Holt-Winters Traffic Prediction on Aggregated Flow Data

Helge Brügner
Advisor: Johannes Naab, Jochen Kögel
Seminar Innovative Internet Technologies and Mobile Communications SS2017
Chair of Network Architectures and Services
Departments of Informatics, Technical University of Munich
Email: helge.bruegner@tum.de

ABSTRACT

This paper investigates how varying parameters for aggregation of flow data impact network traffic predictions based on the Holt-Winters filtering algorithm. Changes in aggregation level (minute to daily granularity) and aggregation functions (e.g., mean, percentiles) are considered as well as a change in training set length. These evaluations are based on real-world flow data collected in a large enterprise network. For this, the two traffic prediction use cases "anomaly detection" and "capacity planning" are considered. It is concluded that Holt-Winters in combination with small aggregation levels and the 90th percentile serving as aggregation function perform best for short-term prediction with focus on anomaly detection. For long-term prediction as a tool for capacity planning, it is demonstrated that Holt-Winters on daily granularity data on the given data set is not sufficient.

Keywords

flow data, traffic prediction, anomaly detection, capacity planning, exponential smoothing, Holt-Winters filtering

1. INTRODUCTION

Visibility in enterprise networks is crucial for success. Having insight into a network is important for multiple reasons:

- debugging and analysis in case of faulty behavior of applications and networks,
- retrospective analysis of anomalies in the network, such as attacks and outages, and
- detection of possible bottlenecks in the network for capacity planning.

While having the possibility to analyze past traffic is sufficient for the scenarios described above, enterprise network traffic is subject to certain patters. These patterns are usually seasonalities (daily, weekly, yearly) and underlying trends that can be leveraged to estimate the future behavior of the network traffic. Machine learning algorithms for time series analysis are able to capture these patterns through the fit of a function to the given data points by leveraging various parameter estimation procedures. The resulting model is then extrapolated to give estimations about future values in the series and can often also quantify the certainty of such predictions. Two main use cases for such prediction in enterprise network traffic were identified:

Anomaly detection allows detection of significant traffic changes that do not fit a previously learned pattern. Such anomalies can range from attacks to an extraordinary number of requests due to a special offer on the website.

Capacity planning focuses on forecasting the underlying trend of the network traffic to enable early identification of possible future capacity issues.

In the implementation, both use cases consist of three distinct steps. First, a model satisfying the use case requirements is fitted to a training data set. Second, a certain number of future data points are estimated using the resulting fit. Third, the prediction is compared to reference values to identify special cases for further investigation by network operators.

For both use cases, the number of predicted data points depend on the granularity of the data (i.e., the number of data points per period). Anomaly detection will require fine-grained data sets (minutes, hours) while for long-term prediction for capacity planning, coarse granularities (daily, weekly) suffice. However, the reference values required for comparison vary per use case. For anomaly detection, actual measured network traffic data forms the reference to identify large deviations from the prediction and potentially raise an alarm. Capacity planning would instead use a fixed threshold as reference that, when likely to be exceeded in the near future, could also generate a notification.

Since preserving fine-grained historic traffic data can be highly storage space-consuming, an alternate strategy to overwriting old records is to aggregate the traffic data with increasing age. A possible aggregation strategy could be to use a granularity of 1 min for the past 4 weeks, then aggregate the data in 5 min intervals for a further 2 months, and then preserve data in 1 h granularity for a full year. A history over past years could then be kept by aggregating in 24 h intervals for another 3 years. Multiple functions for aggregation exist, such as the mean, summation, percentiles, or minimum/maximum operators, and others.

It is unclear whether the use of aggregated training data in prediction algorithms can produce meaningful results and how changes in granularity and the use of different aggregation functions impact the forecast results. The focus of this work will be to study how different aggregation levels and -functions change the accuracy of forecasts made by the Holt-Winters filtering algorithm in context of the anomaly detection use case. Furthermore, the impact of the training

doi: 10.2313/NET-2017-09-1_04

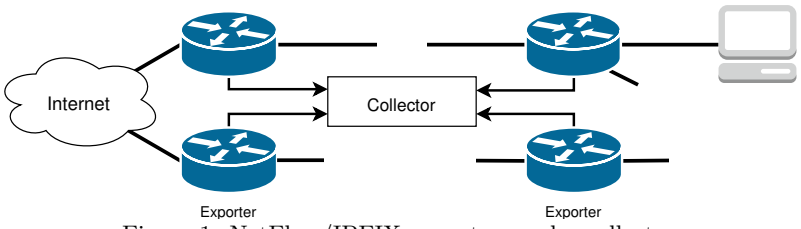


Figure 1: NetFlow/IPFIX exporters and a collector

set size to the forecast in capacity planning is investigated, again using Holt-Winters filtering. Two data sets exported from a large enterprise are used for the experiments; one with minute granularity and length of 4.5 weeks and a second set with daily granularity recorded over 3.5 years.

