Simultaneous Feature Selection and Parameter Estimation for Hidden Markov Models

---

A Dissertation

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

---

in partial fulfillment

of the requirements for the degree

Doctor of Philosophy

by

Stephen Conway Adams Jr.

December

2015

APPROVAL SHEET

The dissertation

is submitted in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

AUTHOR

The dissertation has been read and approved by the examining committee:

Dr. Peter Beling

Advisor

Dr. Matthew Gerber

Dr. Quanquan Gu

Dr. William Scherer

Dr. Scott Acton

Accepted for the School of Engineering and Applied Science:

Craig H. Benson, Dean, School of Engineering and Applied Science

December

2015

University of Virginia

Doctoral Thesis

Simultaneous Feature Selection and Parameter Estimation for Hidden Markov Models

Author:
Stephen Adams

Advisor:
Dr. Peter Beling

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

in the

Department of Systems and Information Engineering

August 2015

UNIVERSITY OF VIRGINIA

Abstract

Department of Systems and Information Engineering

Doctor of Philosophy

Simultaneous Feature Selection and Parameter Estimation for Hidden Markov Models

by Stephen Adams

Prior knowledge about a system is crucial for accurate modeling. Bayesian parameter estimation theory, specifically the use of informative prior distributions, offers one method for conveying prior knowledge to the learning algorithm that may not be present in a data set. This dissertation primarily focuses on the problem of feature selection for hidden Markov models with respect to the test cost of the individual features. Test costs include the financial cost of acquiring the feature, the difficulty in collecting the feature, the time required to collect the feature, etc. We propose using a feature saliency hidden Markov model (FSHMM) that simultaneously selects features and estimates model parameters. We assume that the number of states is known, and use the expectation maximization algorithm for parameter estimation. Informative prior distributions are used to convey the test cost to the learning algorithm. Three formulations are derived for the FSHMM: a maximum likelihood formulation using no priors, and two maximum a posteriori (MAP) formulations using informative priors. These are compared to an existing formulation that uses non-informative priors and variational Bayesian methods for parameter estimation. The proposed formulations are extended to numerous conditional feature distributions, including the gamma distribution and the Poisson distribution, and a semi-Markov model. The FSHMM is tested using synthetic data, a tool-wear data set, an activity-recognition data set, and an event-detection data set. We find that the MAP formulation using a truncated exponential distribution on the feature saliencies generally outperforms the other FSHMM formulations and conventional feature selection techniques in terms of predictive performance and selecting a feature subset.

Acknowledgements

First, I would like to thank my adviser, Dr. Peter Beling, for his support throughout my PhD program and research. His guidance was invaluable and I could not have completed this dissertation without his motivation and knowledge. I look forward to working with him in the future and continuing our professional relationship.

I would also like to thank my dissertation defense committee: Dr. Matthew Gerber, Dr. William Scherer, Dr. Quanquan Gu, and Dr. Scott Acton. Their feedback has been crucial throughout this process. In addition, I would like to thank Dr. James Fitz-Gerald and my original adviser Dr. Randy Cogill for their advice at the beginning of my program.

I would like to thank Barbara Nordin for help and feedback editing the first chapter of this dissertation.

I would like to thank Dr. Ed Hall and the entire University of Virginia Advanced Computing Service and Engagement group for their help implementing the models in Chapter 5 on the UVA high performance cluster.

For the entirety of my PhD program, I have been supported by research funds from the Commonwealth Center for Advanced Manufacturing. I thank all the members of their organization and all the member companies.

And, of course, I could not have completed this without the love and support of my family. I would especially like to thank my girlfriend, Gray Ibbeken, without her, none of this would be possible.

Contents

List of Figures

List of Tables

Abbreviations

AIC	**A**kaike **I**nformation **C**riteria
BIC	**B**ayesian **I**nformation **C**riteria
CDF	**C**umulative **D**istribution **F**unction
EDHMM	**E**xplicit **D**uration **H**idden **M**arkov **M**odel
FS	**F**eature **S**election
FSEDHMM	**F**eature **S**aliency **E**xplicit **D**uration **H**idden **M**arkov **M**odel
FSHMM	**F**eature **S**aliency **H**idden **M**arkov **M**odel
GMM	**G**aussian **M**ixture **M**odel
HMM	**H**idden **M**arkov **M**odel
HSMM	**H**idden **S**emi **M**arkov **M**odel
LOOCV	**L**eave-**O**ne-**O**ut **C**ross **V**alidation
ME	**M**aximum **E**nergy
PCA	**P**rincipal **C**omponent **A**nalysis
PDF	**P**robability **D**ensity **F**unction
PMF	**P**robability **M**ass **F**unction
RMS	**R**oot **M**ean **S**quare
RMSE	**R**oot **M**ean **S**quare **E**rror
SBS	**S**equential **B**ackward **S**election
SFS	**S**equential **F**orward **S**election
SLE	**S**um of **L**og **E**nergies

Dedicated to my parents, Nancy and Steve, for all of their love and support.

Chapter 1

Introduction

Machine learning is a collection of techniques designed to detect hidden patterns in data and use this knowledge to predict the outcome of future data. These predictions can be used to make decisions about the system, which will generate the data in the face of uncertainty. One area of machine learning is focused on learning parametric models that explain the behavior of the data. These types of parametric models often make assumptions about how the data interact with an outcome or the distributions generating the data. Different assumptions can lead to wildly different outcomes in terms of model selection or accuracy. Generally, it is best to back up an assumption with facts or details about the system or process being modeled.

Prior knowledge about a system can come from many sources. For instance, the cost of collecting a data stream (financial, computational, or difficulty in acquiring the feature) is not always easy to convey in a data set. Furthermore, some systems have physical restrictions or properties the model must adapt to, and this can also be difficult to capture in collected data. Using these two types of information, which would not be included if only collected data were considered, will lead to models that more closely represent the system. Bayesian statistics, and specifically the prior distributions used in Bayesian analysis, offer one method of incorporating prior knowledge into system modeling.

In Bayesian statistics, all parameters are treated as random variables. Let θ represent the set of model parameters. Parameter estimation is performed by estimating the posterior distribution $P(\theta|\mathcal{X})$ of the parameters given data $\mathcal{X}$. The posterior can be found through Bayes' rule

$$P(\theta|\mathcal{X}) = \frac{P(\mathcal{X}|\theta)P(\theta)}{P(\mathcal{X})}, \tag{1.1}$$

where $P(\mathcal{X}|\theta)$ is the likelihood of the data, given the parameters; $P(\theta)$ is the prior distribution on the parameters; and $P(\mathcal{X})$ is the marginal distribution of the data. Often, the posterior need only be estimated proportionally, as the marginal of the data is a fixed quantity

$$P(\theta|\mathcal{X}) \propto P(\mathcal{X}|\theta)P(\theta). \tag{1.2}$$

In Bayesian estimation, the prior distribution on the model parameters is typically non-informative, meaning that it assigns equal weight to all possibilities. Priors are chosen by the researchers; therefore, the researcher can influence the estimation by the selection of a prior distribution or the parameters for that distributions. Bayesians who promote non-informative priors wish for the data to be the only factor driving estimation, and prevent any bias or influence being injected into the estimation by the practitioner. Furthermore, non-informative priors allow for easy selection of prior distributions and hyperparameters, as these choices have little effect on the posterior.

On the other hand, informative priors can be used to convey information to the estimation process that is not readily available in the collected data. Informative priors are often avoided for two reasons in Bayesian analysis. The first is the previously discussed researcher influence that can affect the analysis. The second is the lack of consistent methodology for constructing informative priors that is widely accepted over types of models and fields of study. It is reasonable to assume that two different researchers could construct very different prior distributions, given the same prior data or set of assumptions.

In this dissertation, we argue that the use of informative priors when modeling systems is crucial for two reasons. First, knowledge about a system that is not present in the collected data can be conveyed to the estimation process through prior distributions. Second, good informative priors can increase model accuracy and other notions about model performance.

Most decisions when modeling a data set are based on prior information. By choosing a class of models that one believes will accurately reflect the data, the

researcher has begun using prior information and, in a sense, has already established a prior distribution. For example, by choosing logistic regression over other classifiers, one has placed a prior with probability 1 on logistic regression and probability 0 on all other classifiers. If there is strong evidence that logistic regression will outperform other classifiers, or logistic regression offers advantages over other classifiers, this strong informative prior is justified. Using the notion that informative priors encompass any type of decision when modeling data, we use prior knowledge and informative priors for three tasks: (1) selecting the type of model, (2) selecting model structure, and (3) parameter estimation.

For case studies, we investigate two well known manufacturing problems: tool wear and activity recognition. These problems are analyzed to showcase the advantages of informative priors in modeling systems. Other data sets, including synthetic data, are also modeled, but not to the extent of either tool wear or activity recognition.

The tool-wear problem is to predict the unknown wear on the tool given data collected from the cutting process such as force or vibration. The activity-recognition problem is to predict the activity a subject is engaged in, given data collected on the subject such as upper-body joint positions. We use hidden Markov models (HMMs) [95] throughout this dissertation. HMMs are widely used to model sequential data and have been applied in numerous fields, such as speech recognition, finance, and video recognition. An HMM is composed of a sequence of unknown or hidden states modeled as a Markov chain and a correlated sequence of observations. In application, the parameters of an HMM are estimated from data. When both the hidden and observed sequences are available, the parameters can be directly calculated from the data. However, the hidden sequence is generally not available when using HMMs. Therefore, the parameters must be estimated using unsupervised learning algorithms such as Baum-Welch [95], Markov chain Monte Carlo methods [12], or variational Bayesian methods [73]. We choose HMMs for these cases studies because of their success in modeling time-series data and the ability to train HMMs using unsupervised learning algorithms. Because of the difficulty in labeling these types of data sets, unsupervised learning algorithms are desirable.

In these case studies, one area of prior knowledge we wish to convey to the model is that input features are associated with some form of cost. A vector of observations collected from several sources can be available at each time period and the cost

of collecting the feature associated with each source can differ significantly. In addition, some features might not be useful to the model. In order to build a parsimonious model, the features that do not contribute to the usefulness of the model can be removed without significantly degrading its accuracy. One approach to feature selection (FS) is to model every possible subset of features and compare the models based on some metric. As the number of features grows, this quickly becomes impractical. FS for HMMs with respect to cost is the primary problem we address in this dissertation.

As a concrete example, consider a model with two features. The financial cost associated with feature 1 is \$500, and the financial cost associated with feature 2 is \$1,000. If a model built using one feature has similar predictive ability as a model built using two features, the model using one feature should be preferred. Further, if the two models built on the single feature have similar predictive ability, then feature 1 should be preferred because of its lower financial cost.

In light of this knowledge about FS with respect to cost, as well as an HMM's ability to model sequential data and be trained using unsupervised learning, we propose a feature-saliency HMM (FSHMM) that simultaneously estimates model parameters and selects features using unsupervised learning. Informative priors are placed on some model parameters to convey test cost to the algorithm. The case studies demonstrate that these informative priors produce models that compare favorably to similar models that use either no priors or non-informative priors.

With respect to cost, numerous challenges are specific to FS for HMMs:

- The hidden state is often not available in the data set and requires an unsupervised learning technique.
- Training HMMs can be computationally expensive, so an embedded technique that performs simultaneous parameter estimation and FS is desirable.
- Conventional FS techniques can perform poorly when applied to HMMs.
- Conventional FS techniques do not consider the test cost of features.

The work presented in this dissertation can be viewed intellectually at multiple levels. In the broadest sense, it investigates the infusion of prior knowledge into HMMs. At the next deeper level, it is a study of informative priors for HMMs. At

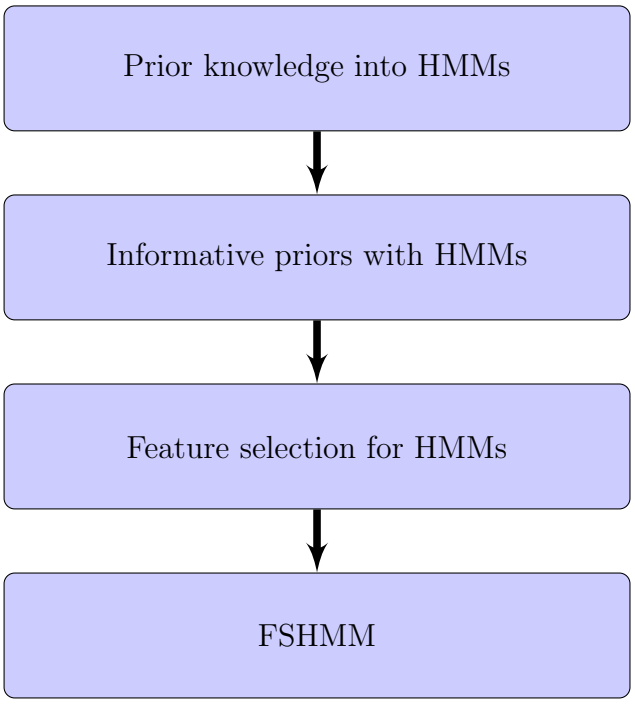

FIGURE 1.1: Intellectual Levels for FSHMM. At the highest level, we investigate incorporating prior knowledge into HMMs. At the lowest level, we study the FSHMM.

an even deeper level, FS with respect to cost for HMMs is considered. At the deepest level, a simultaneous features selection and parameter estimation algorithm is examined. Figure 1.1 depicts this as a flowchart: as one descends the chart, the area of study becomes more focused.

This dissertation is organized as follows. The rest of this chapter consists of a problem statement, the approach chosen to study the problem, a literature review of relevant topics (FS and FS specific to HMMs, tool wear and activity recognition, and informative priors), and an outline of the work's intellectual contributions. Chapter 2 is a summary of background information on HMMs and hidden semi-Markov models (HSMMs). Chapter 3 outlines the FSHMM and presents numerical experiments; the FSHMM is also compared to other feature-saliency formulations and conventional FS techniques. Chapter 4 outlines conditional feature distributions other than the Gaussian, as assumed in Chapter 3: Gaussian mixture models, exponential distributions, gamma distributions, Poisson distributions, and non-parametric discrete distributions. Chapter 5 outlines the feature saliency formulation for a specific form of hidden semi-Markov models. In Chapter 6, we present conclusions and suggest avenues for future research.

1.1 Problem Statement

We investigate the use of informative priors with HMMs. Specifically, we study and develop a method for incorporating the cost associated with collecting individual features into a joint FS and model estimation process. Informative priors are primarily used to deter the selection of features with a higher cost.

1.2 Approach

As mentioned earlier, we use informative priors in three ways. In this section, we expand on and explain them.

- **Model Selection** HMMs have been widely used for modeling tool wear [34, 53, 112, 138] and activity recognition data [39, 48, 120, 125]. Further, tool wear and activity recognition are time-series data with measured features that correlate with a latent or hidden variable. This prior knowledge leads to the first way we use informative priors, as outlined in the introduction: the selection of HMMs to model the data.

- **Model Structure** Tool wear is non-decreasing. In terms of the HMM, this means that once the wear enters a state, it cannot transition to a lower state. This physical attribute of tool wear can be incorporated into an HMM by restricting the Markov chain to be left-to-right (LTR). An LTR Markov chain can only self-transition and transition to the next highest state. All other state transitions have a probability of 0. Decisions similar to choosing HMMs or restricting the Markov chain to be LTR can be considered using informative priors to convey knowledge to the model structure. This is the second way we use informative priors: to select model structure.

- **Parameter Estimation** Test cost plays an important role in both of these data sets. In the tool-wear data set, two types of sensors collect data. The difference in the financial cost of these types of sensors can be significant. It is assumed that the force sensor costs twice as much as the vibration sensor. This assumption was made after reviewing the price of several commercial sensors and we believe it adequately reflects the real world. The features in the activity-recognition data set have a different notion about cost. The

device used to collect data (Microsoft Kinect) samples every 30^{th} of a second. Three coordinates for each of the upper-body joints are recorded. Features can be calculated from the joint locations – e.g. the distance between the hands – but only joint positions are used as features in this dissertation. The size of this type of data can grow rapidly. For the activity-recognition problem, we associate cost with a growth in data. Irrelevant features add to computation time for the model and degrade its accuracy. Each feature is assumed to have the same collection cost, and the smallest feature subset is desired.

The third way we use informative priors is to influence parameter estimates. Most notably, informative priors are used to convey the two previously outlined notions about cost to the FS algorithm by penalizing more costly features or larger feature subsets. The hyperparameters for the informative prior can be increased to discourage the inclusion of features. A higher value indicates that the feature must contribute more useful information to the model. For the tool-wear data, the hyperparameters for features from the more costly sensor (force) are double the hyperparameters for the features calculated from the less costly sensor (vibration). This indicates that one feature costs twice as much as another. When modeling the activity-recognition data set, the hyperparameters are set equal for each feature, but the value is increased to discourage larger feature sets. The hyperparameters for the informative priors were selected based on intuition and not formal methodology. A methodology for choosing hyperparameters (and selecting informative prior distrubtions) is a matter for future work and will be discussed in the future work section.

