COME UP WITH
A THESIS
TITLE

by

James Howard

A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Computer Science).

Golden, Colorado

Date ____________________

Signed: ____________________
James Howard

Signed: ____________________
Dr. William Hoff
Thesis Advisor

Golden, Colorado

Date ____________________

Signed: ____________________
Dr. Big Boss
Professor and Head
Department of Electrical Engineering and Computer Science

ABSTRACT

Forecasting the occupancy of buildings can lead to significant improvement of smart heating and cooling systems. Using a sensor network of simple passive infrared motion sensors densely placed throughout a building, we perform data mining to forecast occupancy a short time (*i.e.*, up to 60 minutes) into the future. Our approach is to train a set of standard forecasting models to our time series data. Each model then forecasts occupancy a various horizons into the future. We combine these forecasts using a modified Bayesian combined forecasting approach. The method is demonstrated on two large building occupancy datasets, and shows promising results for forecasting horizons of up to 60 minutes. Because the two datasets have such different occupancy profiles, we compare our algorithms on each dataset to evaluate the performance of the forecasting algorithm for the different conditions.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Seasonal Auto Regressive Moving Average Model	SARIMA

ACKNOWLEDGMENTS

That you everyone

For those that shall follow after.

CHAPTER 1
INTRODUCTION

TODO CHANGE these paragraphs to argue more for the limitations of control systems relying on fixed data - instead discuss how they NEED accurate forecasts. Perhaps discuss freeway traffic control lights and filling up the freeways if miss timed.

According to the U.S. Department of Energy [1] energy for heating and cooling accounts for approximately 35 - 45% of the total expenditure within a building. With such a large investment of energy being used to regulate the temperature of a building, possible areas of improvement are heavily sought after. Fully automated buildings with control systems to automatically heat and cool individual rooms or spaces have been designed [2, 3] to reduce building energy usage while maintaining proper temperatures throughout the building. These systems are complex and require knowledge of room usage, and in more complex systems room occupancy, as inputs into system control models.

While many factors such as ambient temperature, ventilation air flow, room volume, etc. affect the time it takes to heat or air condition a room, it still takes on the order of many minutes to bring a room to a stable desired temperature [4]. Due to this lag in changing the temperature of a room, it is not sufficient for control systems to begin air conditioning a room upon initial occupancy. Thus, to ensure that room temperatures are appropriate, smart building control systems typically rely on set schedules of occupancy. These systems may use scheduled occupancy unto 24 hours into the future to determine current heating and air conditioning control [5].

However, what happens to the system when building traffic deviates from its schedule? Let us consider a university classroom building. While not often, professors will occasionally cancel class. What should a smart control system do during this time? It doesn't make sense for the system to heat or cool the room as though it was occupied. The system should adapt to the changing environment. Similarly, what happens if snow has caused many of the students and faculty to stay at home? Ideally the building control system should identify a lower than average number of occupants and modify its heating or cooling schedule to account for such situations. In both of these scenarios, a set schedule is insufficient to produce an optimal heating or cooling schedule for the building.

As another example of the usage of complex control systems consider the roadways of the United States. Optimal timing of traffic lights on major roadways across the United States could account for approximately a 22% reduction in emissions along with a 10% reduction in fuel consumption [6]. As of 2005 the total estimated fuel savings would amount to approximately 17 billions gallons of motor fuels annually. This traffic light timing does not only consider city lights, but also takes into account freeway onramp volume lights.

In the United States, traffic light timings are often determined by an individual from the department of transportation standing near the light and manually determining a timing schedule, or in some cases multiple schedules to account for peak traffic times and non-peak times [7]. These schedules are then fixed and are changed either when roadways change to make new timings necessary or if petitioned by local citizens. These timings are then either set in local control box for that traffic light or by a central control system for the city.

As with the building scenario, what happens when the traffic deviates from normal? Fixed timings will not be able to account for changes in traffic. Inclement weather scenarios should require different timings then sunny days. Lights near large sporting events likely require different timings than typical evenings. Even schedules were to be made for such scenarios, there exist certain scenarios for which schedules likely can not be made manually such as lane blockages due to accidents.

In both of the above environments, the control systems have to account for future occupancy of the environment. It is inadequate for these systems to use only current data to control the system. Accurate forecasts of the systems usage are necessary to produce optimal control systems.

1.1 Problem and Objective

We define a time series dataset used within as $\{x_t^{(m)}\}$. Each x_t^m is an aggregate of the readings from sensor m reading at time block t. Our objective in this work is then:

1. Produce an accurate forecast for $x_t^(m)$ δ time steps into the future.
2. Minimize worst case forecasts
3. Keep approach unsupervised

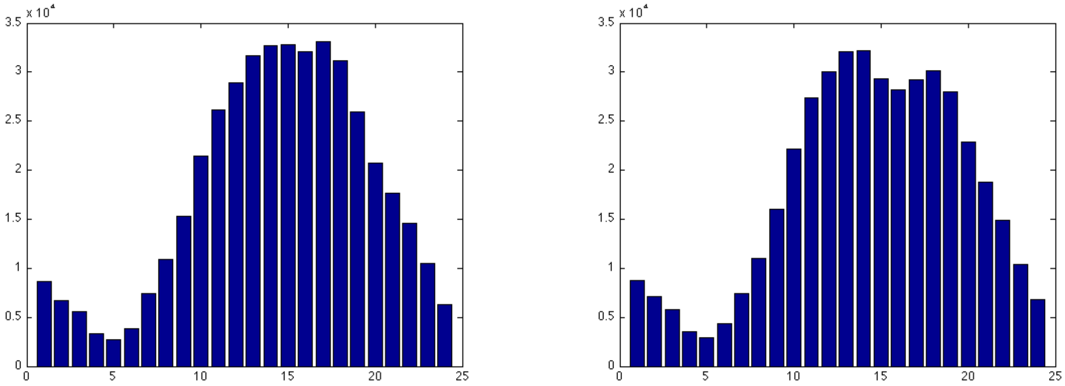
DISCUSS SOME OF OUR ASSUMPTIONS AND ASSUMED CONSTRAINTS HERE. Constraints: Human controlled system Repetitive etc Talk about how everything is empirical, but we will show some evidence of our assumptions bearing fruit later

To assist in constructing models we make the assumption that data is generated from activities produced by human controlled entities moving through the environment. Also we assume that the activities are repetitive and on some schedule. The result of these assumptions is that from such scheduled movement we get sensors which have a spatial correlation and that for example, from week to week on the same day display similar trends. Much of the research on traffic forecasting makes a similar assumption.