In section 2, this paper first describes two protocols for network traffic measurements. Then, the Holt-Winters filtering is introduced, followed by a comparison of quantification algorithms for prediction errors. After section 3 mentions related work, section 4 contains the analysis. First, in subsection 4.1, the test setups and the data sets used are described. Subsequently, subsection 4.2 compares aggregation levels, -functions, and -accuracies using the fine-grained data set with focus on the anomaly detection use case. The second part of the evaluation then analyzes different training set lengths and their impact on long-term traffic prediction accuracy for capacity planning. A conclusion in section 5 finally summarizes the results and presents possible future steps.

2. BACKGROUND

This section introduces two protocols commonly used in network traffic measurements, namely NetFlow and IP Flow Information Export (IPFIX). Then, both the additive and the multiplicative variant of Holt-Winters filtering are described. Finally, the most common algorithms to evaluate prediction errors are discussed.

2.1 Network Traffic Measurement

Network traffic measurements can be performed using either active or passive methods. Active methods such as Cisco IP SLA inject artificial traffic into the network by setting up special probes to actively measure the network performance. In contrast, passive methods monitor the existent traffic on key components in the network (e.g., on routers) by inspecting the transported packets. Since active probing requires changes in the network setup and anomaly detection as well as capacity planning potentially requires information about the packets flown, this paper will make use of passive network measurements.[5]

In contrast to the Simple Network Management Protocol (SNMP) that both allows to monitor and dynamically configure network devices[12], NetFlow and its successor IP Flow Information Export (IPFIX) focus on the monitoring of network devices (also referred to as "observation points") via "flows"[3]. Routers or switches usually function as such observation points. The corresponding RFC defines a flow as "set of packets or frames passing an Observation Point in the network during a certain time interval". Flows are identified by the "flow key", a user-defined set of packet properties used by the observation point to group encountered packets. Such a flow key can be, but is not limited to, the "5-tuple" consisting of source IP and port, destination IP and port, and protocol.[1][7]

The devices that host the observation points are called "exporters". They periodically send the packets containing flow information to "collectors". Collectors can potentially collect flows from multiple exporters simultaneously and perform pre-processing and aggregation for downstream analysis systems. A possible setup of exporters and collectors is shown in Figure 1.

Flows are closed and exported after either one of two specified timeouts occur. One of these timeouts is specified in the active case (i.e., packets of a certain flow are still arriving); the second occurs if the given flow has been inactive/idle for a certain amount of time (i.e., packets of a that flow were not seen during that time span). During these timeout intervals, the exporting devices caches packets and aggregates their relevant properties into a flow. Since NetFlow/IPFIX is capable of inspecting packets the observation points encounter, flow exports can also include information about packet content.[14] Exported values of observed packets can therefore be, but are not limited to, the number of bytes received, packet counters, or protocols observed, and range from OSI layer 2 (Data Link) to 7 (Application).[4]

Since SNMP is restricted to only exporting externally observable information about the encountered packets[12] (also referred to as the "interface view"), it cannot provide insight into the network apart from quantitative information regarding the packets encountered. Therefore, using NetFlow/IPFIX as a method to inspect the network flow is more promising since this potentially enables fine-grained analysis with focus on single hosts, subnets, or protocols.

2.2 Holt-Winters Exponential Smoothing

Time series analysis and forecasting can be performed using numerous different algorithms depending on the properties of the series. The network traffic data investigated in this paper shows seasonality (a pattern that repeats after a fixed number of iterations) while partially also exhibiting a trend over time. An filtering method that can incorporate both seasonality and trend is Holt-Winters filtering and belongs to the group of exponential smoothing procedures.

Exponential smoothing procedures are forecast algorithms that rely on updating equations to calculate predictions. In principle, exponential smoothing forecasts are "weighted averages of past observations, with the weights decaying exponentially as the observations get older"[9]. Multiple versions

doi: 10.2313/NET-2017-09-1_04

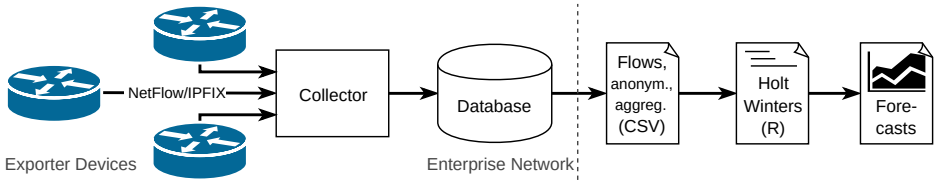


Figure 2: Test setup and evaluation work flow

Granularity	Obs./Week	Reasons
1 min	10080	Most common granularity for NetFlow/IPFIX
5 min	2016	Often used by network monitoring tools; originates from 5 min SNMP polling interval
1 h	168	Natural time interval
4 h	42	Divides 24 h granularity
24 h	7	Natural time interval

Table 1: The data set granularities and the respective number of observations per season (week)

exist, with the most basic form being simple exponential smoothing (SES). SES supports neither trend nor seasonality, but it forms the foundation for more sophisticated exponential smoothing models. Some of these models can support trend, some support seasonality, and some combine both.[2]

Another name for Holt-Winters filtering is "triple exponential smoothing" since it is based on three updating equations: ℓ_t models the level, b_t the trend, and s_t the seasonality of the time series. $\hat{y}_{t+h|t}$ represents the forecast at time $t+h$ given all the data points up to time t, and the constant m represents the seasonality (i.e., the number of observations per season). The equations are defined recursively, allowing for an iterative calculation of the predictions.[2][9]