To a lesser degree, the initial values for the expectation-maximization (EM) algorithm are used as hyperparameters for some model parameters. The initial values are either chosen by intuition about the process and the true values of the parameters or calculated from a supervised initialization set.

1.3 Feature Selection

Feature selection is the process of reducing the set of collected features to a subset of relevant features. FS is also called variable selection, attribute selection,

or feature subset selection. FS speeds the learning process, improves model interpretation, reduces the risk of overfitting, and alleviates the effects of the curse of dimensionality. When the feature set is small, an exhaustive search can be performed. As the number of features grows, this method becomes impractical, because the number of possible feature subsets grows exponentially.

Feature extraction is a separate problem from FS. FS identifies relevant features from a candidate set of features, while feature extraction calculates new features from a given set. FS does not alter the original features, while extraction generates new features. Dimensionality reduction is usually applied to extracted features. Principal component analysis (PCA) [80] is a feature-extraction method. Dimensionality reduction is performed by selecting the first m principal components. This method differs from FS, however, because all input features must still be collected in order to extract the new principal components. Conversely, after FS has been performed, the irrelevant features no longer need to be collected. Further, the selected subset of features can give insight into the process and be interpreted by a domain expert. However, methods such as PCA reduce the noise in the extracted features, and can provide more discriminating features than the original raw features. Independent component analysis (ICA) [80] is another form of feature extraction similar to PCA.

John et al. [52] and Kohavi and John [56] outline four definitions for relevant features that were current in the literature in the mid 1990's. Using a correlated XOR problem, they show that different definitions can lead to the selection of different feature subsets, and, after arguing that definitions of strong and weak relevance are necessary, they provide such. However, in [56], John and Kohavi go on to show, using an example, that relevant features are not always included in the optimal feature set and irrelevant features are not always excluded from the optimal feature set. Blum and Langley [9] provide definitions of feature relevance that include a target concept and a measure of complexity of the selected feature subset. Blum and Langley's final definition of relevance, which first appeared in [16], introduces the idea of relevance with respect to a specific learning algorithm.

Guyon and Elisseeff [44] lay out several steps for general FS. These are in the form of questions, and the answers lead to specific actions to be taken or a particular FS method to be used. For example, their fifth question is, "Do you need to assess features individually?" If the answer is yes, a procedure that ranks features should be used.

FS methods can be divided into three groups: filters, wrappers and embedded. Filters assess feature relevance by investigating the feature's properties. They address the problems of selecting features and building models independently. Wrappers assess feature relevance with regard to a specific learning algorithm. In most cases, a model is built with respect to a subset of features and the model's performance evaluated based on specified criteria. Wrappers then move through the subset space evaluating feature subsets with regard to the evaluation function. Embedded methods simultaneously select features and construct models.

1.3.1 General Feature-selection Techniques

Filters treat FS as a preprocessing step and select features with no regard to the model: They only consider the properties of the collected features and how they distinguish themselves from one another or how they relate to a target class label. These methods are generally fast in terms of computation, but can result in feature subsets that do not yield satisfactory predictive accuracy. Certain types of filters can be applied to unsupervised data. In addition, filters usually scale well to the number of features.

The simplest filtering technique is to select features based on their correlation with the class or continuous response variable. More complex filters include the FOCUS [3] and Relief [55] algorithms. Extensions of Relief [57, 100] can be applied to multi-class problems and unsupervised data. Feature-similarity FS[77] is an unsupervised filter that groups the k closest features. A single feature from each group is chosen to represent the group in the reduced feature subset. The idea behind this method is that features that are similar by some metric are redundant and can be removed with insignificant change in prediction accuracy. Some filters rank or assign weights to features. Correlation-based methods [131] rank features based on their association with a class label. Forman [38] compares several metrics for ranking features. The evaluation is specific to text classification, so results may not generalize to all classification problems. The author concludes that bi-normal separation metrics outperforms other filters on the chosen data sets.

Wrappers test feature subsets, given a model and an evaluation function. These methods are more computationally expensive, but take the model into account and often yield better feature subsets than filters. However, wrappers generally require the use of supervised data for the evaluation function. In a wrapper approach,

data are typically divided into three groups: training, evaluation, and testing. A model using a subset of the candidate features is trained on the training set, then evaluated using the evaluation set. The feature subset is then augmented in some fashion and the process repeated. The feature subset that optimizes the evaluation function is chosen as the final feature subset and tested on the withheld testing set. When data are scarce, the evaluation set can be eliminated by evaluating the feature subset on the training data. However, this can cause a poor generalization error and increase the likelihood of overfitting to the training data. Cross validation can be used in the case of small data sets.

Sequential forward search (SFS) and sequential backward search (SBS) are two types of exploration algorithms [52, 56]. These are considered greedy algorithms, as opposed to an exhaustive search of the feature subsets. Sequential search methods are often referred to as hill-climbing strategies, because they look for improvement in the evaluation function. However, a non-exhaustive search cannot guarantee the optimal feature subset [25]. Another exploration method, the stepwise search, combines SFS and SBS so that at each step in the algorithm, a feature can either be added or removed. When compared to unidirectional methods and filters, stepwise search algorithms outperform unidirectional search and some filtering techniques [15].

Kohavi and John [56] propose a best-first-search feature-exploration technique with compound operators. Compound operators add or remove groups of features, as opposed to adding or removing a single feature in SFS and SBS. The branch and bound algorithm [81, 105] can yield an optimal feature subset, but requires a monotonic evaluation function that is generally not practical. Floating-search methods [93, 94] add and remove different numbers of features to avoid the nested-feature problem.

When using wrappers, the choice of an evaluation function greatly affects the outcome of the method. Dash, Liu and Motoda [28] compare a consistency measure to distance measures, information measures, and dependence measures. The authors argue for a consistency-measure evaluation function, because it is monotonic and lacks search bias. Obviously, the type of evaluation function is based on the available data. Functions that require the class label, such as consistency or accuracy, cannot be implemented in unsupervised learning.

Embedded techniques simultaneously select features and construct models. Therefore, these techniques have the wrapper's advantage of selecting feature subsets with respect to a specific learning algorithm, and the filter's advantage of being more computationally efficient than wrappers. Classification and regression trees (CART) [80] are one example of an embedded FS method. CART recursively divides the feature space to create a classification model. The algorithm only includes features that improve the impurity measure, thereby essentially performing FS and constructing the classifier simultaneously. Daelemans et al. [26] show that jointly selecting features and optimizing model parameters in a natural-language-processing context outperforms optimizing the two processes separately.

1.3.2 Feature Selection for Gaussian Mixture Models

Most of the previously described work on FS requires the use of supervised data – i.e., each observation has a corresponding class or response variable. Data are rarely supervsied when working with HMMs or Gaussian mixture models (GMMs), and require the use of unsupervised FS techniques. Most work with unsupervised FS employs GMMs as the clustering algorithm, but FS methods have been developed specifically for the K-means algorithm [11, 54] or to be independent of a classifier [27].

Several unsupervised FS methods for GMMs have been investigated. Any filter that does not require supervised data can be applied to GMMs; for example, the feature-similarity technique in [77]. When implementing a wrapper for supervised learning, some form of accuracy is used for the evaluation function. This is not possible for unsupervised data, so new evaluation functions have been proposed. In [31–33], the authors use scatter separability and maximum likelihood. However, these evaluation functions are biased depending on the dimensionality of the feature set. To implement them, steps must be taken to remove the bias; the authors suggest a heuristic normalizing scheme. Raftery and Dean [97] use Bayesian information criterion (BIC) with a greedy search. BIC with a backward stepwise search is implemented in [72].

Penalized likelihoods are used in [91] and [118] to shrink estimates of state-dependent means. Features with estimated means that have the same value can be eliminated, since they do not contribute to clustering, and EM is used for the estimation. Carbonetto et al. [14] use a Bayesian feature-weighting technique

that shrinks parameters for irrelevant features to a common value through the use of prior distributions. Because these methods simultaneously select features and estimate model parameters, they are considered embedded FS techniques.

Feature saliency [62], which recasts the FS problem as a parameter-estimation problem, was first introduced for GMMs. New parameters, called feature saliencies, are added to the conditional distribution of the GMM. The emission probability consists of mixture-dependent and mixture-independent distributions, and feature saliencies represent the probability of belonging to the mixture-dependent distribution. The saliencies can be interpreted as the probability that a feature is relevant. The feature-saliency selection method is considered to be an embedded FS technique, because it simultaneously constructs a model and performs selection. However, feature saliencies can be used to rank feature relevance, as is the case with many filtering techniques.

Figueiredo, Jain, and Law [36] use a precursor to feature saliency, the minimum-description length (MDL) criterion, and the idea of component-dependent and component-independent distributions to select features. Vannucci and Stingo [113] outline a Bayesian approach to variable selection that is similar to feature saliency: A binary random variable is used to represent belonging to either a component-dependent and component-independent distribution, and priors can be used to convey biological information to the system. The authors outline both supervised and unsupervised models, and suggest that Gibbs sampling be used to estimate model parameters. However, only the supervised model is tested and evaluated on data.

In their initial studies of feature saliency and GMMs, Law, Jain, and Figueiredo [61] and Law, Figueiredo, and Jain [62] used the EM algorithm and the minimum-message length (MML) criterion to estimate model parameters and the number of clusters. The MML penalty, which is used to aid in model selection, encourages feature-saliency estimates for irrelevant features to go to zero and helps estimate the number of mixtures by forcing the probability of sparsely occupied components to zero. In [62], the authors propose a post-processing step in which feature saliencies are optimized to discriminate between components, after which other parameters of the model are estimated using EM. They also note that a limitation to this method is the assumption that all the features are independent.

Bayesian methodologies have been used to solve for model parameters in the form of variational Bayes (VB) [24, 65, 111] and expectation propagation (EP) [19]. Valente and Wellekens [111] assign a Dirichlet prior to the feature saliencies and compare the EM method in [61, 62] with a variational Bayesian approach. They show that the VB method outperforms EM on a speech-recognition data by yielding better recognition rates, converging faster, and selecting a smaller feature subset. Constantinopoulus, Titsias, and Likas [24] use different priors are used for the component-dependent distribution parameter. No priors are placed on the mixture probability, the feature saliencies or the component-independent distribution parameters; essentially, they are considered model parameters and not random variables, as was the case in [111]. Constantinopoulus et al. do not compare their VB formulation to Valente and Wellekens' formulation but do compare it to the EM method. They demonstrate that VB outperforms EM, but selects more components. Localized feature saliency is presented in [65], in which the saliencies are dependent on the cluster assignment. The authors use the same priors as [24], and show that their method outperforms the global VB formulation in terms of accuracy and the number of components selected on several data sets. Chang, Dasgupta and Carin [19] apply the EP approach to a synthetic data set and compare it to the EM method. They demonstrate that EP outperforms EM when the number of clusters is unknown but EM performs better when the number of clusters is correctly determined. All of these studies, simultaneously solve for model parameters, perform FS and estimate the number of mixture components.

1.3.3 Feature Selection for Hidden Markov Models

While numerous studies have investigated FS in general and GMMs specifically, research on FS with HMMs is lacking. In most applications, features are preselected based on domain knowledge and the FS procedure is completely omitted [78, 126]. Few attempts have been made to reduce the likelihood calculation in the HMM [10, 63, 64], but these methods are not truly FS techniques, as all data streams must be collected. PCA has been used to reduce the feature space and extract features for HMMs [6, 59, 68]. ICA, which is similar to PCA, has also been used with HMMs [122, 136]. Other methods for transforming the original feature set for use with HMMs has also been studied. Yin et al. [130] uses segmental boosting to create a new feature space. As previously discussed, transformation methods such as PCA and ICA reduce the number of features in the model, but

do not eliminate data streams. They are dimensionality-reduction or feature-extraction techniques and should not be considered FS, as all data streams must be collected.

Nouza [84] compares (SFS), discriminative feature analysis (DFA) and PCA. SFS and DFA outperform PCA, but require more computation and supervised data. Lv and Nevatia [70] use boosting to perform FS; for each feature and each class, a single HMM is trained. The AdaBoost algorithm is used to learn weights for each feature by increasing the weight for misclassified observations, which requires supervised data. Ji and Carin [51] pose FS for HMMs as a partially observable Markov decision process (POMDP), in which the action is which feature to query. FS is performed by finding the optimal policy for the model. This method has the advantage of including cost in the FS process. However, the authors cite numerous drawbacks, including significant computation. Further, in the experiments, the data are not a time series, and one of the data sets from the GMM saliency formulation [62] is used.

Zhu, He and Leung [137] use a VB method to jointly estimate model parameters and select features for HMMs. This method does not require the number of states – or the number of mixtures if a GMM is used for the emission distribution – to be known a priori. This lack of information increases computation time, increases model complexity, and, in some cases, decreases parameter-estimation accuracy. When using the mean field assumption, VB can underestimate the variance for the approximate distribution [23]. Chatzis and Kosmopoulos [21] demonstrate that in the presence of outliers, VB gives poor estimates for model parameters and the number of states when using an HMM with Gaussian emissions. When predicting a state sequence given the data, it is more efficient to use point estimates and the Viterbi algorithm than to calculate the distribution of all possible state sequences. Therefore, knowing the approximate posterior distributions is generally not necessary for prediction. Similar to some of the VB methods applied to GMMs, this formulation assumes that the state-independent model parameters and the feature saliencies are not random variables.

1.3.4 Feature Selection with Cost

When the financial cost of sensors varies significantly, this information should be included in the FS process. Also, some data might be difficult or time consuming

to collect. These types of costs (financial, time of collection, difficulty, etc.) are generally referred to as "test cost" [67, 76, 86, 127]. Learning algorithms that incorporate cost are referred to as "cost-sensitive learning algorithms". However, these algorithms take a different approach to the problem than that taken by classical FS techniques. Cost-sensitive learning algorithms assume that each measurement of an observation is associated with a cost. The classifier must decide whether the measurement or feature is needed in each instance given its cost. The classic example is medical diagnosis. When a patient presents with symptoms, which test should the doctor order to achieve the best diagnosis, given that the tests have varying cost? Can the doctor make a diagnosis once the information from the first test is received, or are more tests required?

FS techniques that incorporates test cost have been widely studied [40, 47, 74, 75, 90, 103, 135]. However, these methods balance test cost with misclassification cost and primarily focus on decision systems [74, 75, 135], decision trees [40, 103] and K-nearest neighbor [47]. Because a misclassification measure was used in the FS process, these methods require labeled data.

As previously discussed, Ji and Carin [51] formulate the FS with cost problem as a POMDP in which each action is choosing a feature to sample and the hidden states are mixture components. The authors note several limitations to their work, including significant computational requirements and the difficulty of handling continuous features.

Paclik et. al. [90] investigate FS techniques that incorporate cost when the features are naturally grouped together – for example, all data streams from a particular sensor. If one would like to receive the information from one feature from the sensor, all the other features are essentially free. The authors propose a grouped-wise forward selection and a group-wise nested forward selection method.

Smits and Annoni [104] propose an FS technique with a budget for the total cost of features. Their notion of cost is the sum of computational cost, the cost of confusion error, and the cost of selecting the feature. Features are only included in the reduced feature set if the cost of adding the feature keeps the total cost below the budget designated by the user. A forward selection procedure is implemented with a divergence evaluation function.

Yao et al. [129] break attribute selection into two primary categories. The first is internal information and is essentially FS based on the data; the second is external

information. The external method allows users to assign preferences to all features, both quantitative and qualitative, and a feature subset is selected based on the preference relationships. External information can incorporate any type of test cost but requires that the user be able to supply preference information.

1.4 Case Studies

In this section, we review the existing literature on tool wear and activity recognition, and focus on works that incorporates HMMs.

1.4.1 Tool Wear

As cutting tools are used, they gradually begin to fail and need replacement. Tool wear can be broken into two primary types. Flank wear occurs on the cutting edge that is in contact with the work piece and is characterized by the removal of the sharp edge. Crater wear occurs on the rake face and consists of small indentations or craters. Because this dissertation deals only with flank wear, crater wear will not be considered in the literature review or the analysis.