1.1.1 A motivating example

To illustrate an example of the need to minimize worst case forecasts, we present the following example. The Denver Broncos, as with most American Football teams are incredibly beloved. In 2010 had an average attendance of 74,908 [**?**]. This attendance, combined with additional fans flooding downtown to patronize bars and restaurants creates an interesting affect on Freeway traffic patterns. Unsurprisingly, prior to the game there is an increase in total traffic volume along a freeway which is near the Broncos' Stadium. What is perhaps surprising is a nearly 20% drop in total traffic volume along the same stretch of Freeway during the game. Figure Figure 1.1 shows the total counts of Denver traffic for each hour of the day averaged for the first four Sunday Broncos home games and for the first four

Sunday away games in 2010. Comparing figure Figure 1.1(a) with figure Figure 1.1(b) it is evident that a noticeable change in traffic patterns occur from approximately noon until approximately 6:00 pm. This traffic change corresponds with a 2:05pm kickoff time for the game.



(a)

(b)

Figure 1.1: Total number of cars passing major highway sensors on Sundays in September and October 2010

Such a significant difference in traffic patterns should lead to a change in traffic light control. We have all encountered the frustrating scenario of attempting to leave one of these sporting events. Traffic light are often still on predefined timings and the situation arises where a large number of vehicles attempt to get through an traffic light controlled intersection in one direction with almost no vehicles attempting to get through the intersection in the perpendicular direction. Ideally the light timings should be changed to fix this scenario and increase the green light time in the direction with multiple cars. Of course, optimally other light timings will also need to be changed to account for the increase volume of traffic along certain paths within the city.

Traditional parametric forecasting models have difficulty accounting for these different traffic patterns and the problem becomes more difficult when when it is considered that the Broncos may play a Sunday night game or a Monday night game. Thus our another goal of our approach is to handle these significant deviations from normal traffic patterns which are typically the causes of worse case forecasting scenarios.

1.2 Approach

To satisfy our first objective of producing an accurate forecast for dataset $x_t^(m)$ δ time steps into the future. We have created an ensemble forecasting model based off of the Bayesian combined forecaster created by Petridis [**?**].

DISCUSS HOW THIS RELATES TO OBJECTIVE 1 Discuss how we can only show improvements empirically on our work, but that if datasets follow similar structure, we would expect similar results on other data.

THEN disucss we plan to split our approach into two sections, a background model and activity models. RELATE THIS TO OBJECTIVE 2 here.

We also assume that sufficient deviations from our forecasting function are often not the result of noise, but are due to an activity that does not commonly occur on that day. Because such activities can overlap or occur at different times with varying amount of background noise present, it is a difficult task for one parametric model to accurately encapsulate all possible combinations of activities. In an environment with many activities that could occur at multiple different times such combinations may be prevalent. Past work doesn't address the problem of overlapping activities.

TODO Create a motivating example off broncos game times residuals with each model Should tie the introduction together nicely

r5cm

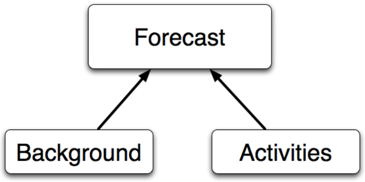


Figure 1.2: Forecasting is based on activities and a background model

Our approach is to split the problem of forecasting into two parts: development of a background model and the development of a set of activity models. Our background model is represented by a seasonal autoregressive integrated moving average (ARIMA) model. To model activities, we propose comparing different models from activity recognition literature along with a new model which we propose here. Forecasting is then performed using an ensemble predictor taking outputs from all trained activity models and the background seasonal ARIMA model. Figure 1.2

DISCUSS HOW ALL OF THIS still allows objective 3 to hold.

1.3 Contributions

The contributions to the field of unsupervised traffic forecasting from this work are:

- Use of activity models for improved forecasting accuracy of multiple time steps into the future.
- An improvement to a combined Bayesian prediction model to improve forecasting accuracy
- A combined prediction model using an ensemble forecaster and activity models to improve short term forecasts during the presence of anomalous activities

1.4 Structure of the Proposal

REDO THE STRUCTURE The remainder of this proposal is outlined as follows. Section two reviews current work related to traffic prediction and activity modeling. Section three gives a summary of each dataset used in this work. Section four details specific pieces of the overall approach. All of the approach is not solved and where possible this section details potential ways to proceed with each unsolved part of the approach. Finally section five is a time line of when the remaining work is expected to be completed.

CHAPTER 2

DATASETS

This chapter contains dataset information

2.1 Notation

As already stated, we define a time series dataset used within as $\{x_t^{(m)}\}$. Each x_t^m is an aggregate of the readings from sensor m reading at time block t.

Forecasts for a given model k from the set of all models K are represented by

$$\bar{T}_{t+1}^{k,m} = f(T_t, ..., T_1; \theta_k). \tag{2.1}$$

Thus the forecast of T_{t+1} is a function of all past data and some trained parameterization θ_k for that model.

In this work we need to forecast more than one time step into the future. Future forecasts are performed through iterative one step ahead forecasts. Also for this work we forecast a model for each individual sensor and for convenience drop the m from our forecasting notation. An example of a forecast two time steps ahead of current time t is given by

$$\bar{T}_{t+2}^{k} = f(\bar{T}_{t+1}, T_t, ..., T_1; \theta_k). \tag{2.2}$$

Such a forecast is simply the forecast for one time step into the future but now with the forecasted value of $\bar{T}_{t+1}$ used as the most recent datapoint to forecast $\bar{T}_{t+2}$. Forecasting in this nature allows for forecasts any number of time steps into the future.

2.2 Forecast measurements

Discuss MAPE, MASE, RMSE, and our approach here

TODO: Perhaps remove this to another section This could be represented by a hypothesis function $h(x, \delta)$. We use mean absolute scale error (MASE) [8] and root mean squared error (RMSE) as cost functions to compare with other previously implemented techniques.

2.3 MERL Dataset

The Mitsubishi Electronic Research Labs dataset is derived from a collection of over 200 passive infrared sensors place densely throughout the 7th and 8th floor of a research office building. The sensors are placed roughly two meters apart on the ceilings, creating a dense sensing area with little non-sensed space. Readings are taken at the millisecond level, but due to the sensors' settling times the inter-detection time of motion is approximately 1.5 seconds.

The data was collected from March 2006 through March 2008 and there are roughly 53 million sensor readings. This building is similar to most office buildings with a number of personal offices along with labs and conference rooms. Employees have roughly set schedules and holidays are observed as normal.

The counts of sensor activations have been aggregated every 10 minutes. Because of the lack of significant motion in the night, we look only at activations that occur between 6:00am and 7:00pm daily. A plot of the average activations of all Wednesdays for a single sensor along with a range of one standard deviation is given in Figure **??**.