Two variants of the Holt-Winters algorithm exist depending on how the a change in mean relates to the seasonal effect. The "additive" method is used if the seasonal effect is constant in each season (i.e., a change in mean does not impact the amplitude of the seasonal curve). Alternatively, the "multiplicative" covers the case when the seasonal effect is proportional to a change in the time series' mean. The additive updating equations are defined as follows:[2][9]

$$\begin{aligned}\hat{y}_{t+h|t} &= \ell_t + hb_t + s_{t-m+h^+_m}\\ \ell_t &= \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1})\\ b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}\\ s_t &= \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}\end{aligned}$$

In contrast, the multiplicative formulates the equations slightly differently:

$$\begin{aligned}\hat{y}_{t+h|t} &= (\ell_t + hb_t)s_{t-m+h^+_m}\\ \ell_t &= \alpha\frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1})\\ b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}\\ s_t &= \gamma\frac{y_t}{\ell_{t-1} + b_{t-1}} + (1-\gamma)s_{t-m}\end{aligned}$$

The equations for level, trend, and seasonality depend on three smoothing parameters α, β, and γ, respectively. To find appropriate values for the parameters, the most common approach is to first define an error term $e_t = y_t - \hat{y}_{t|t}$ with y_t as the training data point at time t and $\hat{y}_{t|t}$ as the estimated value of the algorithm. A minimization of the sum-of-squared-errors term $\sum e_t^2$ can then be used to estimate the three parameters.[2]

For the prediction of network flow data, Holt-Winters filtering seems reasonable since it provides a sufficient formulation of a model with support for both trend and seasonality as exhibited by the measured flow data.

2.3 Prediction Error Quantification

To evaluate the performance of forecasts it is necessary to quantify their error (i.e., the difference between predicted and actual values). Such calculations can either be performed in-sample or using a test set. In-sample error computation compares the fitted model to its training data, while out-of-sample error computation uses a subset of the whole data set for training and then compares the model's predicted values with the expected values from the test set. The former tends to support overfitting (i.e., a lack of generalization by specializing on the training data), which is why this section will focus on out-of-sample error computation.[9]

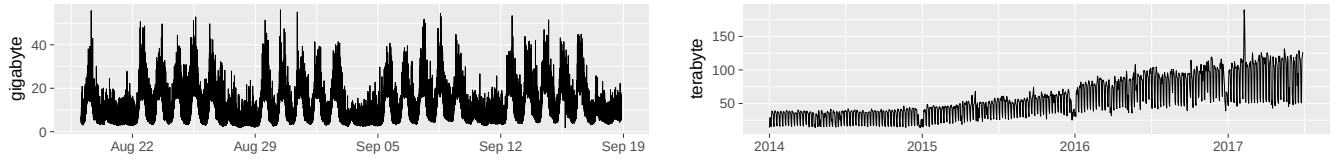
If different models are compared against the same data set, simple scale-dependent errors are often sufficient, while inter-data set comparisons require scale-independent error calculations to achieve comparability.[9] An example of inter-data set comparison is prediction of traffic for an entire network and comparing it with another prediction for a subnet of that network – the scale might differ by orders of magnitude. Since aggregation of the data set might potentially change the data's scale, the evaluations will use scale-independent errors for the calculation.

Scale-dependent errors calculate the error between predicted and actual values by subtraction: $e_i = y_i - \hat{y}_i$. If the scale of the test data is changed, e_i changes, too. Therefore, scale-independent percentage-based errors remove the scale by dividing by y_i: $p_i = 100^{e_i}/_{y_i}$. A popular variant of such errors is the mean absolute percentage error (MAPE):[9]

$$E_{\text{MAPE}} = \text{mean}(|p_i|)$$

Although more sophisticated versions such as the symmetric MAPE exist, Hyndman and Koehler [10] suggest not to use percentage-based scale-independent errors due to their as-

doi: 10.2313/NET-2017-09-1_04



(a) The short data set (b) The long data set

Figure 3: Plots of the original data sets

#	Date Range	Gran.	Periods	Data Pts.	Comments
1	19.08.2016, 13:00 – 18.09.2016, 20:14	1 min	≈ 4 (weeks)	44355	clear weekly seasonality visible, minimal trend, small amount of outliers
2	01.01.2014 00:00 – 29.06.2017, 01:00	1 min	≈ 3.5 (years)	1278	data aggregated by collector, damped trend visible, relatively clean data except a single large anomaly in 2017, 365 days per period except in 2016 due to the leap day

Table 2: The originally exported data sets used to evaluate the aggregation

sumption of meaningful zeros (i.e., a value of 0 indicates that the quantity measured is absent) as well as possible unstable calculations (e.g., an assumption must be that $y_i \neq 0$).[9]