Tool wear leads to increased cutting force and temperature, poor surface finish, and tool breakage. Poor surface finish can necessitate refinishing – which consumes time and other resources – or renders the product unusable. Tool breakage can be catastrophic for the work piece and the machine. In most cases, a tool is replaced after a specified number of cuts. This policy is sub-optimal since the current state of the tool is not considered. A tool that is not worn but is still changed wastes man-hours and machine time. A tool that is worn and not changed jeopardizes the quality of the product. Further, significant randomness is inherent to cutting processes, so it is difficult to determine the optimal number of cuts after which a tool must be changed. Changes in cutting conditions (cutting parameters and type of material) will also change the rate at which a tool wears. Manufacturing processes of the future will require a system that accurately predicts when the tool will be sufficiently worn to warrant changing.

Measuring tool wear can be difficult and time consuming. Halting the cutting process to check the tool's condition is another suboptimal policy that wastes

man-hours and machine time. As previously stated, tool wear causes increased cutting force and temperature. Significant research has been devoted to tool-wear prediction using data collected from the machine. We will focus here on the use of HMMs in tool-wear prediction; for in-depth reviews that consider multiple algorithms, types of sensors, and machining processes, see [29] and [99].

HMMs have been used to predict tool wear in several types of machining processes, including turning [53, 102, 117], drilling [17, 18, 34, 35, 89] and milling [5, 37, 112, 138]. Scheffer, Engelbrecht, and Heyns [102] compare HMMs to neural networks for tool-wear prediction in turning and conclude that each method has advantages, and neither is clearly superior in terms of predictive ability. Neural networks allow for the continuous prediction of wear, while the HMM formulation the authors test requires the entire cut sequence. HMMs are less complex than neural networks and easier to implement. Most of these studies collect data using vibration or acceleration sensors [5, 17, 18, 37, 89, 112, 117], but some collect force signals [34, 35, 102, 138], acoustic emissions [115], or a multiple types of data [53].

Standard HMMs were used in most previous studies. However, in [17] and [18], the authors use more complex HMMs. In the former study, multi-rate HMMs, which can handle inputs on different time scales, are trained. In the latter study, the authors build on their previous work by implementing multi-rate coupled HMMs, which can handle features on different time scales and model multiple Markov chains.

The type and amount of wear data available greatly influences the way HMMs are trained and tested. If every cut is labeled with a wear classification, an HMM can be trained for each class [5, 17, 18, 34, 35, 37, 53, 112, 117, 138]. For example, in a binary classification problem comparing worn and not-worn, only data labeled as worn are used for training the worn HMM. The same is true for the not-worn HMM. For prediction, the label of the HMM with highest likelihood is assigned to unknown data. While this method is widely used, there as some drawbacks. First, supervised data are required. Second, the HMM states lack meaning. Third, the number of HMMs that must be trained increases with the number of classes.

Scheffer, Engelbrecht, and Heyns [102] and Varma and Baras [115] train a single HMM with states representing the wear. This allows the HMMs to be trained with no wear labels. Further, the transition matrix of the Markov chain can be set to reflect the non-decreasing nature of tool wear over time.

When using HMMs, features can either be continuous or discrete. In most studies, the features are modeled as continuous random variables, but Kang et al. [53] and Wang, Mehrabi and Kannatey-Asibu [117] convert the collected data to discrete features. There is not much difference in the modeling or implementation of HMMs with continuous or discrete features. When discrete features are used for tool-wear prediction, conversion from continuous collected data to discrete features requires this extra step.

1.4.2 Activity Recognition

Human activity recognition is the process of identifying and labeling strings of observations collected on a single person or a group with an activity, task or goal. This is of interest in numerous domains, including surveillance, gaming, medicine, and video retrieval. Similar areas of research include image classification (identifying the contents of a single image, as opposed to labeling a series of images), pose recognition (identifying the position of a human body), and object recognition (identifying the presence or position of an object).

As with tool wear, the literature on human activity recognition is extensive. One way to classify the research is by the device used to collect the observation data. Each has its advantages and limitations. For instance, video data, which is collected using a video camera, is easy to collect and is abundant in data sets and on the web. However, analyzing the data - and specifically extracting features from this type of data - can be difficult. Poppe [92] and Turaga et al. [110] survey the literature and techniques for video-based human activity recognition, and Chaquet, Carmona and Fernández-Caballero [20] review the available video data sets. Wearable or on-body sensors are any type of sensor the subjects has on his or her person. These include motion-capture systems, wearable accelerometers and mobile phones. Wearable sensors give accurate information about the movements of the human body, but can be intrusive, and inclusion of the sensor might change the activity. Lara and Labrador [60] survey the human activity-recognition literature on wearable sensors. The newest type of sensors for activity recognition is the depth camera such as the Microsoft Kinect. These sensors can provide detailed information on the human body and are not intrusive. However, research using depth cameras is not as plentiful as the research on video- or wearable-sensor-based systems, but is rapidly increasing. Han et al. [45] survey all research involving the

Kinect and go beyond human-activity recognition to include object tracking and pose estimation.

Due to the breadth of the literature on activity recognition, we will only focus on those studies that relate to this dissertation, specifically, activity recognition using the Kinect, HMMs used for activity recognition, and unsupervised learning algorithms implemented for activity recognition.

While the Kinect was originally developed for gaming platforms, it has become a popular research tool; several studies have focused on its use in activity recognition [85, 88, 107, 108, 116, 125, 128, 134]. Many of these are strikingly similar.

First, they all use data collected in a laboratory environment. Several [88, 116, 125, 128] evaluate their proposed methods on publicly available action-recognition data sets, such as MSR-Action3D and MSRDailyActivity3D. While using these data sets allows for easy comparison of methods, data are collected by simply having subjects repeat specific, predefined, scripted actions several times. There is little or no occlusion of body parts, interaction with other subjects or objects, or unknown activities. Further, the data are generally segmented, so that each clip or sequence of observations only contains a single activity. Data collected in a real-world environment would not be segmented. Also, it would contain transitions between actions, unknown activities, possible interaction with other subjects or objects, occluded body parts, etc. In some cases, the researchers attempt to recreate a real-world setting by collecting their own data [85, 88, 107, 108, 125, 134]. However, these collected data sets have limitations similar to the publicly available sets previously discussed.

Second, the only study using unsupervised learning was that of Zhang and Parker [134], who use a Latent Dirichlet Allocation to model activity. However, their method includes a highly customized feature that uses both the depth and intensity data from the Kinect. All other methods cited require some amount of supervised data. For the supervised methods, HMMs or variants on the standard HMM are used as classifiers [85, 107, 108, 125]. Support vector machines [88, 116] and nearest-neighbor methods [128] are also used to classify the unknown activity.

The primary difference between these is the type of feature extracted from the depth camera and used as inputs to the classifier. Wang et al. [116] propose mining actionlet ensembles. Xia, Chen, and Aggarwal [125] propose histograms of 3D joints while Oreifej and Liu [88] propose histograms of oriented 4D normals

as features, and Yang and Tian [128] develop a feature called eigenjoints. In each of these, the newly proposed feature is the primary contribution. There is no consensus as to the best feature set used for activity recognition with a depth sensor, and FS is not performed.

HMMs and their variants have been routinely employed for activity recognition using the Kinect and other sensors. In the Kinect domain, Nowozin and Shotton [85] propose a variant of an HMM called a "firing hidden Markov model". They use action points to anchor the model and force actions to pass through specific states. Sung et al. [107, 108] use maximum entropy Markov models. Xia, Chen and Aggarwal [125] train an individual HMM for each action.

Behera, Cohn, and Hogg [7], use HMMs in conjunction with probabilistic Latent Semantic Analysis for workflow activity monitoring on data collected using a motion-capture system. Wang, Huang and Tan [120] train a different HMM for each activity for video data.

Some variants of HMMs, such as coupled HMMs [13], switching HMMs [30] and layered HMMs [87], were primarily developed to address problems associated with activity recognition. Variants of HMMs developed in other domains have been applied to activity recognition, such as the variable length HMM [39] and the hierarchical HMM [82].

Sensors used for collecting activity-recognition data can collect several data points per second. For example, the Kinect collects a sample every 30^{th} of a second. This results in data sets with a large number of observations. Labeling each observation with an activity becomes significantly more time consuming and difficult as the data sets grow. Further, accurate labeling of individual observations can be difficult. Labeling is easy for segmented data sets, in which single activities are performed for each sequence. However, data sets collected from real-world applications will not be nicely segmented. Pinpointing the exact transition between two activities is no easy task, and could be interpreted differently when multiple researchers are labeling the data set. For these reasons, unsupervised learning techniques for activity recognition must be developed and evaluated.

Wyatt, Philipose and Choudhury [124] mine web pages to establish links between object use and activity. The subjects wear radio-frequency identification bracelets and interact with tagged objects, and HMMs are used as the classifier. While this method could be useful for controlled environments, in which subjects interact with

specified objects, it requires tagging objects and the use of an object to identify an activity. Krause et al. [58] also use a wearable sensor and unsupervised learning; the subjects wear a SenseWear armband, and the researchers use the k-means clustering algorithm as an unsupervised classifier.

Niebles, Wang, and Fei-Fei [83] outline an unsupervised learning approach for activity recognition on video in which they use probabilistic Latent Semantic Analysis and Latent Dirichlet Allocation for the unsupervised classifiers. Similarly Wang, Ma, and Grimson [119] identify activities in large crowds using surveillance and traffic cameras. For unsupervised classifiers, the authors propose three models: a Latent Dirichlet Allocation mixture model, a Hierarchical Dirichlet Process mixture model, and a Dual Hierarchical Dirichlet Process model.

Semi-supervised learning combines unsupervised data with supervised data. Generally, the amount of supervised data is significantly smaller than the amount of unsupervised data. This allows researchers to gain some of the advantages of supervised techniques without consuming significant man-hours labeling the entire data set. Further, specific portions of the data can be targeted for labeling for convenience or to increase performance. Guan et al. [42] use co-training (training multiple classifiers then labeling the unlabeled data). Stikic, Van Laerhoven, and Schiele [106] test co-training and self-training as well as active learning (determining the most important unlabeled samples for labeling). Mahdaviani and Choudhury [71] train semi-supervised conditional random fields for activity recognition.

1.5 Informative Priors

The use of informative versus non-informative priors has been widely debated. Jaynes [50], who outlines the historical debate up to 1985 and favors the use of informative priors, presents the following example of how prior knowledge is used in every-day inference: A medical diagnostician would not make a diagnosis using only the patient's current condition and ignore his or her medical history. Jaynes concludes by showing how the use of highly informative priors on seasonal parameters for monthly economic data improves forecasting.

Thomas, Witte, and Greenland [109] argue for the use of informative priors in epidemiological studies using hierarchical models in their paper, which is subtitled

"Who's afraid of informative priors?" They state that letting the data "speak for themselves" assumes that the collected data are correct and a good representation of the system being modeled. Further, they state that data per se are not useful without assumptions about the model and the inclusion of prior information, and assert the need to balance data with prior assumptions and information.

Zero-numerator problems involve probabilities, such as a transition matrix for a Markov model, that do not occur in the training data but are possible. During training, their exclusion from the training data causes the probability estimate to be 0, even though the true probability is greater than 0. Winkler, Smith, and Fryback [123] study the role of informative priors in zero-numerator problems. Similarly, Lord and Miranda-Moreno [69] demonstrate that low-sample means for Poisson-gamma models and small sample sizes can affect posterior distributions when vague or non-informative priors are used. Using a non-vague prior decreases the chances of poor parameter estimation. Jang, Lee, and Kim [49] propose a power prior for zero-inflated regression problems.

Informative priors have been used in several types of models. Angelopoulus and Cussens [4] use informative priors in classification and regression trees. Coleman and Block [22] use informative priors in the estimation of parameters for non-linear systems. Mukherjee and Speed [79] study informative priors in network inference.

While previous studies have demonstrated that good informative priors can improve parameter estimation and the predictive ability of models, there is no well-established methodology for converting prior knowledge into prior distributions. Guikema [43] compares five methods for constructing prior distributions for failure probability estimation in reliability analysis: the method of moments, maximum likelihood estimation, maximum entropy estimation, non-informative pre-priors, and confidence/credible interval matching. He demonstrates that the assumptions on the prior greatly affect the posterior. He concludes that if the data used in constructing the prior accurately reflects the data collected later, maximum likelihood estimation gives a posterior with the minimum variance, but maximum entropy estimation is the most robust to differences between the prior data and the observed data. In another study, Yu and Abdel-Aty [132] compare four methods for constructing informative priors: two-stage Bayesian updating, maximum likelihood estimation, method of moments, and expert experience. Each of these has strengths and weaknesses, but the authors conclude that the two-stage Bayesian updating procedure is superior for the data and model used by the authors.

Raina, Ng, and Koller [98] propose the use of transfer learning for constructing informative priors for text classification. Vanpaemel [114] uses hierarchical methods for constructing informative priors, and Washington and Oh [121] outline a methodology for building informative priors from experts' opinions on ranking railroad countermeasures.

1.6 Contributions

In this dissertation, the main contributions are that we:

- Provide the first comprehensive review of FS methods specific to HMMs. In addition, we point out the need for research on FS methods specific to hidden Markov models and outline some requirements for a FS technique when applied to HMMs.
- Derive the FSHMM, which is a novel method for simultaneous FS and parameter estimation for hidden Markov models using the EM algorithm for maximum a priori parameter estimation.
- Fill the gap in the literature regarding feature saliency, HMMs and the EM algorithm by proposing the FSHMM for solving the FS for HMM problem.
- Outline and demonstrate the advantages of the EM algorithm over VB estimation for the FSHMM formulation when the number of hidden states is known a priori.
- Perform the first investigation of the use of informative priors to incorporate test cost of features into the FS process for the FSHMM.
- Demonstrate the advantages of an embedded, unsupervised FS method specifically formulated for HMMs.
- Extend the structure and model formulations of the FSHMM to include emission distributions that are not a simple Gaussian distribution (e.g. GMMs, gamma distributions, Poisson distributions, etc.) and a semi-Markov process for the underlying Markov chain of the FSHMM.

Chapter 2

Hidden Markov Models and Semi-Markov Models

In this chapter, we provide background information on HMMs and HSMMs. It covers model structure, inference, model training, and implementation issues.

2.1 Hidden Markov Models

A hidden Markov model (HMM) is a probabilistic model with joint probability for unobserved states and observed emissions. For this dissertation, a discrete time Markov chain is used for the state, and the state space is assumed to be finite. In a first order model, the current state is only dependent upon the previous state. The emission distribution is dependent upon the current state, and the individual emissions at each time step are considered independent, given the state.

Consider an HMM with continuous emissions and I states. Let $\mathbf{y} = \{y_1, y_2, ..., y_T\}$ be the sequence of observed data where each $y_t \in \mathbb{R}^L$ and the observation for the l-th feature at time t is denoted by y_{lt}. Let $\mathbf{x} = \{x_1, x_2, ..., x_T\}$ be the unobserved state sequence. The transition matrix of the Markov chain associated with this sequence is denoted as $\mathbf{A}$ where the components of this matrix represent the transition probabilities $a_{ij} = P(x_{t+1} = j | x_t = i)$. The initial state distribution is represented by $\boldsymbol{\pi}$. Each element of $\mathbf{A}$ and $\boldsymbol{\pi}$ must be greater than or equal to 0 and the rows of $\mathbf{A}$ and all the elements of π must sum to 1. In terms of these quantities, the complete data likelihood can be written as

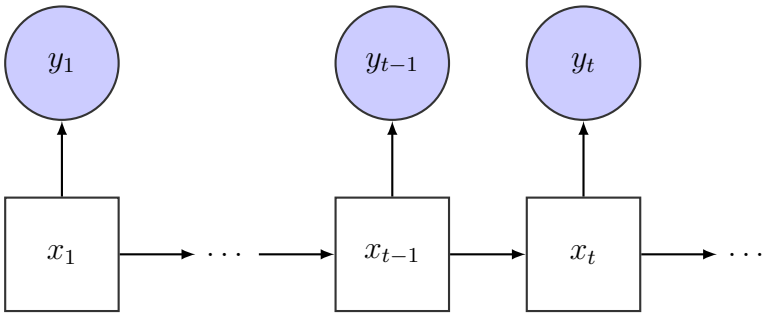

FIGURE 2.1: Graphical model for HMM. Squares are hidden variables and circles are observed variables.

$$P(\mathbf{x}, \mathbf{y}|\Lambda) = \pi_{x_1} f_{x_1}(y_1) \prod_{t=2}^{T} a_{x_{t-1},x_t} f_{x_t}(y_t), \tag{2.1}$$

where Λ is the set of model parameters consisting of the initial distribution, transition probabilities and the emission distribution parameters; and $f_{x_t}(y_t)$ is the emission distribution, given state x_t.