Peak motion unsurprisingly occurs during the middle of the day corresponding to lunch time. There is another small peak of motion near the start of the day corresponding to people entering. Near the end of the day, instead of a peak there is a region corresponding to high variance. This seems to imply that while people enter at roughly the same time, there is a significant variance on when people leave the building.

2.4 Colorado School of Mines Dataset

The Colorado School of Mines dataset is a collection of 50 passive infrared sensors mounted on the ceiling of the second floor of a class and office room building. The density of the sensor placement depends on the location within the building. Outside the auditorium in the lower right of Figure **??** is a dense collection of sensors placed approximately every few meters. Throughout the rest of the building the sensors are placed roughly every 5 meters.

Data was collected for one academic school year from 2008 to 2009 and there are more than 23 million sensor readings. To acquire readings, the sensors were polled every second and recorded data if motion was detected.

This dataset is much different than the MERL dataset as classes typically provide activity on a rigid schedule during the day. Also as students have exams and projects, late night motion is sporadic based on the time of year. The counts of sensor activations have been aggregated over every 10 minutes. Despite occasional late night motion during exam time, most nights have no significant motion. For this reason we focus on data between 7:00am and 7:00pm daily. A plot of the average activations of all Wednesdays for a single sensor along with a range of one standard deviation is given in Figure **??**. The defined peaks in the dataset correlate to class start and end times when most students will be in the hallways of the building.

2.5 Denver traffic dataset

Discuss the denver traffic dataset here.

CHAPTER 3
MODELS

Discuss the relevant models which make up the BCF here.

3.1 Seasonal ARIMA model

The Auto Regressive Moving Average Model (ARMA) or derivations on its form (Auto Regressive Integrated Moving Average, Seasonal Auto Regressive Moving Average, *etc*) have been used in numerous forecasting applications from economics to vehicle traffic systems. While we have been unable to find ARMA based models used on building occupancy data directly, we have found it used to forecast building energy usage and vehicle occupancy [**?** **?** **?**]. Its forecasting accuracy is quite strong and it can serve as a strong baseline of comparison for a forecasting problem.

Due to that fact that our building data has periodic trends and a non stationary mean, a seasonal ARIMA model is best suited to fit our data from the class of ARMA models. The seasonal ARIMA model is defined as:

$$\phi_p(B)\Phi_p(B^s)\nabla^d\nabla_s^D T_t = \theta_q(B)\Theta_Q(B^s)e_t \tag{3.1}$$

where $\{T_t\}$ is our observed time series and $\{e_t\}$ represents an unobserved white noise series ($e_t \sim N(0, \sigma^2)$)the values of which are computed through model training and are not known a priori. B is the backshift operator which is a function that allows access to older time readings. For example $BT_t = T_{t-1}$ and $B^5T_t = T_{t-5}$. ∇_s^D is the seasonal difference operator ($\nabla_s^D T_t = (1 - B^s)^D T_t$)and $\phi,\ \Phi,\ \theta,\ \Theta$ are trainable parameters.

Seasonal ARIMA models are notated as

$$ARIMA(p, d, q)(P, D, Q)_s \tag{3.2}$$

where p is the number of autoregressive terms, d is the number of differences and q is the number of moving average terms. P, D, and Q all correspond to the seasonal equivalents of

Table 3.1: The parameter values that were fit for MERL and CSMBB datasets for a Seasonal ARIMA model

Dataset	p	d	q	P	D	Q	s
MERL	0	0	1	0	1	5	78
CSMBB	0	1	1	0	1	3	72

p, d, and q. The parameter s is the seasonality of the model. For a full discussion of seasonal ARIMA models see Box and Jenkins [**?**].

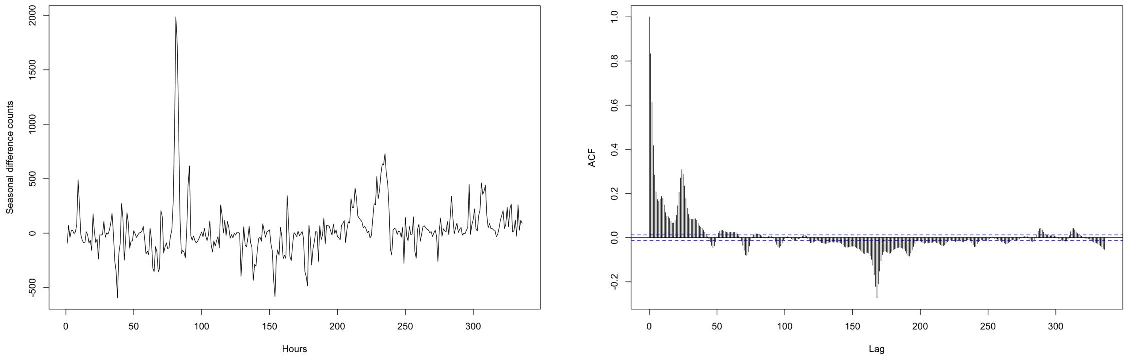
Finding the correct values of p, d, q, P, D, Q, s is traditionally a hard problem. To fit our parameters we use a method similar to Williams [**?**]. As a verification of our model, we applied the LJung-Box test [**?**] on our set of residual data for each model. This tests if any of the auto correlation values on the residual dataset are significantly different from 0. To be valid, the LJung-Box test should return a value of $p > 0.05$. Both residual sets passed: $p = 0.9964$ for MERL and $p = 0.1072$ for CSMBB. Our final model parameters can be seen in Table Table 3.1. Notice that the season is different for each model due to a difference in window of time for each day that we extracted data.

Forecasting from this model is performed by iteratively forward feeding values of the model into itself. Since the set of residuals e from a properly trained seasonal ARIMA model is described by a white noise Gaussian distribution $N(0, \sigma^2)$, we can take the expected value of the residual at time e_{t+1} to be 0. This leaves us with the following forecasting equation:

$$\phi_p(B)\Phi_p(B^s)\nabla^d\nabla_s^D T_{t+1} = \theta_{q-1}(B)\Theta_{Q-1}(B^s)e_t \qquad (3.3)$$

3.1.1 Fitting a Seasonal ARIMA

The underlying math of ARMA models stems from a linear filter operating on input from a stationary stochastic process. ARIMA models were created to handle non-stationary data by differencing the data to induce stationarity. Thus, a necessary step in fitting a ARIMA model to the data is first to determine the steps necessary to make the time series weakly stationary. For a time series to be weakly stationary two conditions must be satisfied: The



(a) Seasonal difference counts

(b) Autocorrelation of seasonal difference counts

Figure 3.1: One week seasonal difference counts and autocorrelation over a two week period.

expected value of $x^{(t)}$ is the same for all t and the covariance between any two observations depends only on the lag.