Another group of accuracy measures not based on such assumptions are the "scaled" scale-independent errors. These make use of a scaling factor calculated using a naïve forecast on the training data set. Such scaling ensures that the result does not depend on the scale of data sets and can therefore also be used for inter-data set comparisons. Three variants to calculate such errors exist and can be used interchangeably depending on what patterns are found in the data. For data exhibiting seasonality, Hyndman [10] suggests using the seasonal naïve forecast ($q_{\mathrm{j,seasonal}}$) by assuming that current observations in the time series have the same value as the respective data point in the previous period. The mean absolute scaled error (MASE) is then defined by computing the mean over all errors:

$$q_{\mathrm{j,seasonal}} = \frac{e_j}{\frac{1}{T-m}\sum_{t=m+1}^{T}|y_t - y_{t-m}|}$$

$$\mathrm{MASE} = \mathrm{mean}(|q_j|)$$

If MASE = 1, the model used to compute the forecast is equally as good as the naïve forecast model used to compute the scaling factor. If the value is > 1, the forecast model performs worse, and consequently, if the result is < 1, it performs better.[10]

Due to the seasonality of the data and the recommendation to not use scale-independent, percentage-based errors, this paper uses MASE in combination with a the seasonal naïve forecast to compute prediction errors. The data sets are therefore divided into training and test set for fitting and evaluation, respectively.

3. RELATED WORK

In [6], Hellerstein et al. identified and inspected the same use cases (prediction for capacity planning and anomaly detection) for traffic prediction on a single web server. They base their calculations on the number of HTTP operations per second aggregated in 5 minute intervals in a data set with 8 months length. Based on this data, they construct a model from ground up that resembles the anomaly-free case and incorporates time-of-day, day-of-week, and monthly seasonality. By then analyzing the remaining error after applying the model to measured data, they propose an algorithm to detect "change points" that, as they argue, indicate anomalies. Additionally, they successfully 6apply the model to capacity planning by predicting the web server traffic multiple months into the future.

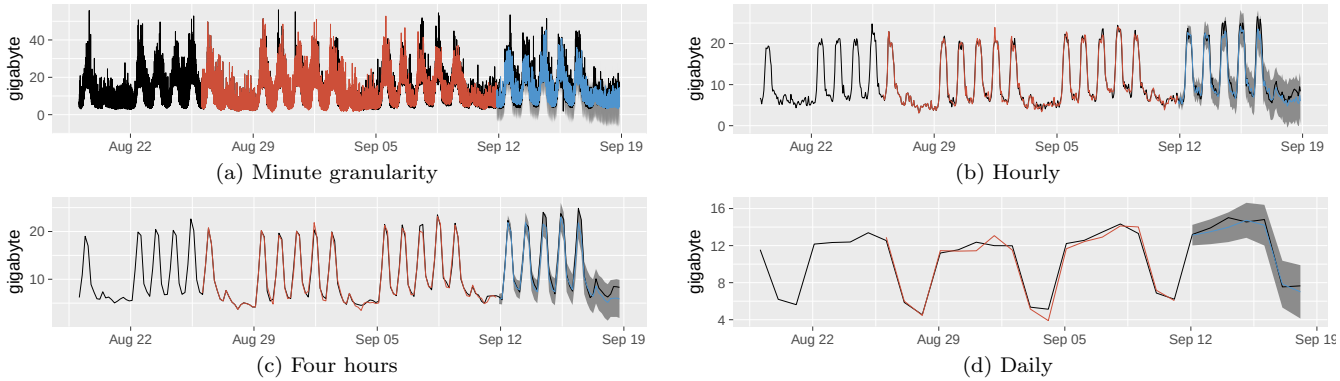
Münz [13] primarily focuses on anomaly detection in networks based on flow-level measurement data collected via NetFlow/IPFIX. Instead of only using bytes per flow, he incorporates additional metrics that are derived from the flow data for detection by using an approach called "multi-metric analysis". For the analysis, he compares multiple techniques for the detection, such as exponential smoothing, control charts, and principal component analysis. In addition to the pure detection of anomalies, he also describes automatic procedures to classify detected anomalies by relevance as well as to identify their causes. He concludes that the success of anomaly detection is highly dependent on the selection of metrics for classification.

In [15], Taylor and Letham propose a "scalable", "intuitive", "fast", and "accurate" way for forecasting business time series. For this, they make use of a model with components for growth, seasonality, and holidays. If the learning process is fed with information about, for example, holidays, the algorithm can learn the impact these days have on the time series, and consequently produce more accurate forecasts if such days occur again. They then developed a framework ("Facebook Prophet"[1]) that implements the algorithm they describe in R and Python. During the research, the author of this paper attempted to test the framework using a minute-granularity data set with around 44000 data points. The framework was unable to predict these points, and contact with the developers yielded that this was both due to the fine granularity as well as the resulting large data set[2].

In his work for the social media company Twitter, Kerjawal [11] developed the "AnomalyDetection" R package[3] to detect outliers in time series that describe the usage of Twitter's so-

[1] `https://github.com/facebookincubator/prophet/`
[2] `https://github.com/facebookincubator/prophet/issues/215`
[3] `https://github.com/twitter/AnomalyDetection`

doi: 10.2313/NET-2017-09-1_04



(a) Minute granularity

(b) Hourly

(c) Four hours

(d) Daily

Figure 4: Comparison of 1 min, 1 h, 24 h granularities aggregated using MEAN (the plot's scales differ)

cial media platform. The main use cases are the detection of rises in social engagement (e.g., during sporting events) and also identify malicious activity and its causes (e.g., spammers or bots). Instead of using fitting, the framework implements a method called "seasonal hybrid extreme studentized deviate" testing. This algorithm first applies decomposition of the time series into, for example, trend and seasonality, and then test if any of the residuals indicate anomalies. Due to its focus on anomaly detection, however, the framework cannot be used for prediction of the time series data.