The marginal likelihood of the observation sequence can be found by summing over all possible state sequences

$$\begin{aligned} P(\mathbf{y}|\Lambda) &= \sum_{x_1} \sum_{x_2} \ldots \sum_{x_T} P(\mathbf{x}, \mathbf{y}|\Lambda) \\ &= \sum_{x_1} \sum_{x_2} \ldots \sum_{x_T} \pi_{x_1} f_{x_1}(y_1) \prod_{t=2}^{T} a_{x_{t-1},x_t} f_{x_t}(y_t). \end{aligned} \tag{2.2}$$

A graphical model for an HMM is displayed in Figure 2.1.

2.1.1 The Three Primary Problems for HMMs

Lawrence Rabiner [95, 96] outlined and solved three primary problems when using HMMs:

1. Given a sequence of observations $\mathbf{y}$ and the model parameters Λ, efficiently calculate the marginal likelihood of the observations $P(\mathbf{y}|\Lambda)$.

2. Given a sequence of observations $\mathbf{y}$ and the model parameters Λ, choose a corresponding state sequence $\mathbf{x}$ that is optimal in some sense.

3. Given a sequence of observations $\mathbf{y}$, calculate model parameters Λ that maximizes $P(\mathbf{y}|\Lambda)$.

The first problem, which is generally referred to as evaluation, is solved by calculating the forward variable $\alpha_t(i) = P(y_1, y_2, ..., y_t, x_t = i|\Lambda)$ through induction

$$\begin{aligned} \alpha_1(i) &= \pi_i f_i(y_1), \quad 1 \leq i \leq I, \\ \alpha_{t+1}(j) &= \left[\sum_{i=1}^{I} \alpha_t(i) a_{ij}\right] f_j(y_t), \quad 1 \leq i \leq I, 1 \leq t \leq T-1. \end{aligned} \tag{2.3}$$

The likelihood of the observed sequence can be easily calculated by

$$P(\mathbf{y}|\Lambda) = \sum_{i=1}^{I} \alpha_T(i). \tag{2.4}$$

The backward variable $\beta_t(i) = P(y_{t+1}, y_{t+2}, ..., y_T|x_t = i, \Lambda)$ is calculated in a similar fashion

$$\begin{aligned} \beta_T(i) &= 1, \quad 1 \leq i \leq I, \\ \beta_t(i) &= \sum_{j=1}^{I} a_{ij} f_j(y_{t+1}) \beta_{t+1}(j), \quad 1 \leq i \leq I, t = T-1, T-2, ..., 1. \end{aligned} \tag{2.5}$$

The most common solution to Problem 2, which is generally referred to as inference, is the Viterbi algorithm. First, let

$$\delta_t(i) = \max_{x_1, x_2, ..., x_t} P(x_1, x_2, ..., x_{t-1}, x_t = i, y_1, y_2, ..., y_t|\Lambda). \tag{2.6}$$

The entire optimal sequence can be found through the following steps:

$$\begin{aligned} \delta_1(i) &= \pi f_i(y_1), \quad 1 \leq i \leq I, \\ \Psi_1(i) &= 0. \end{aligned} \tag{2.7}$$

$$\begin{aligned} \delta_t(i) &= \max_{1 \leq i \leq I} \left[\delta_{t-1}(i) a_{ij}\right] f_i(y_t), \quad 1 \leq i \leq I, 2 \leq t \leq T, \\ \Psi_t(i) &= \arg\max_{1 \leq i \leq I} \left[\delta_{t-1}(i) a_{ij}\right], \quad 1 \leq i \leq I, 2 \leq t \leq T. \end{aligned} \tag{2.8}$$

$$\begin{aligned} P^* &= \max_{1 \leq i \leq I} [\delta_T(i)], \\ x_T^* &= \arg\max_{1 \leq i \leq I} [\delta_T(i)], \\ x_t^* &= \Psi_{t+1}(x_{t+1}^*), \quad t = T-1, T-2, ..., 1. \end{aligned} \tag{2.9}$$

The third problem, which is generally referred to as estimation, can be solved using the expectation-maximization (EM) algorithm. The EM algorithm, called the Baum-Welch algorithm when applied to HMMs [8, 95, 96], is used to calculate maximum-likelihood (ML) estimates for the model parameters. Priors can be placed on the parameters to calculate the maximum a posteriori (MAP) estimates [41]. The Baum-Welch algorithm iterates between two steps: the expectation step (E-step) and the maximization step (M-step). The E-step finds the expected value of the complete log-likelihood with respect to the state, given the data and the current model parameters. The M-step maximizes the expectation computed in the E-step to find the next set of model parameters. These two steps are repeated until some stopping criteria is met. The expectation of the complete log-likelihood is designated the $\mathcal{Q}$ function, given by

$$\mathcal{Q}(\Lambda, \Lambda') = \mathbb{E}[\log P(\mathbf{x}, \mathbf{y}|\Lambda)|\mathbf{y}, \Lambda']. \tag{2.10}$$

In Equation 2.10, Λ represents the set of model parameters for the current iteration and Λ' is the set of parameters from the previous iteration. For ML estimation, the $\mathcal{Q}$ function in Equation 2.10 is calculated in the E-step and then maximized

with respect to Λ in the M-step. For MAP estimation, the $\mathcal{Q}$ function is modified by adding terms corresponding to the prior on the model parameters, $G(\Lambda)$,

$$\mathcal{Q}(\Lambda, \Lambda') + \log(G(\Lambda)). \tag{2.11}$$

That is, the $\mathcal{Q}$ function in Equation 2.10 is calculated for the E-step as in the ML algorithm, but the log of $G(\Lambda)$ is added to the $\mathcal{Q}$ function and maximized in the M-step. For both ML and MAP, the quantities in Equations 2.10 and 2.11 are maximized by computing roots of their derivatives with respect to Λ.

For the E-step, the following quantities are calculated

$$\begin{aligned} \gamma_t(i) &= P(x_t = i, |\mathbf{y}, \Lambda) \\ &= \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{I} \alpha_t(i)\beta_t(i)}, \end{aligned} \tag{2.12}$$

and

$$\begin{aligned} \xi_t(i,j) &= P(x_t = i, x_{t+1} = j|\mathbf{y}, \Lambda) \\ &= \frac{\alpha_t(i)a_{ij}f_j(y_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{I}\sum_{j=1}^{I} \alpha_t(i)a_{ij}f_j(y_{t+1})\beta_{t+1}(j)}. \end{aligned} \tag{2.13}$$

When a Gaussian distribution with mean μ and standard deviation σ is used for the state conditional emission distribution, the M-step parameters updates are

$$\hat{\pi}_i = \gamma_1(i), \tag{2.14}$$

$$\begin{aligned} \hat{a}_{ij} &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1}\sum_{j=1}^{I} \xi_t(i,j)} \\ &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \end{aligned} \tag{2.15}$$

$$\hat{\mu}_i = \frac{\sum_{t=1}^{T} \gamma_t(i) y_t}{\sum_{t=1}^{T} \gamma_t(i)}, \tag{2.16}$$

and

$$\hat{\sigma}_i = \sqrt{\frac{\sum_{t=1}^{T} \gamma_t(i)(y_t - \mu_i)^2}{\sum_{t=1}^{T} \gamma_t(i)}}. \tag{2.17}$$

2.1.2 Scaling and Implementation

Joint probabilities, such as the forward and backward variables, rapidly go to zero as the number of total time steps T increases and fall below machine precision. Rabiner [95, 96] proposes scaling these variables as a solution. At each time step t, calculate a scaling factor c where

$$c_t = \frac{1}{\sum_{i=1}^{I} \alpha_t(i)}. \tag{2.18}$$

Denote $\hat{\alpha}_t(i)$ as the scaled forward probability at time t. The scaled forward probability at time t is calculated by

$$\begin{aligned} \hat{\alpha}_t(i) &= c_t \alpha_t(i) \\ &= \frac{\alpha_t(i)}{\sum_{i=1}^{I} \alpha_t(i)}. \end{aligned} \tag{2.19}$$

The same scaling factor used for the forward probability is used to scale the backward probability

$$\hat{\beta}_t(i) = c_t \beta_t(i). \tag{2.20}$$

Scaling does not affect the parameter estimation process because the scaling factors cancel. The scaled forward and backward variables can be substituted directly for $\alpha_t(i)$ and $\beta_t(i)$ in the Baum-Welch algorithm. Furthermore, the scaling factors

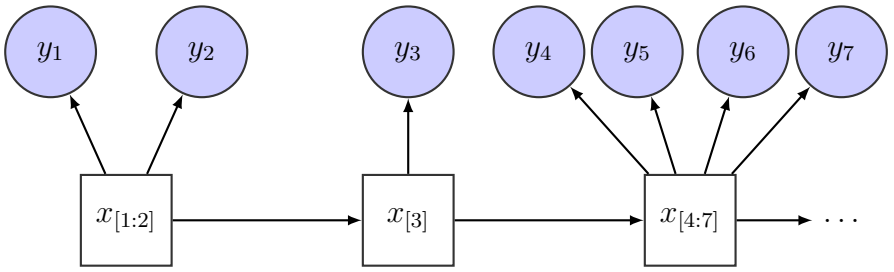

FIGURE 2.2: Graphical model for HSMM. Squares are hidden variables and circles are observed variables. The HSMM differs from the HMM in that the time in each state can be greater than 1 and each state can emit multiple observations.

provide an efficient method of calculating the log-likelihood of the data given the model parameters

$$\log[P(\mathbf{y}|\Lambda)] = \sum_{t=1}^{T} \log c_t. \tag{2.21}$$

2.2 Hidden Semi-Markov Models

In this section, hidden semi-Markov models (HSMMs) are introduced. First, the general HSMM is discussed followed by a discussion of the explicit duration HMM (EDHMM), which is the type of HSMM used in this dissertation. As with the HMMs, model formulation, we outline inference and parameter estimation.

An HSMM [133] is an extension of the standard HMM described in the previous section, where multiple observations are emitted from a single state. The underlying Markov chain follows a semi-Markov process where each state has a duration or sojourn time modeled as a random variable. The duration is the number of time steps in the state before a state transition occurs. The duration takes on discrete values $\mathcal{D} = \{1, 2, ..., D\}$.

Figure 2.2 displays a graphical model of a general HSMM. The model can remain in a specific state for more than one time step, and the hidden state can output more than one observation.

Yu [133] defines the following notation for HSMMs, where a discrete time Markov chain is assumed with a set of hidden states $\mathcal{X} = \{1, ..., I\}$. The following description is quoted directly from [133] with the notation for the state variable x changed to match the notation previously established in this dissertation.

- $x_{t_1:t_2} = i$ - state i that the system stays in during the period from t_1 to t_2. In other words, it means $x_{t_1} = i, x_{t_1+1} = i, ...,$ and $x_{t_2} = i$. Note that the previous state x_{t_1-1} and the next state x_{t_2+1} may or may not be i.
- $x_{[t_1:t_2]} = i$ i state i which starts at time t_1 and ends at t_2 with duration $d = t_2 - t_1 + 1$. This implies that the previous state x_{t_1-1} and the next state x_{t_2+1} must not be i.
- $x_{[t_1:t_2} = i$ - state i that starts at time t_1 and last till t_2, with $x_{[t_1} = i, x_{t_1+1} = i, ..., x_{t_2} = i$, where $x_{[t_1} = i$ means that at t_1 the system switched from some other state to i, i.e., the previous state x_{t_1-1} must not be i. The next state x_{t_2+1} may or may not be i.
- $x_{t_1:t_2]} = i$ - state i that lasts from t_1 to t_2 and ends at t_2 with $x_{t_1} = i, x_{t_1+1} = i, ..., x_{t_2]} = i$, where $x_{t_2]} = i$ means that at time t_2 the state will end and transit to some other state, i.e., the next state x_{t_2+1} must not be i. The previous state x_{t_1-1} may or may not be i.

For the general model [133], the state transition is a pair representing the new state and the new duration time. The state transitions are $(i_n, d_n) \rightarrow (i_{n+1}, d_{n+1})$ for $n = 1, ..., \mathcal{N}$ where $\mathcal{N}$ is the number of state transitions. Note that $\sum_{n=1}^{\mathcal{N}} d_n = T$. The state transition probability for the pair $(i, d') \rightarrow (j, d)$ is

$$\begin{aligned} &a_{(i,d')(j,d)} = P(x_{[t+1:t+d]} = j | x_{[t-d'+1:t]} = i), \\ &\text{subject to } \sum_{j \in \mathcal{X} \setminus \{i\}} \sum_{d \in \mathcal{D}} a_{(i,d')(j,d)} = 1, \quad i, j \in \mathcal{X}, \quad d, d' \in \mathcal{D}. \end{aligned} \tag{2.22}$$

The self transition probability $a_{(i,d')(i,d)}$ is equal to zero. In the general formulation, the current state and the current duration are both dependent upon the previous state and the previous duration. The emission probability is

$$b_{j,d}(y_{t+1:t+d}) = P(y_{t+1:t+d}|x_{[t+1:t+d]} = j). \tag{2.23}$$

The initial state distribution is

$$\pi_{j,d} = P(x_{[t-d+1:t]} = j), \quad t \leq 0, d \in \mathcal{D}. \tag{2.24}$$

The set of model parameters Λ is comprised of the initial state distribution, the transition probabilities, and the emission probabilities. The state duration can be either parametric or non-parametric. When defined separately from the state transition $a_{(i,d')(j,d)}$, the state duration probability is denoted as $p_j(d)$, and these probabilities are included in the set of model parameters Λ.

The forward and backward variables, outlined by Rabiner [95] for the HMM, are also modified for the general HSMM and outlined by Yu [133] as

$$\alpha_t(j,d) = P(x_{[t-d+1:t]} = j, y_{1:t}|\Lambda), \tag{2.25}$$

and

$$\beta_t(j,d) = P(y_{t+1:T}|x_{[t-d+1:t]} = j, \Lambda). \tag{2.26}$$

The formulas for the forward-backward algorithm for an HSMM are

$$\alpha_t(j,d) = \sum_{i \in \mathcal{X} \setminus \{j\}} \sum_{d' \in \mathcal{D}} \alpha_{t-d}(i,d') a_{(i,d')(j,d)} b_{j,d}(y_{t-d+1:t}), \quad t > 0, d \in \mathcal{D}, j \in \mathcal{X}, \tag{2.27}$$

and

$$\beta_t(j,d) = \sum_{i \in \mathcal{X} \setminus \{j\}} \sum_{d' \in \mathcal{D}} a_{(i,d')(j,d)} b_{j,d'}(y_{t+1:t+d'}) \beta_{t+d'}(i,d'), \quad t < T. \tag{2.28}$$

Yu [133] gives two assumptions for the initial conditions. The general assumption is not used in this dissertation so only the simplifying assumption is given. For the

simplifying assumption, it is assumed that at the first time step t_1 the first state begins, and at the last time step T the last state ends. When using the simplifying assumption, the initial conditions for the forward and backward variables are:

$$\begin{aligned} \alpha_0(j,d) &= \pi_{j,d}, \quad && d \in \mathcal{D}, \\ \alpha_\tau(j,d) &= 0, && \tau < 0, d \in \mathcal{D}, \\ \beta_T(i,d) &= 1, && d \in \mathcal{D}, \\ \beta_\tau(i,d) &= 0, && \tau > T, d \in \mathcal{D}. \end{aligned} \tag{2.29}$$

It can also be assumed that the initial distribution $\pi'_{j,d}$ can be defined as $P(x_{[1:d]} = j|\Lambda)$. From this, the initial condition for the forward variable is $\alpha_d(j,d) = \pi'_{j,d} b_{j,d}(y_{1:d})$.