In general it is difficult to prove stationarity, but there exists a number of methods which assist in determining if a time series is close enough to stationary to be modeled by an ARMA model. Visual inspection of both the raw data and the autocorrelation function is a useful tool to test for stationarity. Figure **??** shows the raw counts and autocorrelation values at hourly lags of vehicle counts for one sensor over a two week period. The data shows no constant mean and thus can not be stationary. The graph of autocorrelation values shows local peaks every 24 hours with a significant peak at one week of lag (168 hours).

Intuitively a one week seasonal difference should yield a stationary time series and visually, outside of an anomalous reading, Figure Figure 3.1(b) shows such stationarity. Applying the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) [**?**] test for stationarity on the seasonally differenced data confirms the visual inspection. Using R's implementation of KPSS gives a p-value greater than 0.1. This is significantly higher than the standard value to reject the stationarity hypothesis of 0.05.

Figure 3.1(a) Figure 3.1

Most of the input parameter values for seasonal ARIMA models tend to be 0, 1, 2, or 3 [?]. Due to this small range of input values the total input space is relatively small (Six parameters with four possible values equates to $4^6 = 4096$) allowing us to apply a brute-force search for the best model. Model performance is determined by the Akaike information criterion (AIC) [?]. Our optimal model is a seasonal ARIMA $(1, 0, 1)(0, 1, 1)_{168}$.

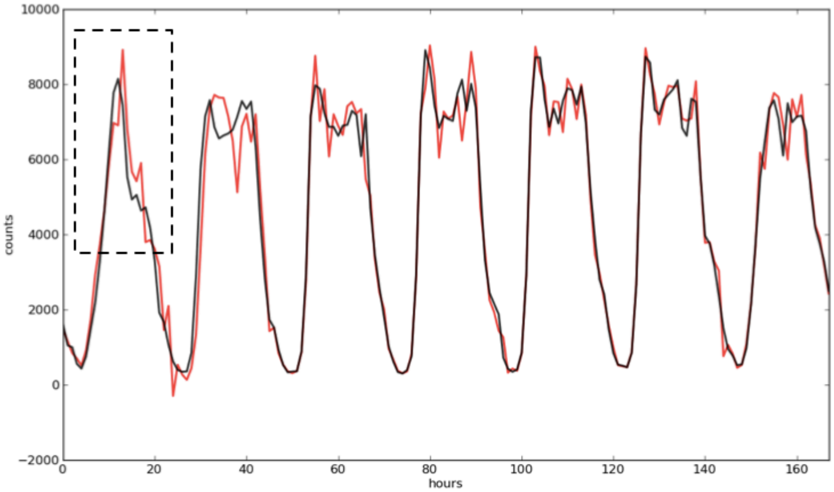


Figure 3.2: One-step ahead prediction for a sample week. Black line is original data. Red line is forecasted data. Dotted box shows an example of mis-forecasting due to a broncos game.

Figure Figure 3.2 shows an example of one-step ahead prediction performed on a sample week of test data. The dotted line boxes a time when a Broncos game was occurring. Forecasting during the Broncos game was initially low while traffic was unusually high as people were traveling to the game and then too high for much of the duration of the game. This pattern of mis-forecasting is inline with the pattern demonstrated in Figure ??. The mean absolute percentage error (MAPE) for this week was approximately 8.2%. This MAPE is close to the results from other authors on other vehicle traffic datasets [? ?].

3.2 Historic average

This model is simply the per day average of readings at each time step. For certain types of data this model is has been shown to be more accurate than seasonal ARIMA forecasting [**?**], specifically when the data has a strong historic correlation. Average forecasts have the advantage of being extremely computationally fast and having a forecast accuracy that does not depend on the forecasting horizon. This result will be shown later.

3.3 Time delayed neural networks

Time delayed neural networks are a special subset of regression neural networks where the input data is a local history of data from the time series. Commonly the output is a single point forecast from that same time series at some point $t + \delta$ in the future. The form of our 1 hidden layer time delayed neural network is:

$$T_{t+1} = \phi\{\sum_{j=1}^{J} w_j \psi_j \left[\sum_{l=0}^{m} w_{ji} T_{t-l\delta} + w_{j0} \right] + w_0\} \tag{3.4}$$

where $\phi()$ is a linear activation function on the output layer and $\psi()$ is the standard sigmoid function. A visual representation of the node architecture of a time delayed neural network is displayed in Figure Figure 3.3.

Forecasting is performed by computing the output for a $m + 1$ length window of time and then iteratively forecasting a set of time steps in the future by using forecast data as inputs into the next forecast.

The number of input nodes and hidden nodes for each dataset is given in Table Table 3.2.

Table 3.2: Number of delayed input nodes and hidden nodes for MERL and CSMBB datasets

Dataset	Delayed input nodes	Hidden nodes
MERL	15	8
CSMBB	12	8

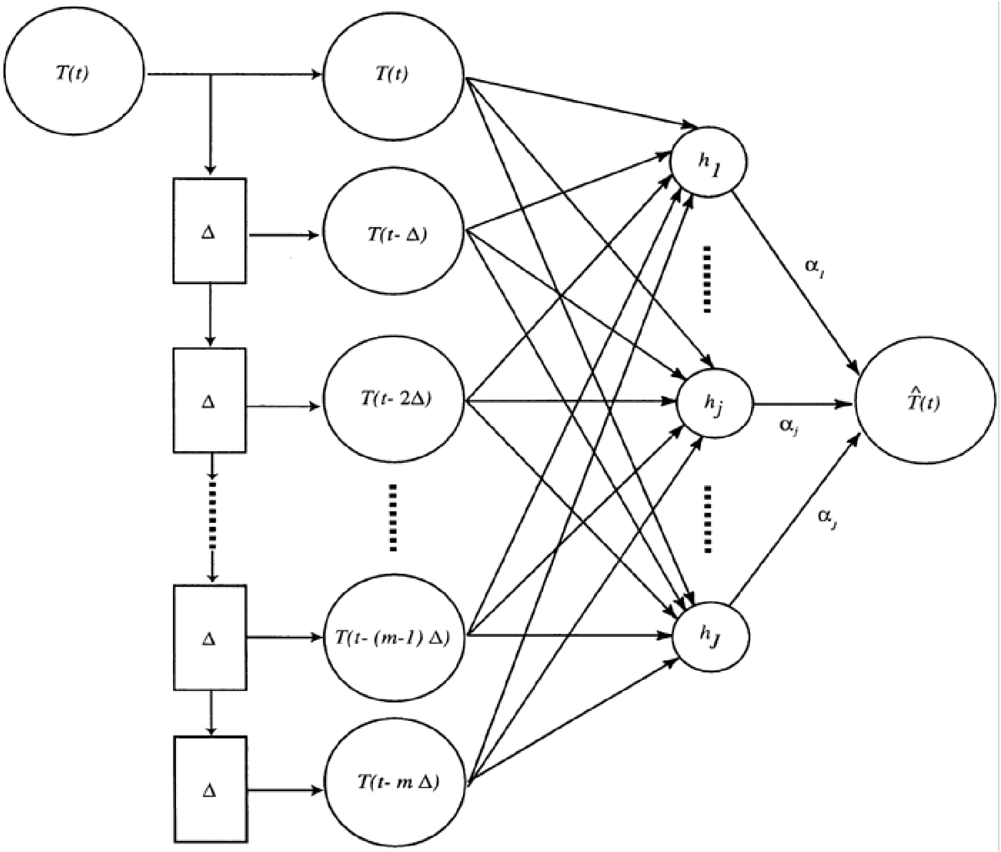


Figure 3.3: Architecture of a time delayed neural network with $m + 1$ inputs and J outputs [**?**].