4. EVALUATION

This section first introduces the test setup and the two data sets used. Then, it is investigated how a change in parameters impacts the traffic predictions produced with Holt-Winters. First, different aggregation levels are compared. Second, it is shown how aggregation functions impact the forecasts. Third, accuracies of forecasts with varying aggregation level and -function are compared. Finally, it is shown how a change in training set length impacts long-term forecasts.

4.1 Test Setup & Data Sets

Figure 2 displays the process from the data set acquisition to the generation of the test results. The exporter devices on the left are shown qualitatively; the actual number of monitored devices is approximately 50. The NetFlow/IPFIX exports are collected by a single collector instance that logs and pre-processes the flows and loads them into a database system. In the production environment, an analysis system would directly connect to this database and extract the data required for network analysis. However, in the test environment, the data is exported as CSV files including a timestamp and the total bytes flown in the network. This helps to make the tests reproducible and eases both anonymization and aggregation of the flow data. Subsequently, the data sets are fed into the Holt-Winters implementation built into the language R. The resulting model is then used to predict and plot the forecasts. Although an alternative implementation to Holt-Winters in R exists with the "ets" function from the package "forecast", the basic "HoltWinters"-implementation was preferred since ets does not allow for forecasting periods longer than 24 data points[8].

The data sets used are listed in Table 2. Set 1 was recorded in August and September 2016 over a period of a month in a network of a large enterprise with a minute granularity. The traffic of all NetFlow/IPFIX exporters in the network was aggregated by timestamp using a summation to represent the total number of bytes flown in the network at a certain point in time. Furthermore, the set only contains a small number of anomalies (e.g., in the form of holidays). A plot of the data set is shown in Figure 3a.

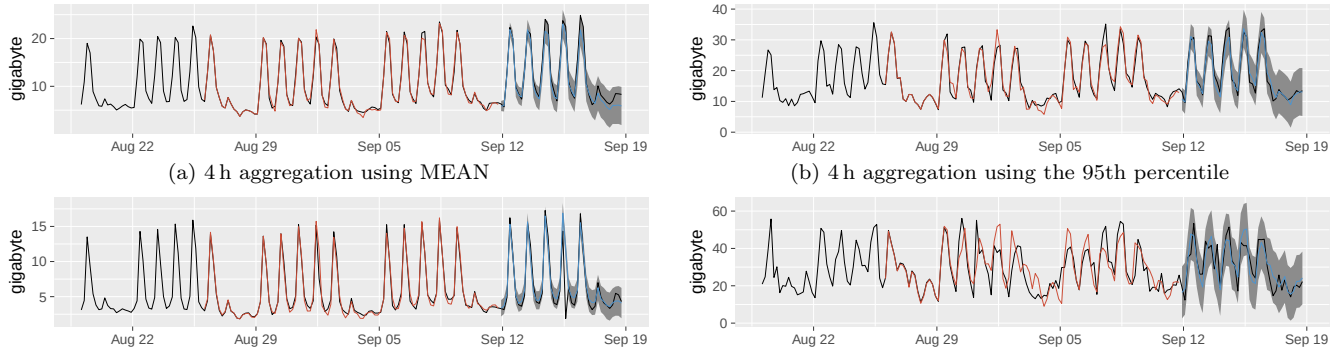
Data set 2 was aggregated by the collector prior to exporting it from the enterprise network to a granularity of 24 h using the MEAN function. Such aggregation is necessary due to storage constraints in the collecting system – a data set length of 3.5 years in a finer granularity was not possible while the data set was collected. As shown in Figure 3b, the data set contains a trend and both weekly and yearly seasonality. Furthermore, the seasonal effect is proportional to the time series' mean. A single large outlier is visible in the beginning of 2017, and furthermore, the data set experiences irregular seasonality due to the leap day added in the 3rd period (the year 2016).

The different levels of aggregation are produced by first dividing the minute granularity data into subsets with the size of the given aggregation factor f and then applying the aggregation function. Since the data sets are not aligned with the beginning of a day, the aggregation starts with the first observation and then aggregates the first f data points. This may result in an incomplete last aggregation, but this effect is minimal due to the large number of data points in general. The granularities investigated are listed in Table 1.

Seven aggregation functions are compared during the evaluation. MEAN and SUM are interesting since they both incorporate all data points from the respective aggregation interval, while MEDIAN (equal to the 50th percentile) has the interesting property of not being influenced by outliers. The 80th and 95th percentile have the same property as the MEDIAN and are also popular measures used for data sets. The two functions MIN and MAX are investigated as well since they could potentially reveal interesting properties about bottoms and peaks of the data set.