Using the forward and backward variables, Yu [133] shows that several quantities of interest can be calculated. Filtered probabilities are probabilities conditioned on a partial observation sequence. The filtered forward probability is

$$P(x_{[t-d+1:t]} = j|y_{1:t}, \Lambda) = \frac{\alpha_t(j,d)}{\sum_{j,d} \alpha_t(j,d)}. \tag{2.30}$$

Further, the predicted probability that state j has duration d and begins and ends at times $t+1$ and $t+d$, given the observation sequence from time 1 to t is

$$P(x_{[t+1:t+d]} = j|y_{1:t}, \Lambda) = \frac{\sum_{i \neq j,d'} \alpha_t(i,d') a_{(i,d')(j,d)}}{\sum_{j,d'} \alpha_t(i,d')}. \tag{2.31}$$

Summing Equations 2.30 and 2.31 over all d yield

$$\sum_{d \in \mathcal{D}} P(x_{[t-d+1:t]} = j|y_{1:t}, \Lambda) = P(x_{[t]} = j|y_{1:t}, \Lambda), \tag{2.32}$$

and

$$\sum_{d \in \mathcal{D}} P(x_{[t+1:t+d]} = j|y_{1:t}, \Lambda) = P(x_{[t+1} = j|y_{1:t}, \Lambda). \tag{2.33}$$

Yu [133] goes on to derive several joint probabilities that include the entire observation sequence $y_{1:T}$.

$$\begin{aligned} \eta_t(j,d) &= P(x_{[t-d+1:t]} = j, y_{1:T}|\Lambda) \\ &= \alpha_t(j,d)\beta_t(j,d), \end{aligned} \tag{2.34}$$

$$\begin{aligned} \xi_t(i,d';j,d) &= P(x_{[t-d'+1:t]} = i, x_{[t+1:t+d]} = j, y_{1:T}|\Lambda) \\ &= \alpha_t(i,d')a_{(i,d')(j,d)}b_{j,d}(y_{t+1:t+d})\beta_{t+d}(j,d), \end{aligned} \tag{2.35}$$

$$\begin{aligned} \xi_t(i,j) &= P(x_{t]} = i, x_{[t+1} = j, y_{1:T}|\Lambda) \\ &= \sum_{d'\in\mathcal{D}}\sum_{d\in\mathcal{D}} \xi_t(i,d';j,d), \end{aligned} \tag{2.36}$$

and

$$\begin{aligned} \gamma_t(j) &= P(x_t = j, y_{1:T}|\Lambda) \\ &= \sum_{\tau\geq t}\sum_{d=\tau-t+1} \eta_\tau(j,d). \end{aligned} \tag{2.37}$$

The joint probabilities in Equations 2.34 to 2.37 can be converted to posterior probabilities by dividing by the likelihood of the observation sequence

$$\begin{aligned} P(y_{1:T}|\Lambda) &= \sum_{j\in\mathcal{X}} P(x_t = j, y_{1:T}|\Lambda) \\ &= \sum_{j\in\mathcal{X}} \gamma_t(j). \end{aligned} \tag{2.38}$$

The Viterbi algorithm for state prediction is also modified for the HSMM and outlined by Yu [133]. The forward variable for the HSMM Viterbi algorithm is

$$\begin{aligned}\delta_t(j,d) &= \max_{x_{1:t-d}} P(x_{1:t-d}, x_{[t-d+1:t]} = j, y_{1:t}|\Lambda) \\ &= \max_{i\in\mathcal{X}\setminus\{j\},d'\in\mathcal{D}} \{\delta_{t-d}(i,d')a_{(i,d')(j,d)}b_{j,d}(y_{t-d+1:t})\},\end{aligned} \tag{2.39}$$

The above equation is the maximum likelihood that the partial state sequence ends in state j at time t and had duration d. The previous state selected by $\delta_t(j,d)$ is recorded by $\Psi(t,j,d) = (t-d, i^*, d^*)$, with i^* and d^* being the previous surviving state and duration, and $(t-d)$ being the end time for the previous surviving state and duration. $\Psi(t,j,d)$ is selected using

$$(i^*, d^*) = \arg\max_{i\in\mathcal{X}\setminus\{j\},d'\in\mathcal{D}} \{\delta_{t-d}(i,d')a_{(i,d')(j,d)}b_{j,d}(y_{t-d+1:t})\}. \tag{2.40}$$

The most likely state sequence is found by selecting the last state that maximizes the likelihood. When using the simplifying assumption, start at the end of the sequence and find

$$(j_T^*, d_T^*) = \arg\max_{i\in\mathcal{X},d\in\mathcal{D}} \delta_T(i,d). \tag{2.41}$$

Then backtrack to find the entire most likely state sequence

$$\begin{aligned}&(T-1, j_{T-1}^*, d_{T-1}^*) = \Psi(T, j_T^*, d_T^*), \\ &... \\ &(t_1, j_1^*, d_1^*) = \Psi(t_2, j_2^*, d_2^*).\end{aligned} \tag{2.42}$$

The maximum likelihood state sequence consists of both states and durations – $(j_1^*, d_1^*), ..., (j_T^*, d_T^*)$.

The first two primary problems for HMMs have now been covered for HSMMs. The third problem, parameter estimation, can be solved using the EM algorithm. For the general HSMM, Yu [133] the EM algorithm, where the expectations $\eta_t(j,d)$ and $\xi_t(i,d';j,d)$ are calculated in the E-step, and the parameters are estimated in the M-step as follows

$$\hat{\pi}_{j,d} = \frac{\eta_t(j,d)}{\sum_{j,d}\eta_t(j,d)}, \quad t \leq 0, \tag{2.43}$$

$$\hat{a}_{(i,d')(j,d)} = \frac{\sum_t \xi_t(i,d';j,d)}{\sum_{j\neq i,d}\sum_t \xi_t(i,d';j,d)}, \tag{2.44}$$

and

$$\hat{b}_{j,d}(v_{k_1:k_d}) = \frac{\sum_t [\eta_t(j,d)\mathbb{I}(y_{t+1:t+d} = v_{k_1:k_t})]}{\sum_t \eta_t(j,d)}, \tag{2.45}$$

where the indicator function $\mathbb{I}(y_{t+1:t+d} = v_{k_1:k_t}) = 1$ if $y_{t+1} = v_{k_1}, ..., y_{t+d} = v_{k_d}$ and otherwise zero.

The joint probabilities in the forward-backward algorithm will rapidly become small and cause underflow problems when calculating on a computer. To avoid this when implementing an HSMM, Yu [133] suggests calculating predicted probabilities. The forward and backward predicted probabilities are

$$\bar{\alpha}_t(j,d) = P(x_{[t-d+1:t]} = j | y_{1:t}, \Lambda), \tag{2.46}$$

and

$$\bar{\beta}_t(j,d) = \frac{P(y_{t-d+1:T}|x_{[t-d+1:t]} = j, \Lambda)}{P(y_{t-d+1}|y_{1:t-d}, \Lambda)}. \tag{2.47}$$

Likewise, the emission probabilities are redefined as

$$\bar{b}_{j,d}(y_{t-d+1:t}) = \frac{b_{j,d}(y_{t-d+1:t})}{P(y_{t-d+1:t}|y_{1:t-d}, \Lambda)}. \tag{2.48}$$

The conversions between the joint forward and backward variables and the predicted probabilities are

$$\alpha_t(j,d) = P(y_{1:t-d}|\Lambda)\bar{\alpha}_t(j,d)b_{j,d}(y_{t-d+1:t)}, \tag{2.49}$$

and

$$\bar{b}_{j,d}(y_{t-d+1:t})P(y_{t-d+1:T}|y_{1:t-d},\Lambda) = b_{j,d}(y_{t-d+1:t}\beta_t(j,d). \tag{2.50}$$

The calculations for the predicted forward and backward variables are now

$$\bar{\alpha}_t(j,d) = \sum_{i\in\mathcal{X}\setminus\{j\}}\sum_{d'\in\mathcal{D}} \bar{\alpha}_{t-d}(i,d')\bar{b}_{i,d'}(y_{t-d-d'+1}^{t-d})a_{(i,d')(j,d)}, \tag{2.51}$$

and

$$\bar{\beta}_t(j,d) = \bar{b}_{j,d}(y_{t-d+1:t})\sum_{i\in\mathcal{X}\setminus\{j\}}\sum_{d'\in\mathcal{D}} a_{(i,d')(j,d)}\bar{\beta}_{t+d'}(i,d'). \tag{2.52}$$

Probabilities for partial observation sequences can be calculated using

$$\begin{aligned} P(y_{1:t}|\Lambda) &= \sum_{j\in\mathcal{X}}\sum_{d\in\mathcal{D}} P(x_{[t-d+1:t]} = j, y_{1:t}|\Lambda) \\ &= \sum_{j\in\mathcal{X}}\sum_{d\in\mathcal{D}} P(y_{1:t-d}|\Lambda)\bar{\alpha}_t(j,d)b_{j,d}(y_{t-d+1:t}), \end{aligned} \tag{2.53}$$

and

$$P(y_{t-d+1:t}|y_{1:t-d},\Lambda) = \frac{P(y_{1:t}|\Lambda)}{P(y_{1:t-d})}. \tag{2.54}$$

After the forward-backward algorithm is performed, the probabilities in Equations 2.34 and 2.35 can be calculated using

$$\frac{\eta_t(j,d)}{P(y_{1:T})|\Lambda)} = \bar{\alpha}_t(j,d)\bar{\beta}_t(j,d), \tag{2.55}$$

and

$$\frac{\xi_t(i,d';j,d)}{P(y_{1:T}|\Lambda)} = \bar{\alpha}_t(j,d)b_{i,d'}(y_{t-d'+1:t})a_{(i,d')(j,d)}\bar{\beta}_t(j,d). \tag{2.56}$$

Further, simpler forward and backward recursion can be implemented by making

$$\bar{\alpha}'_t(j,d) = \bar{\alpha}_t(j,d) b_{j,d}(y_{t-d+1:t}), \tag{2.57}$$

and

$$\bar{\beta}'_{t+1}(j,d) = \bar{\beta}_{t+d}(j,d). \tag{2.58}$$

The forward and backward variables are now calculated by

$$\bar{\alpha}'_t(j,d) = \bar{b}_{j,d}(y_{t-d+1:t}) \sum_{i \in \mathcal{X} \setminus \{j\}} \sum_{d' \in \mathcal{D}} \bar{\alpha}'_{t-d}(i,d') a_{(i,d')(j,d)}, \tag{2.59}$$

and

$$\bar{\beta}'_t(j,d) = \bar{b}_{j,d}(y_{t:t+d-1}) \sum_{i \in \mathcal{X} \setminus \{j\}} \sum_{d' \in \mathcal{D}} \bar{\beta}'_{t+d}(i,d') a_{(i,d')(j,d)}. \tag{2.60}$$

2.2.1 Explicit Duration Hidden Markov Model

The explicit duration hidden Markov model (EDHMM) is one of many specific types of HSMMs. Other types of HSMMs include the variable transition HMM and the residential time HMM. The general HSMM becomes one of these models when specific assumptions about the state transitions and durations are made.

Yu [133] outlines the EDHMM in the following manner. The EDHMM assumes that the state transition and duration of the previous state are independent. Further, the distribution of the duration is assumed to be determined only by the current state. The transition probability in the general model can now be broken into two terms, $a_{(i,d')(j,d)} = a_{ij} p_j(d)$ with zero probability of a self transition $a_{ii} = 0$, and where the transition probability is defined as $a_{ij} = P(x_{[t} = j | x_{t-1]} = i)$. The duration probability $p_j(d)$ can be either parametric or non-parametric. These independence assumptions drastically decrease the complexity when compared to the general HSMM by reducing the number of parameters.

In addition to the changes to the transition probabilities, the following changes are made to the general HSMM to convert it to an EDHMM: the observation probability $b_{j,d}(y_{t+1:t+d})$ is replaced with $\prod_{\tau=t+1}^{t+d} b_j(y_\tau)$, the backward variable $\beta_t(j,d)$ is

replaced with $\beta_t(j) = P(y_{t+1:T}|x_{t]} = i, \Lambda)$, and the forward variable is redefined as $\alpha_t(j) = P(x_{t]} = j, y_{1:t}|\Lambda) = \sum_{d\in\mathcal{D}} \alpha_t(j,d)$. The forward and backward recursions for the EDHMM are now

$$\alpha_t(j) = \sum_{d\in\mathcal{D}} \alpha^*_{t-d}(j) p_j(d) u_t(j,d), \tag{2.61}$$

$$\begin{aligned} \alpha^*_t(j) &= P(x_{[t+1} = j, y_{1:t}|\Lambda) \\ &= \sum_{i\in\mathcal{X}\setminus\{j\}} \alpha_t(i) a_{ij}, \end{aligned} \tag{2.62}$$

and

$$\begin{aligned} \beta^*_t(j) &= P(y_{t+1:T}|x_{[t+1} = j, \Lambda) \\ &= \sum_{d\in\mathcal{D}} p_j(d) u_{t+d}(j,d) \beta_{t+d}(j), \end{aligned} \tag{2.63}$$

$$\beta_t(j) = \sum_{i\in\mathcal{X}\setminus\{j\}} a_{ji} \beta^*_t(j), \tag{2.64}$$

where

$$u_t(j,d) = \prod_{\tau=t-d+1}^{t} b_j(y_\tau). \tag{2.65}$$

When using the simplifying assumptions, the boundary conditions are $\alpha_0(i) = \pi_i$ and $\alpha_\tau(i) = 0$ for $\tau < 0$, and $\beta_T(i) = 1$ and $\beta_\tau(i) = 0 \; \forall \; \tau > T$.

There are some slight adjustments to the formulations outlined by Yu in [133] when solving for the parameter estimates and implementing the Viterbi algorithm,. Specifically, the forward variable and the forward recursion in Viterbi must include the duration.

The joint probability of the hidden states $\mathbf{x}$, the duration in each state $\mathcal{D}$, and the observed emissions $\mathbf{y}$ is

$$P(\mathbf{x}, \mathbf{y}, \mathcal{D}) = \pi_{x_1} p_{x_1}(d_1) \left[\prod_{\tau=1}^{d_1} f_{x_1}(y_\tau) \right] \left\{ \prod_{n=2}^{\mathcal{N}} a_{x_{n-1},x_n} p_{x_n}(d_n) \left[\prod_{\tau=\hat{d}_n+1}^{\hat{d}_n+d_n} f_{x_n}(y_\tau) \right] \right\}, \tag{2.66}$$

where

$$\hat{d}_n = \sum_{\hat{n}=1}^{n-1} d_{\hat{n}}. \tag{2.67}$$

As previously noted, $\sum_{n=1}^{\mathcal{N}} d_n = T$. It also should be noted that the equation for the likelihood of the EDHMM in Equation 2.66 uses the number of transitions n as the subscript on the state. This is different from the likelihood for the standard HMM in Equation 2.1 which uses time t as the subscript.

In this dissertation, a Gaussian distribution is used for the emission distribution, and a truncated Poisson distribution is used to model the duration. The general Poisson distribution has a range of $[0, \infty)$ and the probability mass function (PMF) for the Poisson distribution is

$$P(d|\lambda) = \frac{\lambda^d e^{-\lambda}}{d!}. \tag{2.68}$$

When using an EDHMM, the minimum duration must be greater than 0 but does not need to be restricted to 1. The maximum duration D is a finite number; therefore, a truncated Poisson with a minimum and maximum on the range is used for $p_x(d)$. The truncated Poisson is the general Poisson PMF divided by the sum of the PMF over the range of d

$$P(d|\lambda) = \frac{\frac{\lambda^d e^{-\lambda}}{d!}}{\sum_{d_{min}}^{D} \frac{\lambda^d e^{-\lambda}}{d!}} = \frac{\lambda^d \sum_{d_{min}}^{D} d!}{d! \sum_{d_{min}}^{D} \lambda^d}. \tag{2.69}$$

The truncated Poisson can be easily calculated by dividing the Poisson PMF by the difference of the cumulative distribution function (CDF) at the max duration and the CDF at one minus the minimum duration.

The parameters of the EDHMM can be estimated using Baum-Welch when the state sequence and duration sequence are unknown. As mentioned earlier, the

forward variable must be adjusted from the formulation given by Yu [133] to include the state duration

$$\alpha_t(j,d) = \sum_{i=1}^{I} \sum_{d'=d_{min}}^{D} \alpha_{t-d}(i,d') a_{ij} p_j(d) \prod_{\tau=t-d+1}^{t} f_j(y_\tau). \tag{2.70}$$

The forward and backward variables in Equations 2.70 and 2.64 are used during the E-step to calculate the expectations – Equations 2.34, 2.35 and 2.37. During implementation, these expectation calculations should be adjusted to the predicted variables as described in the general HSMM section. The M-step parameter updates for the EDHMM are:

$$\hat{\pi}_i = \gamma_1(i), \tag{2.71}$$

$$\begin{aligned} \hat{a}_{ij} &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{j=1}^{I} \xi_t(i,j)} \\ &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \end{aligned} \tag{2.72}$$

$$\hat{\mu}_{il} = \frac{\sum_{t=1}^{T} \gamma_t(i) y_{lt}}{\sum_{t=1}^{T} \gamma_t(i)}, \tag{2.73}$$

$$\hat{\sigma}_{il} = \sqrt{\frac{\sum_{t=1}^{T} \gamma_t(i)(y_{lt} - \mu_{il})^2}{\sum_{t=1}^{T} \gamma_t(i)}}, \tag{2.74}$$

and

$$\lambda_i = \frac{\sum_{t=1}^{T} \sum_{d=d_{min}}^{D} \eta_t(i,d) d}{\sum_{t=1}^{T} \sum_{d=d_{min}}^{D} \eta_t(i,d)}. \tag{2.75}$$

The forward recursion for the Viterbi algorithm is

$$\delta_t(j,d) = \max_{i,d'} \{\delta_{t-d}(i,d') a_{ij}\} p_j(d) \prod_{\tau=t-d+1}^{t} f_j(y_\tau). \tag{2.76}$$

The rest of the Viterbi algorithm is the same as described in the general HSMM section.