Table 3.3: RMSE forecast values per model for a horizon equal to one.

Dataset	ARIMA	TDNN	AVG	SVM	BCF	BCF-TS
MERL	5.5	2.67	4.35	2.29	2.28	**2.26**
CSMBB	10.94	14.13	27.14	11.01	8.72	**8.72**

3.4 Support Vector Regression

Support Vector Machine Regression (SVM) offers a powerful way to forecast time series. It has been used in the past successfully to forecast travel times for vehicle traffic [**?**].

As training SVM's is not done in the same way as other time series models, we first had to transform our dataset to a series of examples with a fixed window. For a fixed window of size w, training input data is of the form $\{T_t, T_{t-1}, ..., T_{t-w+1}\}$. Target data is of the form T_{t+1}.

Thus the training examples which we provided to our SVM was $\{T_{t+1}, [T_t, T_{t-1}, ..., T_{t-w+1}]\}$.

To perform SVM training we used the popular *libsvm* package and as parameter selection is a notoriously difficult problem for SVM. We followed the guidelines as outlined by Hsu, Chih-Chang and Lin, creators of the libsvm package [**?**]. We first scaled the data by normalizing it between $[0, 1]$. Then we searched for our best values of C, ϵ and γ using the root mean squared error of the validation set a factor to determine performance of those parameters.

For both the MERL and CSMBB datasets we used a window of 5. This happened to be the same window length used by [**?**].

Table 3.3

CHAPTER 4
ENSEMBLE APPROACH

write about approach here

4.1 Bayesian Combined Forecasting

The BCF approach [**?**] is one of several types of methods which attempt to combine other forecasting models for time series. We selected this forecasting method over other multiple model forecasting methods (such as mixture of experts or ensembles of neural networks) due to its modularity and strong statistical backing. BCF is modular in that it allows for the component forecasting models to come from any trained forecaster with a well defined distribution of the forecaster's mis-forecasts. Its statistical backing comes from its direct derivation from Bayes' rule.

This section derives the BCF approach for readers unfamiliar with it and then describes some modifications of the approach which improves its performance for our application.

4.2 Notation

We define the time series dataset used in these models as $\{T_t^{(m)}\}$. In our application the data used for these models comes from a set of M binary infrared sensors. Each T_t^m is a 10 minute aggregate of the readings from sensor m reading at time block t.

Forecasts for a given model k from the set of all models K are represented by

$$\bar{T}_{t+1}^{k,m} = f(T_t, ..., T_1; \theta_k). \tag{4.1}$$

Thus the forecast of T_{t+1} is a function of all past data and some trained parameterization θ_k for that model.

In this work we need to forecast more than one time step into the future. Future forecasts are performed through iterative one step ahead forecasts. Also for this work we forecast

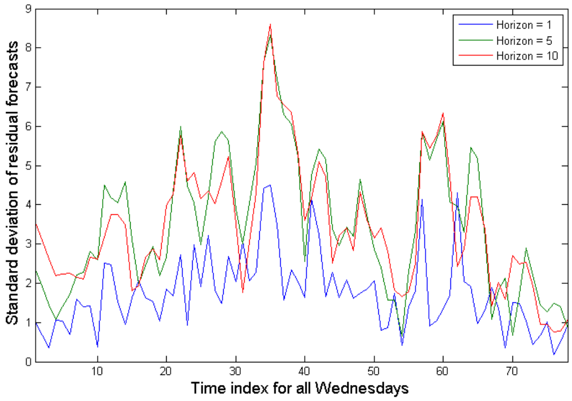


Figure 4.1: Standard deviation of support vector machine residuals for all Wednesdays in MERL dataset. Time index represents 10 minute intervals from 6:00am to 7:00pm.

a model for each individual sensor and for convenience drop the m from our forecasting notation. An example of a forecast two time steps ahead of current time t is given by

$$\bar{T}^k_{t+2} = f(\bar{T}_{t+1}, T_t, ..., T_1; \theta_k). \tag{4.2}$$

Such a forecast is simply the forecast for one time step into the future but now with the forecasted value of $\bar{T}_{t+1}$ used as the most recent datapoint to forecast $\bar{T}_{t+2}$. Forecasting in this nature allows for forecasts any number of time steps into the future.

4.3 Bayesian Combined Forecasting Derivation

To derive BCF we first assume the existence of K models. From these K models, we want to create a probability distribution on a new random variable z that is used to determine if model k is the correct model from which to forecast at time t. To do this we use the notation of Petridis [**?**] and define p^k_t as follows

$$p^k_t = p(z = k | T_t, ..., T_1). \tag{4.3}$$

From here we apply Bayes rule and get

$$p^k_t = \frac{p(T_t | z = k, T_{t-1}, ..., T_1) \cdot p(z = k | T_{t-1}, ..., T_1)}{p(T_t, ..., T_1)}. \tag{4.4}$$

Notice that $p(z = k | T_{t-1}, ..., T_1) = p^k_{t-1}$. Thus we can create a recursive estimation based on prior p^k_t.

With recursive values for p_t^k and replacing $p(T_t, ..., T_1)$ with a conditional probability on z we get

$$p_t^k = \frac{p(T_t|z = k, T_{t-1}, ..., T_1) \cdot p_{t-1}^k}{\sum_{j=1}^K p(T_t|z = j, T_{t-1}, ..., T_1) \cdot p_{t-1}^j}. \tag{4.5}$$

We use the empirically observed forecasting error for each model to estimate $p(T_t|z = k, T_{t-1}, ..., T_1)$. The forecasting error for a given model at time t is

$$e_t^k = \bar{T}_t^k - T_t. \tag{4.6}$$

We can use these forecasting errors to estimate a probability distribution for each model on the random variable e_t^k. This is typically modeled as a white noise zero mean Gaussian process. For our work, we represent this as a distribution of error terms with some parameterization ω_k. Thus for each model the probability error distribution function on the model error random variable is given by $q(e_t^k; \omega_k)$.