MASE is the measure used to evaluate the prediction accuracy. Since MASE requires a training set to compute the scaling factor and a test set to compute the actual accuracy, the data sets are split into such sets. In data set 1, the given

doi: 10.2313/NET-2017-09-1_04



(a) 4 h aggregation using MEAN

(b) 4 h aggregation using the 95th percentile

(c) 4 h aggregation using MIN

(d) 4 h aggregation using MAX

Figure 5: Comparison of MEAN, 95th percentile, MIN, and MAX aggregation on 4 h granularity data (the plot's scales differ)

4.5 periods (weeks) are splitted into approximately 3.5 periods for training and one period for testing, which comes close to the popular split of 80 % to 20 % for training and testing, respectively. Furthermore, the models are trained and evaluated on the same granularity. If, for example, hourly aggregation is used in the test set, hourly aggregation is also used in the training set.

4.2 Test Results & Comparison

With the data sets and the process described above, it is possible to test numerous combinations of aggregation levels and -functions to determine the best fit for the given use cases. The first three comparisons are based on the minute granularity data (data set 1) since this is the most useful for aggregation, and is therefore focused on the anomaly detection use case. Subsequently, it is investigated how Holt-Winters predictions and accuracies vary based on the size of the training set. Since it is useful to have a data set for such variations over a long period of time, these comparisons are based on data set 2.

The prediction plots all follow the same color scheme. The black line resembles the collected/aggregated flow data, while red and blue show the Holt-Winters fit and prediction, respectively. The gray area surrounding the blue prediction line is the 95 % confidence interval.

4.2.1 Aggregation Levels

Since it is of interest how the different levels of aggregation impact the ability of the Holt-Winters algorithm to predict the flow data, different aggregation levels are compared first. The graph shown in Figure 4a displays the data that has been exported with minute granularity originally which was then fitted with a additive Holt-Winters model. The black line shows that due to the fine granularity of a minute, a lot of noise is visible. This also appears to impact the training phase of the model, since both fit and prediction are noisy. Such a fit is likely the result of overfitting.

The data that was used to generate Figure 4b is based on the same data set as the previous plot but is aggregated with factor 60 (1 h granularity) using the MEAN function. The noise mostly is compensated by the aggregation function, which results in a less noisy fit. Besides preventing overfits, the noise reduction has another interesting effect. At noon and midnight (i.e., the peaks and bottoms of the fit), small drops and increases in capacity used become visible that were invisible in the 1 min granularity data. The increase at noon is likely caused by the employee's lunch breaks while the increase at night could be the result of scheduled nightly backups. If aggregated with 4 h granularity, the noise is further reduced, as visible in Figure 4c.

The graph shown in Figure 4d is plotted using flow data aggregated to 24 h granularity by again making use of the MEAN function. The noise that is still visible in the 1 h aggregation mostly disappeared, but as expected, any subday effects disappear as well. This, however, reveals different interesting aspects in the data. First, it can be seen that during the week, a peak in the traffic either appears during Thursday or Wednesday. Furthermore, the plot indicates that the peak day seems to change every other week. Furthermore, due to the lack of noise, the Holt-Winters fit appears to fit the data well.

4.2.2 Aggregation Functions

Figure 5 shows a comparison of four aggregation functions (MEAN, 95th percentile, MIN, MAX) on the same aggregation level, namely 4 h. The plots in Figure 4c and Figure 5a are both based on 4 h aggregation with MEAN for comparison.

As expected, the aggregations based on MEAN and SUM only differ in the scale. Both functions can therefore be used interchangeably depending on whether it is required to preserve the original scale (i.e., bytes per minute with MEAN) or the total number of bytes flown in each time interval (e.g., bytes per 4 h with SUM. Furthermore, the plots show that SUM, MEAN, MEDIAN and the percentiles preserve the seasonality well by either compensating (SUM, MEAN) or ignoring outliers (MEDIAN, percentiles).

Aggregation with the MIN function as shown in Figure 5c appears to also preserve seasonality at 1 h and 4 h granularity, because the data set appears to have a clear lower bound in the number of bytes tracked. For 5 min granularity a clear lower bound does not exist. Similarly, predictions based on 24 h MIN aggregation do not perform well since only the bottoms at night are preserved. Both result in a bad MASE value as shown in Figure 3. However, with 4 h granularity, MIN can reveal drops in the bytes captured, as seen on Thursday of the fourth week in the plot. The MAX func-

doi: 10.2313/NET-2017-09-1_04

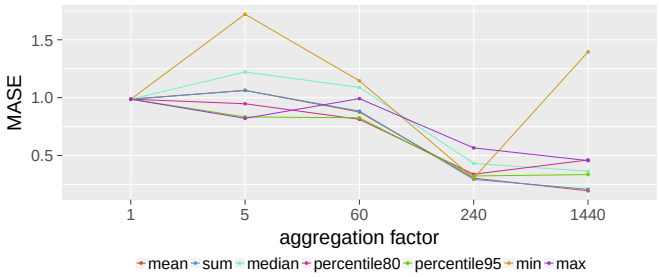
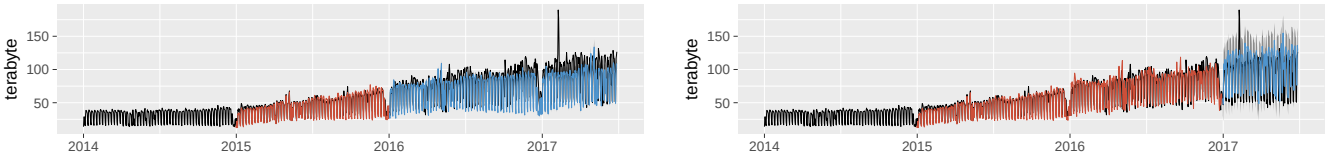