Chapter 3

Feature Saliency Hidden Markov Model

In this chapter, we outline multiple formulations for the FSHMM. The maximum likelihood (ML) formulation does not use priors on any model parameters. The maximum a posteriori (MAP) formulation uses priors on all model parameters and an exponential prior on the feature saliencies. MAP-beta uses mostly the same priors as MAP, a beta distribution is used as a prior on the feature saliencies instead of the exponential. These formulations are compared to the VB formulation in [137] on the PHM and Kinect data sets. Conventional FS techniques are also tested on these data sets. The section outlining the model and the results on the PHM and Kinect data sets are taken from [1].

3.1 Feature Saliency Hidden Markov Model

The feature saliency hidden Markov model (FSHMM) uses a feature saliency model for the emission distribution allowing for simultaneous feature selection and parameter estimation [62]. A feature is considered to be relevant if its distribution is dependent on the underlying state, and irrelevant if its distribution is independent of the state. Let $\mathbf{z} = \{z_1, \ldots, z_L\}$ be a set of binary variables indicating the relevancy of each feature. If $z_l = 1$, then the l-th feature is relevant. Otherwise, if $z_l = 0$ the l-th feature is irrelevant. The feature saliency ρ_l is defined as the probability that the l-th feature is relevant. Assuming the features are conditionally

independent given the state, allows the conditional distribution of y_t given $\mathbf{z}$ and $\mathbf{x}$ to be written as

$$P(y_t|\mathbf{z}, x_t = i, \Lambda) = \prod_{l=1}^{L} p(y_{lt}|\mu_{il}, \sigma_{il}^2)^{z_l} q(y_{lt}|\epsilon_l, \tau_l^2)^{1-z_l}, \tag{3.1}$$

where $p(y_{lt}|\mu_{il}, \sigma_{il}^2)$ is the Gaussian conditional feature distribution for the l-th feature with state-dependent mean μ_{il} and state-dependent variance σ_{il}^2; and $q(y_{lt}|\epsilon_l, \tau_l^2)$ is the state-independent Gaussian feature distribution with mean ϵ_l and variance τ_l^2. The set of model parameters Λ for the FSHMM is $\{\pi, A, \mu, \sigma, \rho, \epsilon, \tau\}$.

The marginal probability of $\mathbf{z}$ is

$$P(\mathbf{z}|\Lambda) = \prod_{l=1}^{L} \rho_l^{z_l} (1 - \rho_l)^{1-z_l}. \tag{3.2}$$

The joint distribution of y_t and $\mathbf{z}$, given $\mathbf{x}$, is

$$\begin{aligned} P(y_t, \mathbf{z}|x_t = i, \Lambda) &= P(y_t|\mathbf{z}, x_t = i, \Lambda) P(\mathbf{z}|\Lambda) \\ &= \prod_{l=1}^{L} [\rho_l p(y_{lt}|\mu_{il}, \sigma_{il}^2)]^{z_l} [(1 - \rho_l) q(y_{lt}|\epsilon_l, \tau_l^2)]^{1-z_l}. \end{aligned} \tag{3.3}$$

The marginal distribution for y_t, given $\mathbf{x}$, can be found by summing Equation 3.3 over z

$$\begin{aligned} f_{x_t}(y_t) &= P(y_t|x_t = i, \Lambda) \\ &= \prod_{l=1}^{L} \left(\rho_l p(y_{lt}|\mu_{il}, \sigma_{il}^2) + (1 - \rho_l) q(y_{lt}|\epsilon_l, \tau_l^2) \right). \end{aligned} \tag{3.4}$$

The derivations for Equations 3.3 and 3.4 are found in Appendix A Section A.1.

The complete data likelihood for the FSHMM is

$$P(\mathbf{x}, \mathbf{y}, \mathbf{z}|\Lambda) = \pi_{x_1} P(y_1, \mathbf{z}|x_1, \Lambda) \prod_{t=2}^{T} a_{x_{t-1}, x_t} P(y_t, \mathbf{z}|x_t, \Lambda). \tag{3.5}$$

3.2 EM Algorithm for FSHMM

For the FSHMM, the set of hidden variables is comprised of the states and the indicator of the feature relevance. The $\mathcal{Q}$ function outlined for the standard HMM in Section 2.1.1 is modified to include $\mathbf{z}$

$$\begin{aligned} \mathcal{Q}(\Lambda, \Lambda') &= \mathbb{E}[\log P(\mathbf{x}, \mathbf{y}, \mathbf{z}|\Lambda)|\mathbf{y}, \Lambda'] \\ &= \sum_{\mathbf{x},\mathbf{z}} \log(P(\mathbf{x}, \mathbf{y}, \mathbf{z}|\Lambda)) P(\mathbf{x}, \mathbf{z}|\mathbf{y}, \Lambda'). \end{aligned} \tag{3.6}$$

For MAP estimation, the log of the priors is added to the $\mathcal{Q}$ function

$$\mathcal{Q}(\Lambda, \Lambda') + \log(G(\Lambda)). \tag{3.7}$$

The derivations for the $\mathcal{Q}$ function are in Appendix A Section A.2.

3.2.1 E Step

For the E-step, calculate $\gamma_t(i)$ and $\xi_t(i, j)$ using the forward-backward algorithm (Equations 2.12 and 2.13). Additionally, calculate

$$\begin{aligned} e_{ilt} &= P(y_{lt}, z_l = 1|x_t = i, \Lambda') \\ &= \rho_l p(y_{lt}|\mu_{il}, \sigma_{il}^2), \end{aligned} \tag{3.8}$$

$$\begin{aligned} h_{ilt} &= P(y_{lt}, z_l = 0|x_t = i, \Lambda') \\ &= (1 - \rho_l) q(y_{lt}|\epsilon_l, \tau_l^2), \end{aligned} \tag{3.9}$$

$$\begin{aligned} g_{ilt} &= P(y_{lt}|x_t = i, \Lambda') \\ &= e_{ilt} + h_{ilt}, \end{aligned} \tag{3.10}$$

$$\begin{aligned} u_{ilt} &= P(z_l = 1, x_t = i|\mathbf{y}, \Lambda') \\ &= \frac{\gamma_t(i)e_{ilt}}{g_{ilt}}, \end{aligned} \tag{3.11}$$

and

$$\begin{aligned} v_{ilt} &= P(z_l = 0, x_t = i|\mathbf{y}, \Lambda') \\ &= \frac{\gamma_t(i)h_{ilt}}{g_{ilt}} \\ &= \gamma_t(i) - u_{ilt}, \end{aligned} \tag{3.12}$$

where $\gamma_t(i)$, $\xi_t(i,j)$, u_{ilt} and v_{ilt} will be used in the M-step.

3.2.2 ML M-step

The parameter estimates for the initial state distribution and the transition probabilities are the same as in the Baum-Welch algorithm. The estimates for the parameters of $p(\cdot|\cdot)$ and $q(\cdot|\cdot)$ and the feature saliencies use the probabilities defined in the previous section – Equations 3.11 and 3.12. Specifically, all ML parameter estimates are given by

$$\hat{\pi}_i = \gamma_1(i), \tag{3.13}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \tag{3.14}$$

$$\hat{\mu}_{il} = \frac{\sum_{t=1}^{T} u_{ilt} y_{lt}}{\sum_{t=1}^{T} u_{ilt}}, \tag{3.15}$$

$$\hat{\sigma}_{il}^2 = \frac{\sum_{t=1}^{T} u_{ilt}(y_{lt} - \hat{\mu}_{il})^2}{\sum_{t=1}^{T} u_{ilt}}, \tag{3.16}$$

$$\hat{\epsilon}_l = \frac{\sum_{t=1}^{T} \big(\sum_{i=1}^{I} v_{ilt}\big) y_{lt}}{\sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}}, \tag{3.17}$$

$$\hat{\tau}_l^2 = \frac{\sum_{t=1}^{T} \big(\sum_{i=1}^{I} v_{ilt}\big)(y_{lt} - \hat{\epsilon}_l)^2}{\sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}}, \tag{3.18}$$

and

$$\begin{aligned}\hat{\rho}_l &= \frac{\sum_{t=1}^{T}\sum_{i=1}^{I} u_{ilt}}{\sum_{t=1}^{T}\sum_{i=1}^{I} u_{ilt} + \sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}} \\ &= \frac{\sum_{t=1}^{T}\sum_{i=1}^{I} u_{i,l,t}}{T}.\end{aligned} \tag{3.19}$$

Derivations for these parameter update equations are in Appendix A Section A.3.

3.2.3 MAP M-step

The priors used for MAP estimation are listed below. Dir is the Dirichlet distribution, $\mathcal{N}$ is the Gaussian distribution, IG is the inverse gamma distribution, and A_i is row i of the transition matrix.

$$\pi \sim \mathrm{Dir}(\pi|\bar{\boldsymbol{p}}), \tag{3.20}$$

$$A_i \sim \mathrm{Dir}(A_i|\bar{\boldsymbol{a}}_i), \tag{3.21}$$

$$\mu_{il} \sim \mathcal{N}(\mu_{il}|m_{il}, s_{il}^2), \tag{3.22}$$

$$\sigma_{il}^2 \sim \mathrm{IG}(\sigma_{il}^2|\zeta_{il}, \eta_{il}), \tag{3.23}$$

$$\epsilon_l \sim \mathcal{N}(\epsilon_l|b_l, c_l^2), \tag{3.24}$$

$$\tau_l^2 \sim \mathrm{IG}(\tau_l^2|\nu_l, \psi_l), \tag{3.25}$$

$$\rho_l \sim \frac{1}{Z} e^{-k_l \rho_l}, \tag{3.26}$$

where Z is the normalizing constant. The probability distributions are defined in Appendix C.

The parameter update equations are

$$\hat{\pi}_i = \frac{\gamma_1(i) + \bar{p}_i - 1}{\sum_{i=1}^{I}\left(\gamma_1(i) + \bar{p}_i - 1\right)}, \tag{3.27}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1}\xi_t(i,j) + \bar{a}_{ij} - 1}{\sum_{j=1}^{I}\left(\sum_{t=1}^{T-1}\xi_t(i,j) + \bar{a}_{ij} - 1\right)}, \tag{3.28}$$

$$\hat{\mu}_{il} = \frac{s_{il}^2 \sum_{t=1}^{T} u_{ilt} y_{lt} + \hat{\sigma}_{il}^2 m_{il}}{s_{il}^2 \sum_{t=1}^{T} u_{ilt} + \hat{\sigma}_{il}^2}, \tag{3.29}$$

$$\hat{\sigma}_{il}^2 = \frac{\sum_{t=1}^{T} u_{ilt}(y_{lt} - \hat{\mu}_{il})^2 + 2\eta_{il}}{\sum_{t=1}^{T} u_{ilt} + 2(\zeta_{il} + 1)}, \tag{3.30}$$

$$\hat{\epsilon}_l = \frac{c_l^2 \sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) y_{lt} + \hat{\tau}_l^2 b_l}{c_l^2 \sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) + \hat{\tau}_l^2}, \tag{3.31}$$

$$\hat{\tau}_l^2 = \frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) (y_{lt} - \hat{\epsilon}_l)^2 + 2\psi_l}{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) + 2(\nu_l + 1)}, \tag{3.32}$$

and

$$\hat{\rho}_l = \frac{T + k_l - \sqrt{(T + k_l)^2 - 4k_l(\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt})}}{2k_l}. \tag{3.33}$$

Derivations for these parameter update equations are in Appendix A Section A.4. The ML and MAP parameter updates can be easily adjusted for multiple observation sequences.

The graphical model for the MAP formulation using the exponential prior is in Figure 3.1.

The MAP formulation, specifically the use of an informative exponential prior on ρ, allows the user to affect the estimated saliency of each feature. For higher values of k_l, more evidence that a feature is relevant is needed for the algorithm to estimate higher values of ρ_l, and thus select the l^{th} feature as relevant. This can be used to limit the number of features selected by the algorithm, or input the cost of collecting the feature into the feature selection process.

Another choice for a prior on ρ is the beta distribution, because estimates of ρ are restricted to [0,1]. However, we found that it does not perform as well when estimating the relevance of features, even though it seems more appropriate than the truncated exponential prior. The parameter update for ρ using the beta prior $\mathcal{B}(\rho_l | k_l, \kappa_l)$ is

$$\rho_l = \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt} + k_l - 1}{T + k_l + \kappa_l - 2}. \tag{3.34}$$

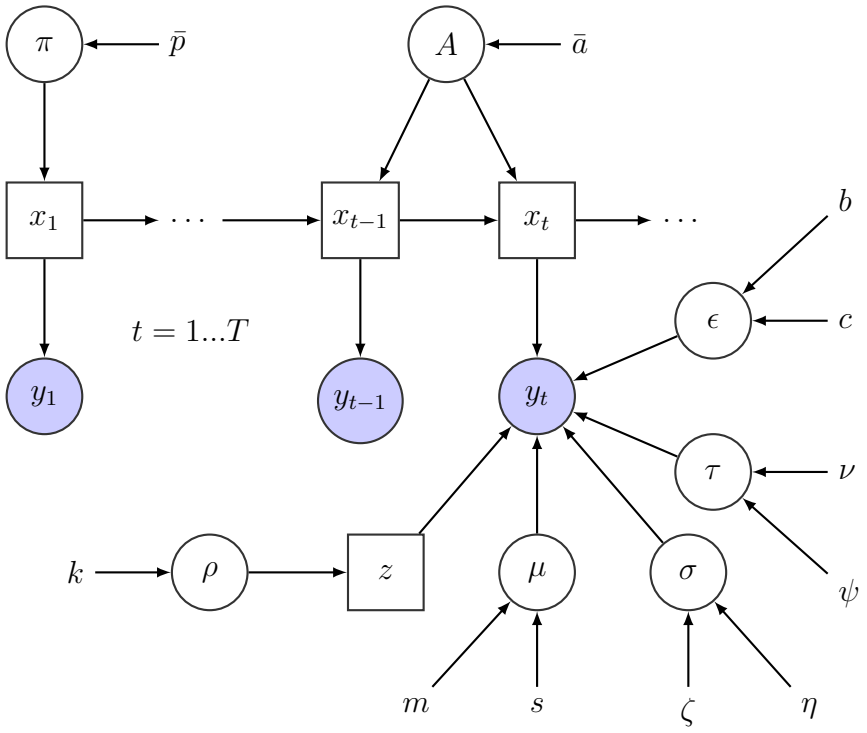

FIGURE 3.1: Graphical model for MAP formulation using exponential prior. Squares represent hidden variables. Filled circles are observable variables. Open circles are model parameters.

In order to ensure that $\rho_l \geq 0$, we must set $k_l \geq 1 \, \forall l$. When $k_l = 1$, the parameter estimate for ρ simplifies to

$$\rho_l = \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt}}{T + \kappa_l - 1}. \tag{3.35}$$

As long as there is more than one observation in $\mathbf{y}$, the denominator of Equation (3.35) will be greater than 0. For the remainder of the dissertation, we will designate the formulation using the truncated exponential as MAP, and the formulation using the beta prior as MAP-beta. The graphical model for the MAP formulation using a beta prior is displayed in Figure 3.2.

An algorithm for estimating the parameters of the MAP FSHMM is given in Algorithm 1. MAP-beta follows the same algorithm, but must select the additional hyperparameter κ_l in Step 2. This algorithm can be modified for ML estimation by removing Step 2 and using the M-step update equations in Section 3.2.2 for Step 6.