The final equation for the posterior probability of a given model k is

$$p_t^k = p(z = k|T_t, ..., T_1) = \frac{p_{t-1}^K \cdot q(T_t - \bar{T}_t^k; \omega_k)}{\sum_{j=1}^K p_{t-1}^j \cdot q(T_t^j - \bar{T}_t^j; \omega_j)}. \tag{4.7}$$

An example of these changing normalized posterior probabilities for a small section of the MERL dataset is shown in Figure Figure 4.2.

Forecasting using BCF is done by either computing a weighted forecast δ time steps into the future for each forecasting model or by simply selecting the model with the highest likelihood. For this paper we forecast using a weighted forecast of all models. The forecasting equation is

$$T_{t+\delta}^{ALL} = \sum_{k=1}^K p_t^k \cdot \bar{T}_{t+\delta}^k. \tag{4.8}$$

4.4 BCF Modifications

In this subsection we discuss a number of modifications to maximize the effectiveness of BCF for our data. We refer to the modified BCF algorithm as Bayesian Combined

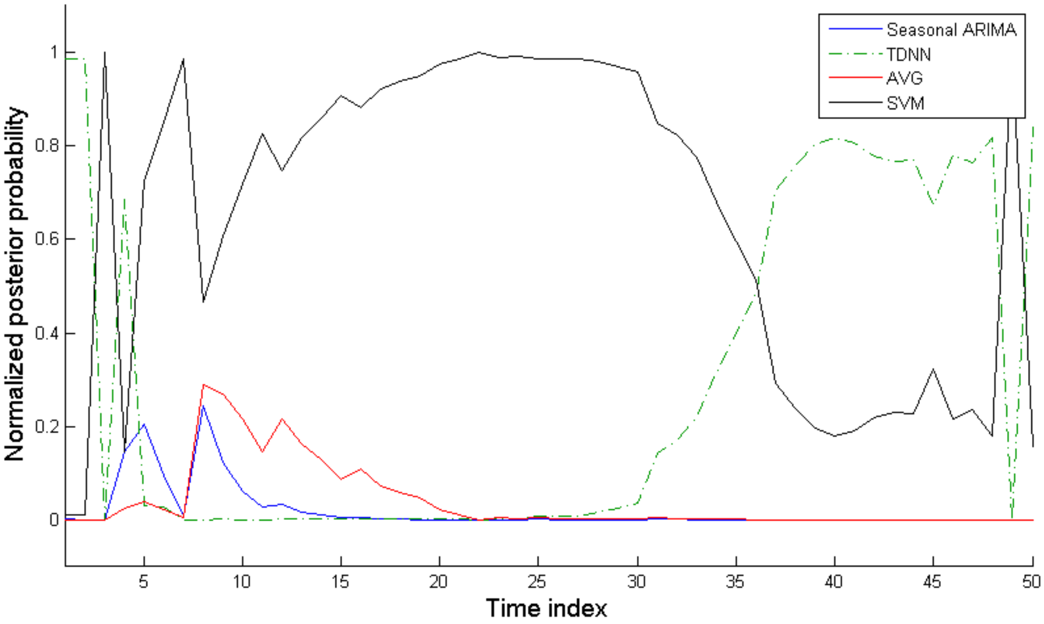


Figure 4.2: Normalized posterior probabilities of component models on a section of MERL dataset.

Forecasting for multiple Time Steps or BCF-TS for short. These modifications enable BCF to work with forecasting horizons greater than one in the future.

4.4.1 Forecast δ time steps into the future

Traditional implementations of BCF in other domains [**? ?**] are interested only in 1 time step ahead forecasts. For our work we require forecasts that are δ steps ahead which requires a small change to the BCF method. Instead of generating a model's error distribution from

$$e_k^t = \bar{T}_t^k - T_t = f(\bar{T}_t, T_{t-1} ..., T_1; \theta_k) - T_t. \tag{4.9}$$

The error distribution is instead generated from

$$e_k^t = f(\bar{T}_t, ..., \bar{T}_{t-\delta+1}, T_{t-\delta}, ..., T_1; \theta_k) - T_t. \tag{4.10}$$

The reason for this change is due to the assumption that our error distribution is an accurate representation forecasting accuracy. The forecasting error distribution for models at 1 time step into the future is not necessarily the same as models at δ time steps. Thus we compute a different error distribution for each forecast time step.

4.4.2 Improving model error distributions

Despite other implementations of BCF using fixed error distributions, our data has clear daily trends. For some of our models, the forecasted residuals follow these same trends. See Figure Figure 4.1 for an example of how the forecasting error distribution for a trained support vector regression model on the MERL dataset depends on the time and on the forecasting horizon.

To represent a more realistic error distribution instead of a fixed white noise Gaussian that is commonly used in the literature, we fit a Gaussian for each 10 minute slice of a given day. The data from the MERL dataset was used from 6:00am to 7:00pm. The thirteen hours of data used per day represent 78 time slices. For example taking the data for each time slice for each Wednesday results in 78 Gaussian error distributions for each forecasting horizon. These Gaussians are computed from a validation set representing 20% of our data. It is from this set of models error distributions that we compute BCF.

As a possible improvement to this set of error distributions, we note that using a generalized autoregressive conditional heteroskedastic (GARCH) model [?] or some other appropriate model to forecast future variance based on local and historic changes in variance would likely outperform our time based average Gaussian models. GARCH models are similar to seasonal autoregressive moving average models which we use as one of our component forecasting models.

4.4.3 Model selection thresholding

Diebold [?] cautions against the use of forecasting using a Bayesian combination of models in all cases. Diebold points out that under certain situations a convex combination of forecasts for models may not be optimal, and cases exist where taking negative likelihood weighting may be optimal. These conditions are likely to arise during instances where the data may not be accurately described by any of the forecasting models.

Furthermore when such cases where no model is able to provide an accurate forecast, then it is often the case that forecasts come from the worst model.

To combat this case, we have implemented a model selection threshold h_k. If the likelihood of all component models is below h_k, then we forecast from only the model which is historically the most accurate based on our validation set.

The threshold is different for each model, and should depend on the error distribution of the model. In practice we have found that 2σ serves as a good threshold. Basing the threshold on σ is useful as it provides a threshold value which does not depend on e_t^k. For a zero mean Gaussian the probability of the 2σ threshold is

$$p(2\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-2}. \tag{4.11}$$

Because the Bayesian combined forecasting approach is iterative, it is possible that a long section of forecasts that indicate one model correct or incorrect can lead to likelihood underflow. Due to this problem we adjust our normalized likelihoods so that no model may reach a value below 0.001. This empirically chosen value is low enough to not have a great impact on forecasts while still being high enough to allow model likelihoods to change quickly.

