, pp. 657–666, 2012.

[3] R. S. Tsay, “Time series and forecasting: Brief history and future research,” *Journal of the American Statistical Association*, vol. 95, no. 450, pp. 638–643, 2000.

[4] S. Makridakis and M. Hibon, “The m3-competition: results, conclusions and implications,” *International Journal of Forecasting*, vol. 16, pp. 451–476, 2000.

[5] S. Makridakis, S. C. Wheelwright, and R. J. Hyndman, *Forecasting: methods and applications.* Wiley, 1998.

[6] G. Zhang, E. B. Patuwo, and M. Y. Hu, “Forecasting with artificial neural networks: The state of the art,” *International Journal of Forecasting*, vol. 14, no. 29, pp. 35–62, 1998.

[7] C. A. Mitrea, C. K. M. Lee, and Z. Wu, “A comparison between neural networks and traditional forecasting methods: A case study,” *International Journal of Engineering Business Management*, vol. 1, no. 2, pp. 19—24, 2009.

[8] S. F. Crone, M. Hibon, and K. Nikolopoulos, “Advances in forecasting with neural networks? empirical evidence from the nn3 competition on time series prediction,” *International Journal of Forecasting*, vol. 27, pp. 635–660, 2011.

[9] R. D. Reed and R. J. Marks, *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks.* Cambridge, MA, USA: MIT Press, 1998.

[10] R. Rojas, *Neural networks - A systematic indroduction.* Berlin: Springer-Verlag, 1996.

[11] O. Kişi and E. Uncuoğlu, “Comparison of three back-propagation training algorithms for two case studies,” *Indian Journal of Engineering and Materials Sciences*, vol. 12, pp. 434–442, 2005.

[12] B. Karlik and A. V. Olgac, “Performance analysis of various activation functions in generalized mlp architectures of neural networks,” *International Journal of Artificial Intelligence and Expert Systems*, vol. 1, pp. 111–122, 2010.

[13] K. Hornik, “Approximation capabilities of multilayer feedforward networks,” *Neural Networks*, vol. 4, pp. 251–257, 1991.

[14] J. Heaton, *Introduction to Neural Networks for Java.* Heaton Research, Inc., 2005.

[15] F.-J. Chang, Y.-M. Chiang, and L.-C. Chang, “Multi-step-ahead neural networks for flood forecasting,” *Hydrological Sciences Journal*, vol. 52, 2007.

[16] M. Marcellino, “A comparison of direct and iterated multistep ar methods for forecasting macroeconomic time series,” *Journal of Econometrics*, vol. 135, pp. 499–526, 2006.

[17] J. Hoover, “How to track forecast accuracy to guide forecast process improvement,” *Foresight: The International Journal of Applied Forecasting*, pp. 17–23, 2009.

[18] R. J. Hyndman and A. B. Koehler, “Another look at measures of forecast accuracy,” *International Journal of Forecasting*, vol. 22, pp. 679–688, 2006.

[19] B. Krollner, B. Vanstone, and G. Finnie, “Financial time series forecasting with machine learning techniques: A survey,” in *European Symposium on Artificial Neural Networks: Computational and Machine Learning*, (Bruges, Belgium), April 2010.

[20] I. Alon, M. Qi, and R. J. Sadowski, “Forecasting aggregate retail sales: A comparison of artificial neural networks and traditional methods,” *Journal of Retailing and Consumer Services*, pp. 147–156, 2001.

[21] C. M. Ali and A. Haider, “Neural network models for inflation forecasting: an appraisal,” *Applied Economics*, vol. 44, pp. 2631–2635, 2012.

[22] G. Li and J. Shi, “On comparing three artificial neural networks for wind speed forecasting,” *Applied Energy*, vol. 87, pp. 2313—2320, 2010.

[23] W. Wong, M. Xia, and W. Chu, “Adaptive neural network model for time-series forecasting,” *European Journal of Operational Research*, vol. 207, pp. 807—-816, 2010.

[24] C. Hamzaçebi, “Improving artificial neural networks’ performance in seasonal time series forecasting,” *Information Sciences*, vol. 178, pp. 4550–4559, 2008.

[25] P. Danaher, T. Dagger, and M. S. Smith, “Forecasting television ratings,” *International Journal of Forecasting*, vol. 4, no. 27, pp. 1215–1240, 2011.

[26] C. Hamzaçebi, “Improving artificial neural networks performance in seasonal time series forecasting,” *Information Sciences: an International Journal*, vol. 178, pp. 4550–4559, 2008.