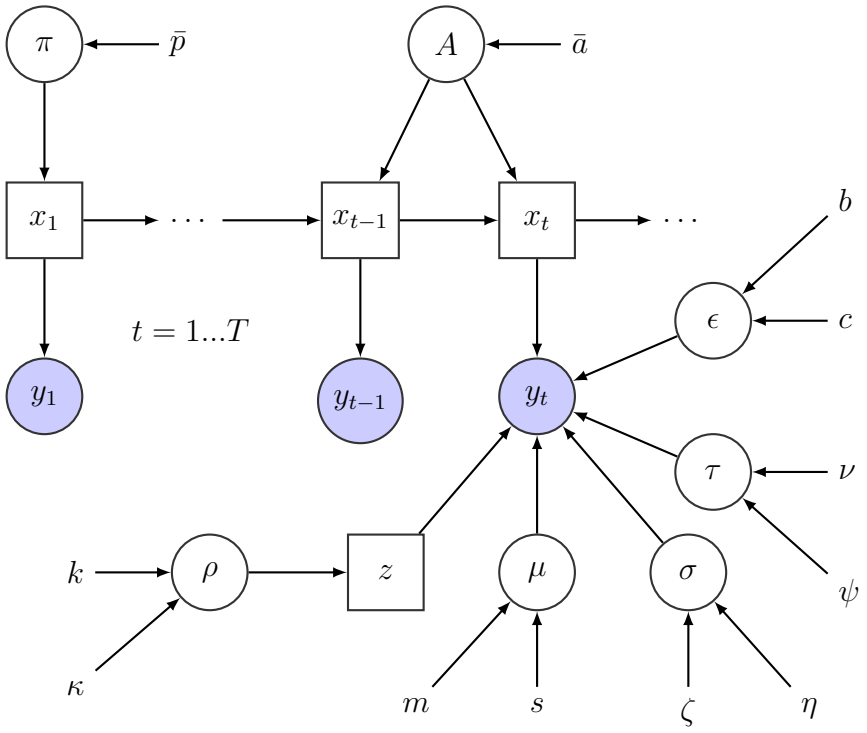

FIGURE 3.2: Graphical model for MAP formulation using beta prior. Squares represent hidden variables. Filled circles are observable variables. Open circles are model parameters.

Algorithm 1 MAP FSHMM Algorithm

1. Select initial values for $\pi_i, a_{ij}, \mu_{il}, \sigma_{il}, \epsilon_l, \tau_l$ and ρ_l for $i = 1..I, j = 1...I$, and $l = 1...L$
2. Select hyperparameters $\bar{p}_i, \bar{a}_{ij}, m_{il}, s_{il}, \zeta_{il}, \eta_{il}, b_l, c_l, \nu_l, \psi_l$, and k_l for $i = 1..I, j = 1...I$, and $l = 1...L$
3. Select stopping threshold δ and maximum number of iterations M
4. Set absolute percent change in the posterior probability between current iteration and previous iteration $\Delta\mathcal{L}$ to ∞ and the number of iterations it to 1
4. **while** $\Delta\mathcal{L} > \delta$ and $it < M$ **do**
5. E-step: calculate probabilities in Section 3.2.1
6. M-step: update parameters in Section 3.2.3
7. calculate $\Delta\mathcal{L}$
8. $it = it + 1$
9. **end while**
10. Perform feature selection based on ρ_l and construct reduced models

3.3 Prediction

When predicting states using the Viterbi algorithm, there are two possible choices for $p(y_t|x_t = i, \Lambda)$: (1) the full conditional distribution using both state-dependent and state-independent parameters (Equation 3.1), or (2) a reduced conditional

distribution using only the state-dependent parameters. The reduced conditional distribution is

$$P(y_t|x_t = i, \Lambda) = \prod_{l=1}^{L} p(y_{lt}|\mu_{il}, \sigma_{il}^2). \tag{3.36}$$

The reduced conditional decreases the number of parameters that need to be stored after training, and reduces the computational resources need for prediction. The model using Equation 3.36 during prediction resembles a standard HMM. The rationale behind the reduced conditional distribution is that if ρ is close to 1 for a particular feature, the contribution of the $q(y_{lt}|\epsilon_l, \tau_l^2)$ to $P(y_t|x_t = i, \Lambda)$ is small. This will be confirmed by experiments on the PHM data. However, in cases where ρ is not close to 1, the full conditional distribution might give better predictive results. However, it is not clear that the reduced conditional should be used for prediction under all circumstances.

3.4 Synthetic Data Experiments

In this section, ML, MAP, MAP-beta, and VB are tested on synthetic data. Three observation sequences are produced by a model with two relevant features. The two dimensional vectors of relevant features are generated from $\mathcal{N}(\mu_i, \Sigma)$. The model has two states, and 500 time steps are generated for each sequence. The model parameters are:

$$\mu_1 = \begin{bmatrix} 10 & 20 \end{bmatrix}, \quad \mu_2 = \begin{bmatrix} 30 & 60 \end{bmatrix}, \quad \Sigma = \begin{bmatrix} 25 & 0 \\ 0 & 25 \end{bmatrix},$$

$$A = \begin{bmatrix} 0.75 & 0.25 \\ 0.4 & 0.6 \end{bmatrix}, \quad \pi = \begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix}.$$

Three irrelevant features of random noise, generated from $\mathcal{N}(0, I)$, are added to the data, resulting in a model with five features in total.

The hyperparameters for the priors in the MAP formulation are: $\bar{p}_i = \bar{a}_{ij} = 2, s_{il} = \zeta_{il} = \eta_{il} = \nu_l = \psi_l = 0.5$, and $c_{il} = 1$. b_l is the mean of the observations

Algorithm	$\hat{\pi}_1$	$\hat{\pi}_2$	$\hat{a}_{11}$	$\hat{a}_{12}$	$\hat{a}_{21}$	$\hat{a}_{22}$
ML	0.6667	0.3333	0.7797	0.2203	0.3933	0.6067
MAP	0.6	0.4	0.7792	0.2208	0.3937	0.6063
MAP-beta	0.6	0.4	0.7782	0.2218	0.3885	0.6115
VB	0.7309	0.2691	0.7795	0.2205	0.3921	0.6079

TABLE 3.1: Parameter estimates for initial distribution and transition matrix. The initial state distribution is biased by the training set. The transition probabilities are within 0.03 units of the true value.

Algorithm	$\hat{\mu}_{11}$	$\hat{\mu}_{12}$	$\hat{\mu}_{21}$	$\hat{\mu}_{22}$
ML	10.1368	20.2196	30.1373	60.0852
MAP	9.7914	19.7352	29.8442	59.1704
MAP-beta	9.1135	19.1568	30.0574	59.1704
VB	10.1237	20.2042	30.1245	60.0841

TABLE 3.2: Parameter estimates for μ for relevant features. All estimates are within 1 unit of the true value.

for the l^{th} feature. m_{1l} is b_l minus 1 standard deviation, and m_{2l} is b_l plus 1 standard deviation. For the feature saliencies, the weight parameter k_l is set to 50. (At $k_l \geq 50$, the majority of the exponential prior's density lies between zero and one.) For MAP-beta, the hyperparameters are the same, except that $k_l = 1$ and $\kappa_l = 49$, giving the same expectation as the truncated exponential prior. The hyperparmeters for the VB formulation are the same as described in [137].

The algorithms are initialized using equal initial probabilities and transition probabilities. μ is randomly selected, $\sigma = 4$, $\epsilon = b$, and τ is the standard deviation of the data. The feature saliencies are always initially set to 0.5. The algorithms are run for a maximum of 1000 iterations. Convergence is tested by calculating the absolute percent change in the likelihood for ML, the posterior probability for MAP and MAP-beta, and the lower bound for VB. The convergence threshold is 10^{-9}.

The estimates for the VB formulation are the expectations of the approximate posterior distributions. The synthetic data set has two observation sequences that start in state 1, and the other starts in state 2. The model parameters estimated by the three approaches are all listed in Tables 3.1 to 3.6.

The estimates for π do not match their true values because the training data is skewed – i.e., two of the three sequences begin in state 1. ML exactly estimates from the training data the proportion of sequences that start in each state, while MAP, MAP-beta, and VB are affected by their priors. All the estimates for A are

Algorithm	$\hat{\sigma}_{11}$	$\hat{\sigma}_{12}$	$\hat{\sigma}_{21}$	$\hat{\sigma}_{22}$
ML	4.8290	4.8286	5.0016	4.9561
MAP	4.8100	4.8413	4.9696	5.0260
MAP-beta	4.3589	4.4238	4.7860	5.0261
VB	4.8159	4.8204	5.0102	4.9607

TABLE 3.3: Parameter estimates for σ for relevant features. All estimates are within 1 unit of the true value.

Algorithm	$\hat{\epsilon}_3$	$\hat{\epsilon}_4$	$\hat{\epsilon}_5$
ML	-0.0388	-0.4889	0.0059
MAP	-0.0109	-0.0776	0.0014
MAP-beta	-0.0109	-0.0776	0.0014
VB	-0.0109	-0.1190	0.0014

TABLE 3.4: Parameter estimates for ϵ for irrelevant features. All estimates are within 0.5 units of the true value which is 0.

Algorithm	$\hat{\tau}_3$	$\hat{\tau}_4$	$\hat{\tau}_5$
ML	0.9538	0.8306	0.9298
MAP	0.9828	1.0022	0.9697
MAP-beta	0.9828	1.0022	0.9697
VB	0.9672	0.9481	0.9415

TABLE 3.5: Parameter estimates for τ for irrelevant features. All estimates are within 0.5 units of the true value which is 1.

Algorithm	$\hat{\rho}_1$	$\hat{\rho}_2$	$\hat{\rho}_3$	$\hat{\rho}_4$	$\hat{\rho}_5$
ML	0.9935	1.0000	0.2125	0.3293	0.1953
MAP	0.9897	0.9999	2.9670×10^{-7}	1.3311×10^{-5}	2.5308×10^{-6}
MAP-beta	0.8873	0.9159	1.7390×10^{-7}	1.8658×10^{-5}	1.4998×10^{-6}
VB	0.9935	0.9989	10^{-9}	0.0234	10^{-9}

TABLE 3.6: Parameter estimates for feature saliencies of all features. ML overestimates the relevance of the irrelevant features. MAP-beta underestimates the relevance of the relevant features. MAP and VB successfully identify the relevant and irrelevant features.

within 12% of their true value. The estimates for the state dependent model parameters (μ and σ) are within 1 unit of their true values for all four methods. The estimates for ρ corresponding to the relevant features are above 0.99 for ML, MAP and VB. MAP-beta underestimates ρ for the relevant features. We find that underestimating the relevance of relevant features is the primary disadvantage of the MAP-beta formulation. When $k_l > 1$ and κ_l is chosen so the expectation matches the exponential prior, MAP-beta continues to underestimate ρ for the relevant features. The ML method produces the highest ρ for the irrelevant features, while

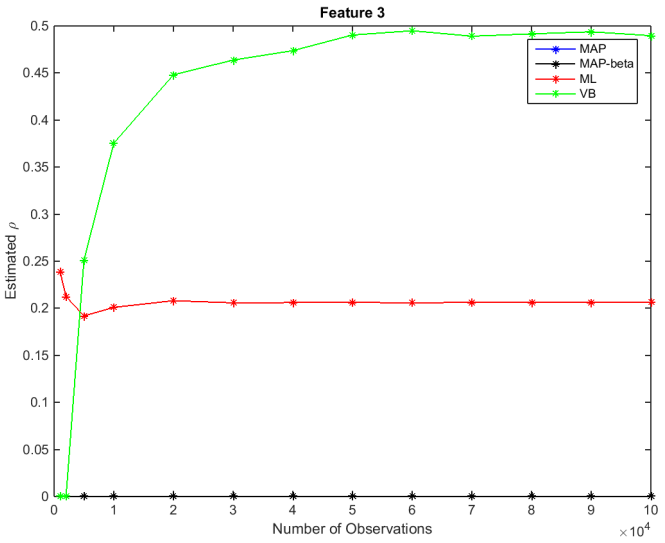

FIGURE 3.3: Plot for estimated ρ for feature 3, a non-relevant feature, as the number of observations in the training set increases. The VB estimate for ρ increases with the number of observations. The ML estimate remains constant. The prior on ρ in MAP and MAP-beta is used to keep estimates of ρ close to 0.

MAP, MAP-beta and VB produce smaller estimates. (In order to prevent division by zero in the VB algorithm, an upper and lower bound is placed on the saliencies. For this experiment, these were $1 - 10^{-9}$ and 10^{-9}.) Overestimating the relevance of irrelevant features is one disadvantage of the ML formulation.

After 1000 iterations, the VB method had not converged. MAP converged after 359 iterations, MAP-beta after 366 iterations, and ML after 372 iterations. When either the number of iterations is decreased or the convergence threshold is increased, the VB method produces greater saliency estimates for the irrelevant features. The estimates produced by MAP, MAP-beta and ML are not significantly affected by changes in the maximum number of iterations or the convergence threshold.

Through testing, we found that the VB method is sensitive to the number of observations in the training data. Specifically, the estimated feature saliencies for non-relevant features increase as the number of observations increase. To illustrate, we test on synthetic data. The same model as above is used to generate a sequence of 100,000 observations. We train models using an increasing number of observations from the synthetic data.

The same starting values and hyperparameters as the previous test are used. However, we change k in the MAP formulation to scale with the number of observations ($k = T/4$). For MAP-beta, we use $k = 1$ and scale κ.

As previously stated, the non-relevant feature saliencies in the VB formulation increase with the number of observations. There is an increase in the relevant feature saliencies but it is very small in comparison. The feature saliencies for relevant and non-relevant features for ML, MAP and MAP-beta appear unaffected by the number of observations. The estimated ρ for each algorithm for feature 3, a non-relevant feature, are plotted in Figure 3.3. The estimates for MAP and MAP-beta are indistinguishable and very close to 0, because Feature 3 is irrelevant.

This example leads to a heuristic for choosing the value of k_l for the exponential prior and κ_l for the beta prior: given no other information about the system or cost of features, one quarter of the total number of observations is a reasonable initial choice for the hyperparameters of the priors on ρ.

3.5 PHM Data

The ML, MAP, MAP-beta and VB algorithms are compared using a tool wear data set. The 2010 Prognostics and Health Management (PHM) conference[1] used this data set for their data challenge. The data consists of six tools used for 315 cuts on a CNC milling machine. Three tools (designated Tools 1, 4 and 6) are supervised and have corresponding wear measurements. Three tools (Tools 2, 3 and 5) are unsupervised. Force and vibration in three directions (X, Y and Z) are collected for each cut.

Root mean square (RMS), the sum of log energies (SLE) and maximum energy (ME) are calculated from each sensor and used as features. The features are normalized so their scales are similar. The cost of each force sensor is assumed to be \$2,400, and the cost of each vibration sensor is assumed to be \$1,200. These values were selected after reviewing the prices of several commercial sensor. For this set of numerical experiments, we assume that each direction of each sensor can be removed while leaving the other directions in place. For example, the force sensor in the X direction can be removed from the mill without removing the Y and Z directions.

A leave-one-out cross validation (LOOCV) testing methodology is implemented: one of the supervised tools is removed from the data , a model is trained on the

[1] http://www.phmsociety.org/competition/phm/10

remaining five tools, and then the model is tested on the withheld tool. LOOCV is applied to each of the three supervised tools so that all tools with wear measurements are tested. The wear measurements are divided into equally spaced discrete bins. The wear bins correspond to the hidden states $\mathbf{x}$ of the HMM. The Viterbi algorithm using the reduced conditional distribution (Equation 3.36) is used to predict the state of the test tool. The median of the wear bin is used as the predicted wear value. Root mean squared error (RMSE) between the true wear measurement, and the predicted wear value is calculated as a measure of accuracy. Models with 5, 10 and 20 hidden states are trained. Models using all 18 features are referred to as full models. The estimates for ρ are used to remove features and construct reduced models.

For reduced models, all features associated with a single sensor are removed reducing the feature set by three features. $\bar{\rho}$ is the mean of the three ρ associated with a particular sensor. For example, $\bar{\rho}_{ForceX}$ is the mean of ρ for RMS, SLE and ME for the force in the X direction. The sensor with the smallest $\bar{\rho}$ is removed to form the reduced model, decreasing the number of features from 18 in the full model to 15 in the reduced model.

Tool wear is non-decreasing, so a left-to-right Markov chain is assumed for the hidden states. However, the stick breaking representation of the hierarchical Dirichlet process in the VB formulation does not allow for a left-to-right Markov chain.

The ML, MAP and MAP-beta algorithms are initialized with the same values. The initial self transition a_{ii} is 0.95, and the transition to the next state $a_{i,i+1}$ is 0.05. Because we assume that the tool at $t = 1$ is new, $\pi_1 = 1$. The state-dependent means μ_{il} are equally spaced between -2 and 2. The state-dependent standard deviation σ_{il} is 1 for all states and features. The state-independent parameters are calculated from the training data. For MAP, the prior parameters are $\bar{a}_{ii} = \bar{a}_{i,i+1} = 2$, $\bar{a}_{ij} = 1$ for $j \neq i$ and $j \neq i + 1$, $\bar{p}_1 = 2$, $\bar{p}_{i\neq 1} = 1$, $m = \mu_{init}$, $s = 0.5$, $\zeta = \eta = \nu = \psi = 0.5$, $b = 0$, and $c = 1$. Half of the assumed cost of each sensor is used for k_l ($k_l = 1200$ for the force features and $k_l = 600$ for the vibration features). For MAP-beta, the hyperparameters are the same as for MAP, except $k_l = 1$ and κ_l is half the assumed cost of the sensor. The hyperparameters for VB are the same as in [137], where their value is chosen to be as non-informative as possible. The initial parameter values for the VB algorithm are set as close as possible to the initial values of the EM methods. The convergence threshold for this experiment is increased to 10^{-6} for all four algorithms.