4.5 Results of Ensemble

BCF and BCF-TS (BCF with our specific set of modifications) were trained and tested using all component models described above. All of the models were trained on 60% of the total datasets. Another 20% was used for model validation and the final 20% used for testing. All results shown below are on the test set only. Figure Figure 4.3 shows an sample section of test data from the MERL dataset along with BCF-TS forecasts for horizons of 1 and 5. As expected as the forecasting horizon increases the forecasts become less accurate.

It is common for one model's normalized posterior probability to be near one when that model is currently accurate. Figure Figure 4.4 shows as example of this behavior. From time index 1 to 8, the SVM component model has a posterior probability near 1.0 and as a result BCF-TS forecasts nearly completely from this model. Then from time index 9 on

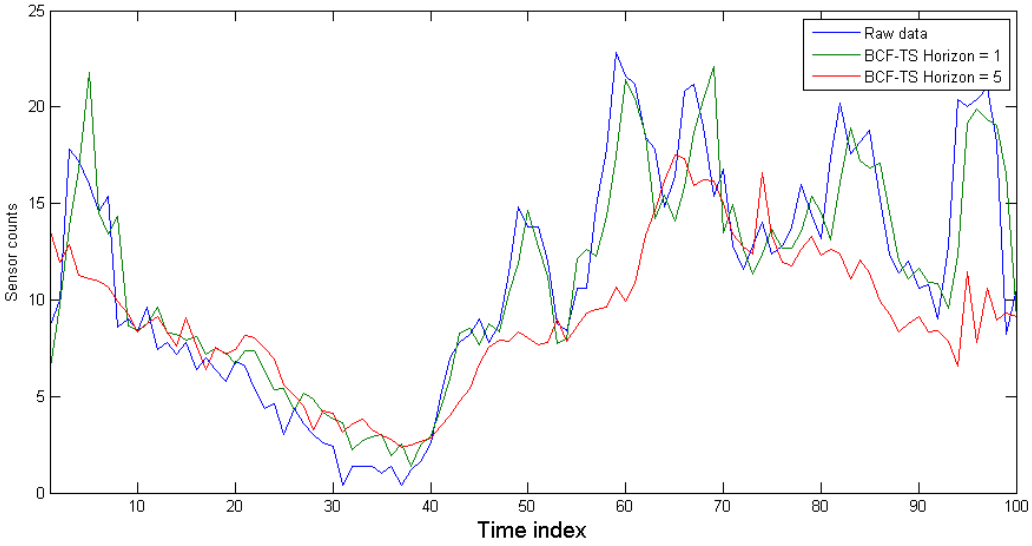


Figure 4.3: A comparison of forecasts at various horizons against real data for an sample time segment using BCF-TS.

the model's posterior probability is lower and as a result BCF-TS uses other model for its combined forecast.

Figure Figure 4.5 shows the results of the root mean squared error (RMSE) of forecasts across a forecast horizon up to 10 time steps (100 minutes) into the future for each model. These plots show that BCF-TS has the lowest error. However, the average model shows itself to be a strong indicator of future activity for forecasts beyond 60 minutes into the future. Forecasts were performed for significantly longer horizons, but the results were uninteresting as the total RMSE of models converged to roughly the values at a forecasting horizon of 10 time steps.

Table 4.1: Run times (in seconds) for each forecasting horizon.

Algorithm	1	2	3	5	8	10
Average	0.001	0.001	0.001	0.001	0.001	0.001
ARIMA	0.043	0.045	0.046	0.053	0.058	0.063
SVM	0.048	0.910	0.137	0.227	0.357	0.444
TDNN	20.87	21.60	21.82	21.28	22.73	22.50
BCF-TS	20.97	21.26	21.27	21.34	21.57	21.63

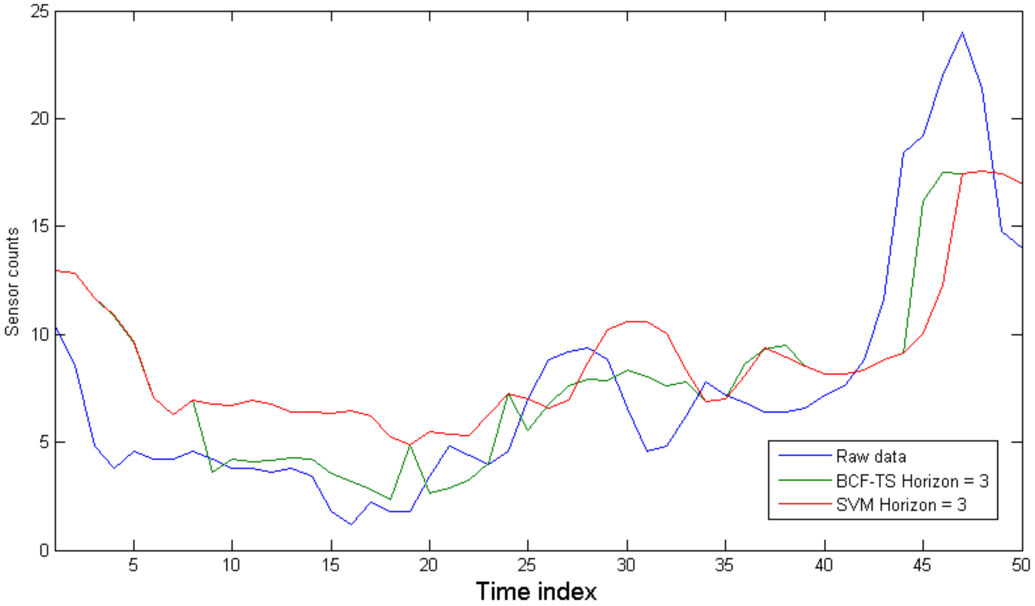
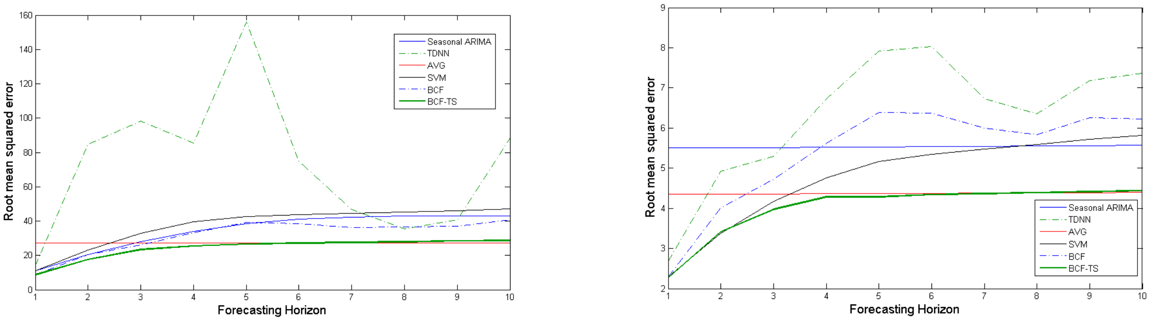


Figure 4.4: A comparison of BCF-TS and SVM forecasts at horizon equal to three against real data.