Figure 6: Comparison of prediction accuracy using different aggregation levels

	MEAN	SUM	MEDIAN	80th PT.	90th PT.	MIN	MAX
1 min	0.98	0.98	0.98	0.98	0.98	0.98	0.98
5 min	1.06	1.06	1.22	0.95	0.86	1.72	0.82
1 h	0.88	0.88	1.09	0.81	0.73	1.15	0.99
4 h	0.30	0.29	0.43	0.34	0.31	0.30	0.57
24 h	0.19	0.21	0.36	0.46	0.24	1.40	0.46

Table 3: MASE by aggregation levels and -functions



(a) 2 years training, MASE = 2.05

(b) 3 years training, MASE = 1.60

Figure 7: Comparison of training set lengths and the corresponding error

tion is shown in Figure 5d. In contrast to MIN, MAX does not preserve structure at any aggregation level due to the apparent lack of a clear upper bound in traffic.

4.2.3 *Aggregation Accuracies*

Figure 6 and Table 3 display the MASE of the Holt-Winters with varying aggregation levels and -functions. Since an aggregation factor of one results in the same data set for all aggregation functions, all plots start at the same error value for 1 min granularity.

The error for 1 min aggregation is insignificantly smaller than one. This is a sign of overfitting – the noisy fit of Holt-Winters in Figure 4a supports this hypothesis. Furthermore, the very similar error of the MEAN and SUM aggregation functions reveal that both perform almost equally, which is again due to the similarity of both aggregation function. Therefore, both are overlapping in the graph (here, the MEAN plot is covered by SUM). Aggregation using the 90th percentile performs most stable and results in MASE values greater than one6 for all granularities. The performance of the 80th percentile is close, which means that both functions ignore outliers well. With an increasing aggregation factor, outliers are more likely to be compensated, thus allowing MEAN and SUM to outperform the quantiles.

In general, a clear trend towards a better fit with an increasing aggregation level is visible. This can be explained by the decreasing number of data points and the resulting, increasing compensation of outliers, which prevents Holt-Winters from fitting the error and therefore modeling the data better. One clear exception is the MIN function – its plot shows outliers with large jumps and no clear trend. Although the plot of the MAX function accuracy does not show similar large jumps or poor MASE results, plots of its fit (e.g., Figure 5d) exhibit large confidence intervals in the prediction due to the lack of a clear structure in the aggregated data.

4.2.4 *Training Set Size*

The previous analyses primarily focused on short-term prediction due to the structure of the data set used. Data set 2, however, suits well for long term predictions due to its length of 3.5 years. The original data set is plotted in Figure 3b. Due to the length of the data set, a comparison of fit accuracies with varying training set lengths (2, 2.5, 3 years) is possible. Such a comparison is crucial, since it can reveal for what period of time a forecast can be trusted. Furthermore, the multiplicative formulation of Holt-Winters is used since it allows to capture the multiplicative effect in the data.

The resulting forecasts with their MASE result are shown in Figure 7. Looking at Figure 7a shows that the fit can estimate the data relatively well until mid-2016 and then clearly underestimates due to the growing trend in late 2016. Similar results with a MASE value of 2.02 have been collected with a training set length of 2.5 years, however, the plot has been omitted for simplicity. In Figure 7b, the fit overestimates the actual trend immediately.

The fits for 2 and 2.5 years of training have in common that the prediction is off towards the end of 2016 due to the additional leap day. This effect is not visible in the third plot since the beginning of Christmas 2016 is still part of the training set. However, in the last fit, the irregular seasonality learned might impact the prediction at the end of 2017.

In general, the poor MASE result of ≥ 1.60 for all training set lengths indicates that the prediction based on Holt-Winters was unable to learn the pattern and trends of this data set sufficiently.

5. CONCLUSION

In conclusion, short-term prediction for use case 1 appears to be promising. The best function for the given data is the 95th percentile-function since it results in good MASE results for all aggregation factors. MEDIAN, MIN, and MAX should generally be avoided unless special requirements have

doi: 10.2313/NET-2017-09-1_04

to be met. Provided that the data contains a seasonal, stable lower bound, MIN could, for example, be used to detect anomalies resulting in bandwidth drops (e.g., a failure of a core router) as a scenario of the anomaly detection use case.

Given that the proper aggregation function is used, short-term prediction appears to work well for small aggregation levels (potentially between 5 min-1 h). Aggregation generally seems to be a powerful tool to compensate for outliers to prevent overfitting and, therefore, enable better detection of anomalies. Anything with a coarser granularity than approximately 1 h should, however, not be used for anomaly detection since this potentially results in a loss of important sub-day effects. In such a case it may become impossible to detect anomalies either due to the delay in aggregation or suppression of such outliers by the aggregation function.