In the CSMBB dataset the Seasonal ARIMA model was a good forecaster of future activity while in the MERL set it performed significantly worse than even the average model on all forecasting horizons. This is likely due to a stronger seasonal component to the CSMBB dataset due class schedules. Instead on the MERL dataset there is little seasonal correlation and thus natural variance from a prior season may incorrectly affect current forecasts. This result is similar to that of other papers that use seasonal ARIMA models [**?**]; where in the case of strong seasonal data, results are better for short horizon forecasts, but longer forecasts favor historic averages.

BCF and BCF-TS were both better at a horizon of one time step for all component models (see Table Table 3.3). In the MERL dataset standard BCF was outperformed by SVM and later the average model for all forecast beyond one horizon. However the BCF-TS model showed significant improvement in RMSE scores for all forecasting horizons unto 60 minutes. For horizons of 10 time steps and greater, the average model is about as good as the BCF-TS approach.

Table Table 4.1 shows the run time in seconds of each forecasting algorithm at a given forecasting horizon. The times are for forecasting the entire test set on the MERL dataset for a single sensor, approximately twenty weeks worth of data. In general BCF-TS was slower than any component model, but the times are still such that real-time forecasting is possible.



(a)

(b)

Figure 4.5: Root mean square error of forecasting for each model vs forecasting horizon.

CHAPTER 5

IMPROVED ENSEMBLE

Discuss our improved ensemble method here

5.1 Activity recog

discuss activity recognition and its importance here

5.2 HMM vs TSMG

discuss why we are not using hidden markov models as stated in the proposal

5.3 TSMG derivation

A mixture of Gaussians is a strongly supported stochastic data clustering technique used in activity recognition. Traditionally, a mixture of Gaussians is implemented for either a one dimensional time series (CITE PAPER TO SUPPORT THIS) or for data vectors with no time element. Here we combine the two approaches, creating a mixture of Gaussians for multi-dimensional time series data. While this approach has not yet been implemented, based on the success of mixture of Gaussians in other domains, we expect good results.

The goal of mixture of Gaussians is to find a set of models which will maximize the log likelihood of the parameters of some models to the dataset. Given dataset $\{x^{(i)}\}$ we maximize

$$\ell(\theta) = \sum_{i=1}^{\mathbf{M}} log\{p(x^{(i)}|\theta)\} \tag{5.1}$$

where **M** is the total number of time series instances.

The expectation maximization (EM) algorithm is commonly used to maximize dataset likelihood. To use this algorithm we need to define a set of variables

$$w_k^{(i)} = p(z = k|x^{(i)}) \tag{5.2}$$

where $\mathbf{K}$ is the total number of Gaussians to train and k is an index of $\mathbf{K}$.

The general equation for the likelihood of the models is:

$$\ell(\theta|x) = \sum_{i=1}^{\mathbf{M}} \sum_{k=1}^{\mathbf{K}} w_k^{(i)} \log\{\frac{p(x^{(i)}|z=k)p(z=k)}{w_k^{(i)}}\} \tag{5.3}$$

In the traditional mixture of Gaussians algorithm each model is ostensibly a Gaussian. To make this algorithm work with multi-dimensional time series, we define the models instead by

$$p(x^{(i)}|z=k) = \prod_{n=1}^{\mathbf{N}} \mathcal{N}_n(x^{(i)}) \tag{5.4}$$

where $\mathbf{N}$ is the length of each time series instance. Thus our model for each time series is $\mathbf{N}$ independent multivariate Gaussians.

Combining equations 5.3 and 5.4 gives the following log likelihood

$$\ell(\theta|x) = \sum_{i=1}^{\mathbf{M}} \sum_{k=1}^{\mathbf{K}} w_k^{(i)} \{\log \frac{p(z=k)}{w_k} + \sum_{n=1}^{\mathbf{N}} \log \mathcal{N}_n(x^{(i)})\} \tag{5.5}$$

E-Step The E-step hardly changes from the traditional EM mixture of Gaussians algorithm. We simply need to calculate

$$w_k^{(i)} = p(z=k|x^{(i)}) \tag{5.6}$$

M-Step For the maximization step, it is assumed that we know the values of $w_k^{(i)}$. Thus, we need to maximize equation 5.5 with respect to μ, Σ, and θ. The results of these maximizations are given below:

$$\theta_k = \frac{1}{\mathbf{M}} \sum_{i=1}^{\mathbf{M}} w_k^{(i)} \tag{5.7}$$

$$\mu_{k,n} = \frac{\sum_{i=1}^{\mathbf{M}} w_k^{(i)} x_n^{(i)}}{\sum_{i=1}^{\mathbf{M}} w_k^{(i)}} \tag{5.8}$$

$$\Sigma_{k,n} = \frac{\sum_{i=1}^{\mathbf{M}} w_k^{(i)} (x^{(i)} - \mu_{k,n})(x^{(i)} - \mu_{k,n})^{\mathrm{T}}}{\sum_{i=1}^{\mathbf{M}} w_k^{(i)}} \tag{5.9}$$

5.4 Improved Ensemble Derivation

Derive our improved ensemble method here.

CHAPTER 6

CONCLUSION

Discuss our conclusions here

REFERENCES CITED

[1] US DOE. Building Energy Databook, 2010. URL http://buildingsdatabook.eren.doe.gov/.

[2] KMC Controls. Understanding Building Automation and Control Systems, 2013.

[3] KMC Controls. Zone Control with Variable Air Volume Controls (VAV), 2013.

[4] In-Ho Yang and Kwang-Woo Kim. Prediction of the time of room air temperature descending for heating systems in buildings. *Building and Environment*, 39(1):19–29, January 2004. ISSN 0360-1323. doi: http://dx.doi.org/10.1016/j.buildenv.2003.08.003.

[5] Yudong Ma, Francesco Borrelli, and Brandon Hencey. Model predictive control for the operation of building cooling systems. In *America Control Conference*, 2010. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5739562.

[6] US DOT. National Traffic Signal Report Card, 2007. URL http://www.ite.org/reportcard/NTSRCExecSummaryfinal.pdf.

[7] Peter Koonce, Wayne Kittelson, Lee Rodegerdts, Kevin Lee, and Caroline Swartz. The Signal Timing Manual. Technical report, Federal Highway Administration, 2008.

[8] R.J. Hyndman and A.B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688, 2006.