For long-term prediction required for the capacity planning use case, the best aggregation function still remains unclear due to the lack of data. The MAX function could potentially be a good candidate since it may preserve peaks crucial for capacity planning. Daily granularity appears to be sufficient, since it preserves weekly seasonalities and reduces the number of data points in the data set to a size suitable for fast learning.

It also remains unclear which prediction algorithm works best for long-term prediction. Due to the poor MASE results and the lacking possibility to incorporate special days as well as seasons with different lengths, Holt-Winters may not be a good candidate in this scenario – although this claim should be verified by testing further data sets. Simpler algorithms such as a simple fitting with a trend component only or more sophisticated frameworks such as Facebook Prophet may constitute alternatives and could be subject to further investigation.

Future work could potentially also investigate more sophisticated aggregation functions, such as an aggregation by business hours (i.e., only use values between 6AM and 6PM for aggregation). Additionally, it may be interesting to analyze subsets of the data (e.g., specific interfaces, protocols). Furthermore, a better training could result in even better fits with Holt-Winters. A possibility for that might be to train the model with a coarse-grained training set ($\geq$1 h) and evaluating it against a fine-grained test set ($\leq$5 min).

Acknowledgement

This work was partly performed at IsarNet Software Solutions GmbH and funded as part of the AutoMon project by the German Federal Ministry of Education and Research (BMBF) with contract number 16KIS0408K.

References

[1] P. Aitken, B. Claise, and B. Trammell. *Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information*. RFC 7011. 2013. DOI: 10.17487/RFC7011. URL: https://rfc-editor.org/rfc/rfc7011.txt.

[2] C. Chatfield. *The Analysis of Time Series: An Introduction*. 6th ed. Texts in statistical science. Boca Raton, Fla. and London: Chapman & Hall/CRC, 2004. ISBN: 9781584883173.

[3] B. Claise. *Cisco Systems NetFlow Services Export Version 9*. RFC 3954. 2004. DOI: 10.17487/RFC3954. URL: https://rfc-editor.org/rfc/rfc3954.txt.

[4] B. Claise, P. Aitken, and N. Ben-Dvora. *Cisco Systems Export of Application Information in IP Flow Information Export (IPFIX)*. RFC 6759. 2012. DOI: 10.17487/RFC6759. URL: https://rfc-editor.org/rfc/rfc6759.txt.

[5] L. Cottrell. *Passive vs. Active Monitoring. Stanford Linear Accelerator Center*. Stanford Linear Accelerator Center, 2001. URL: https://www.slac.stanford.edu/comp/net/wan-mon/passive-vs-active.html (accessed on 08/13/2017).

[6] J. Hellerstein, F. Zhang, and P. Shahabuddin. "Characterizing Normal Operation of a Web Server: Application to Workload Forecasting and Problem Detection". In: *In Proceedings of the Computer Measurement Group*. Morgan Kaufmann, 1998, pp. 54–61.

[7] R. Hofstede et al. "Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX". In: *IEEE Communications Surveys & Tutorials* 16.4 (2014), pp. 2037–2064. DOI: 10.1109/COMST.2014.2321898.

[8] R. Hyndman. *Forecasting with long seasonal periods*. 2010. URL: https://robjhyndman.com/hyndsight/longseasonality/ (accessed on 09/29/2010).

[9] R. J. Hyndman and G. Athanasopoulos. *Forecasting: Principles and practice*. Heathmont: OTexts, 2016. ISBN: 978-0987507105. URL: https://www.otexts.org/fpp (accessed on 06/20/2017).

[10] R. J. Hyndman and A. B. Koehler. "Another look at measures of forecast accuracy". In: *International Journal of Forecasting* 22.4 (2006), pp. 679–688. ISSN: 01692070. DOI: 10.1016/j.ijforecast.2006.03.001. URL: http://www.sciencedirect.com/science/article/pii/S0169207006000239.

[11] A. Kejariwal. *Introducing practical and robust anomaly detection in a time series. Twitter Blog*. Twitter Blog, 2015. URL: https://blog.twitter.com/engineering/en_us/a/2015/introducing-practical-and-robust-anomaly-detection-in-a-time-series.html (accessed on 06/01/2017).

[12] D. R. Mauro and K. J. Schmidt. *Essential SNMP*. 2nd ed. Sebastopol, Calif. and Farnham: O'Reilly, 2005. ISBN: 978-0-596-00840-6.

[13] G. Münz. *Traffic Anomaly Detection and Cause Identification Using Flow-Level Measurements*. Vol. 2010,06. Network architectures and services. München: Network Architectures and Services Techn. Univ. München, 2010. ISBN: 3937201122.

[14] G. Sadasivan et al. *Architecture for IP Flow Information Export*. RFC 5470. 2009. DOI: 10.17487/RFC5470. URL: https://rfc-editor.org/rfc/rfc5470.txt.

[15] S. J. Taylor and B. Letham. *Forecasting at Scale. Facebook Research*. 2017. URL: https://facebookincubator.github.io/prophet/static/prophet_paper_20170113.pdf (accessed on 08/13/2017).