The hyperparameters for MAP and MAP-beta are chosen based on intuition and domain knowledge. As mentioned earlier in the synthetic data experiments, $T/4$ is a good initial choice for the hyperparameters on ρ if no other prior information is available. For the PHM data, $T/4$ is roughly 400. However, the cost of the sensors is known. Given the $T/4$ heuristic, using the cost of sensors as the values for the hyperparameters would probably result in underestimating the relevance of features. Therefore, half of the assumed cost is used to bring the values closer to $T/4$.

An informative prior is also used for μ and ϵ. The data in this experiment is standardized; therefore, one can assume that irrelevant features follow a standard Gaussian distribution $\mathcal{N}(0, 1)$. The mean of the data in the training set is close to 0, so it was used as the mean of the prior on ϵ. Using similar logic and the knowledge that the data is standardized, it is reasonable to assume that the means of the state-dependent distributions should be roughly evenly spaced from -2 to 2 and ascending. Thus, the initial value for μ is used for m. All other hyperparameters are chosen to minimize their effect on the posterior. This logic is used for selecting the hyperparameters for all PHM experiments in this dissertation.

Tables 3.7 to 3.9 contain the RMSE values for the full and reduced models, the $\bar{\rho}$ for the removed sensor, and the sensor removed during feature selection. Figures 3.4 to 3.6 display plots of the true wear values, the estimated wear values for the full models, and the estimated wear values for the reduced models.

Removing a sensor decreases average RMSE in only 4 of the 12 tests. However, when a sensor is removed, the average RMSE increases more than 2 μm in only two tests (VB using 20 states and MAP-beta using 10 states). The largest increase in RMSE for the reduced model occurs for the MAP-beta test on Tool 6 using 10 states. We believe this is due to the model underestimating relevant features resulting in skewed parameter estimates.

In the experiments using 20 states, ML yields the lowest average RMSE for the full (21.81 μm) and reduced models (22.14 μm). In the experiments using 10 states, VB yields the lowest average RMSE for the full (20.77 μm) and reduced models (21.52 μm). In the experiments using 5 states, MAP yields the lowest average RMSE for the full model (24.07 μm) and MAP-beta yields the lowest for the reduced model (24.59 μm). For all three assumed number of states, MAP produces full and reduced models with average RMSE below 27 μm while the results for

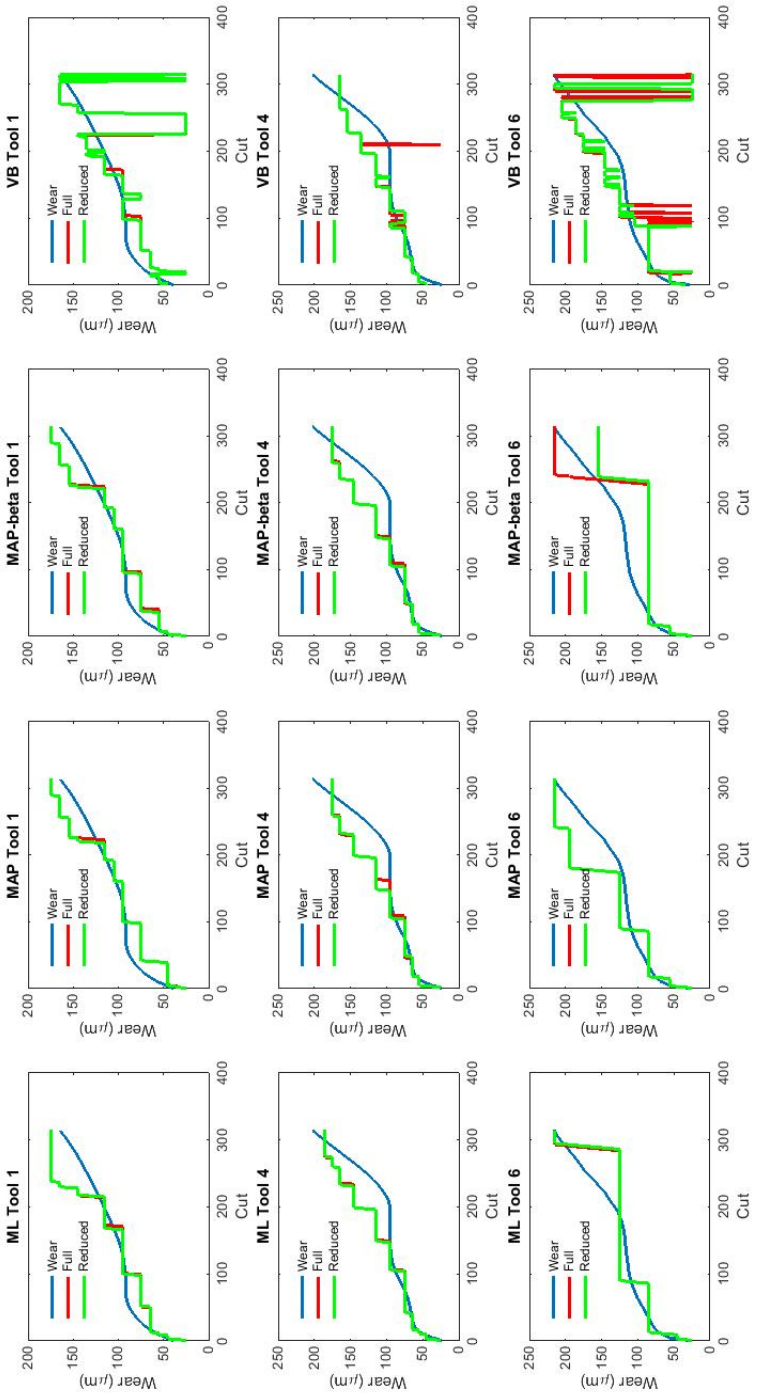

FIGURE 3.4: Plot for 20 state model. Wear is the true wear measurement, Full is the full model using 18 features, Reduced is the reduced model using 15 features. ML, MAP and MAP-beta predicts wear curves that are possible, while VB predicts impossible curves which can transition to a lower wear state.

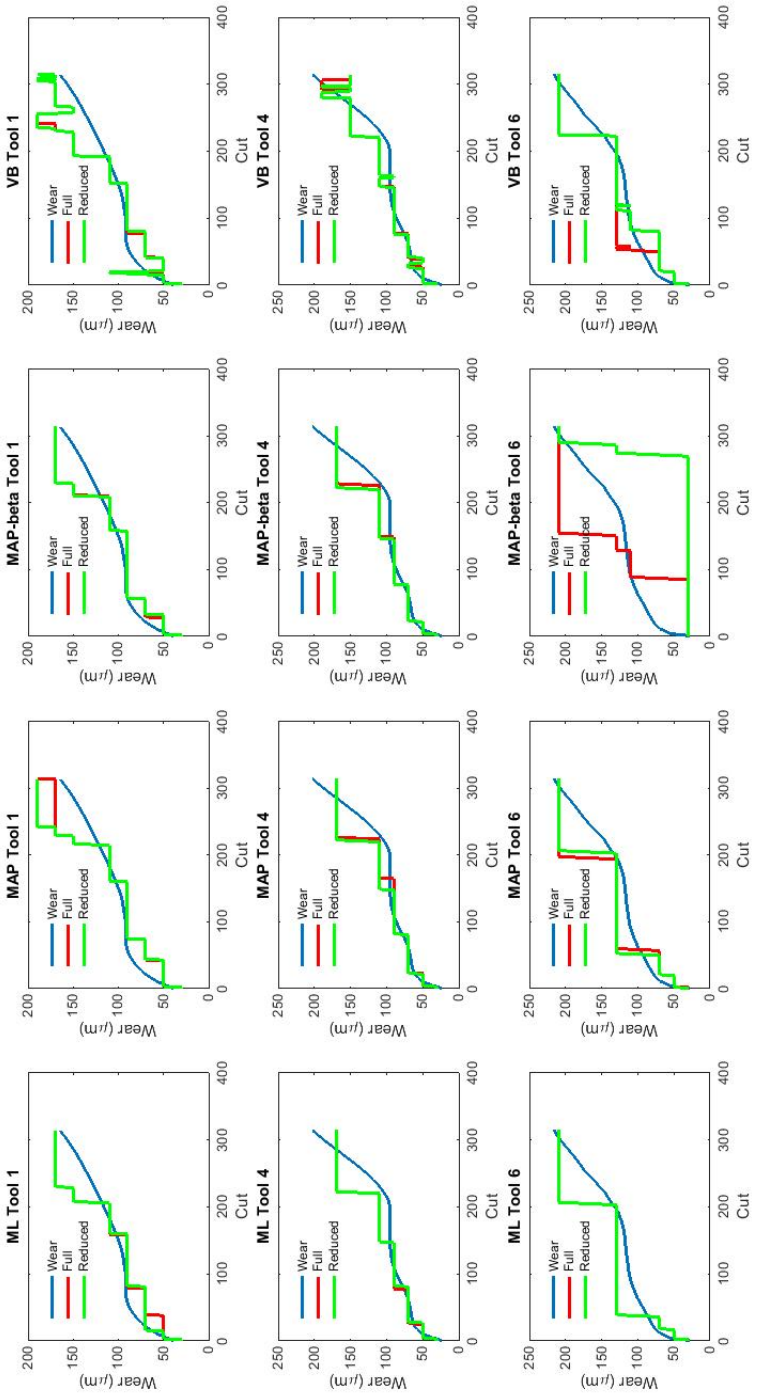

FIGURE 3.5: Plot for 10 state model. Wear is the true wear measurement, Full is the full model using 18 features, Reduced is the reduced model using 15 features. ML, MAP and MAP-beta predicts wear curves that are possible, while VB predicts impossible curves which can transition to a lower wear state.

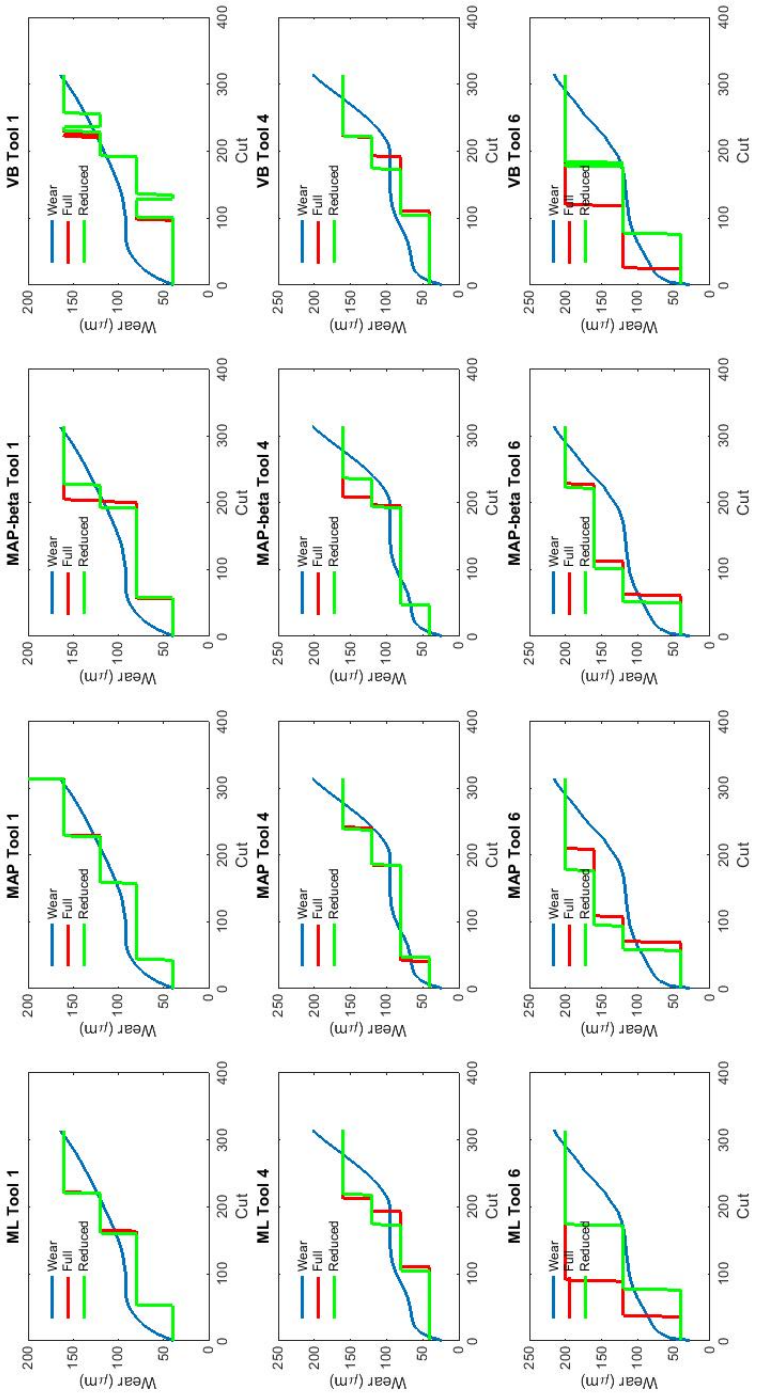

FIGURE 3.6: Plot for 5 state model. Wear is the true wear measurement, Full is the full model using 18 features, Reduced is the reduced model using 15 features. ML, MAP and MAP-beta predicts wear curves that are possible, while VB predicts impossible curves which can transition to a lower wear state.

Algorithm and Tool	RMSE Full Model (μm)	RMSE Reduced Model (μm)	Smallest $\bar{\rho}$	Removed Sensor
ML Tool 1	19.00	18.96	0.9700	Force Y
ML Tool 4	22.19	22.57	0.9676	Vibration Y
ML Tool 6	24.23	24.91	0.9672	Force X
Average	21.81 ± 2.64	22.14 ± 3.00		
MAP Tool 1	16.03	16.19	0.6482	Force Y
MAP Tool 4	22.46	22.64	0.6708	Force Y
MAP Tool 6	30.26	30.03	0.5919	Force Y
Average	22.92 ± 7.13	23.03 ± 6.93		
MAP-beta Tool 1	14.53	14.32	0.3579	Force Y
MAP-beta Tool 4	22.10	22.25	0.3751	Force Y
MAP-beta Tool 6	31.06	34.25	0.3685	Force Y
Average	22.56 ± 8.27	23.61 ± 10.03		
VB Tool 1	38.57	38.10	0.9844	Force Y
VB Tool 4	20.13	19.98	0.9815	Force X
VB Tool 6	51.99	60.03	0.9727	Vibration Y
Average	36.90 ± 16.00	39.68 ± 20.06		

TABLE 3.7: Results for 20 state PHM experiments. MAP and MAP-beta consistently select force in the Y direction for removal, which is the more expensive sensor. ML and VB, which do not consider cost, select varying sensors for removal. The average RMSE and ± 1 standard deviation are given for each formulation.

ML, MAP-beta and VB vary depending on the number of states. For this application when only prediction accuracy is considered, the ML, MAP and VB are essentially interchangeable depending on the initial assumptions, while MAP-beta gives poor accuracy results for the 10 state models. The formulations using the cost of features distinguish themselves during the feature selection process.

The MAP-beta formulation gives the smallest estimated feature saliency for all features. The estimated feature saliencies for the MAP algorithm are smaller than either ML or VB but larger than MAP-beta. This is expected due to the use of the prior, and the fact that we have previously established that MAP-beta can underestimate ρ. For ML and VB, the lack of a prior results in ρ for each sensor greater than 0.75. In general, these ρ close to 1 would indicate that the feature is relevant and should not be removed from the model. However, our goal is to remove one sensor in order to reduce the cost of the sensor setup. The total cost of all six sensors is \$10,800. When a vibration sensor is removed, the total cost of the remaining sensors is \$9600. When a force sensor is removed, the total cost is \$8400. For all the MAP and MAP-beta tests, the force in the Y direction

Algorithm and Tool	RMSE Full Model (μm)	RMSE Reduced Model (μm)	Smallest $\bar{\rho}$	Removed Sensor
ML Tool 1	19.24	18.30	0.9483	Vibration Y
ML Tool 4	21.15	21.13	0.9376	Vibration Y
ML Tool 6	28.87	28.87	0.9410	Force Y
Average	23.09 ± 5.10	22.77 ± 5.47		
MAP Tool 1	18.87	25.73	0.4761	Force Y
MAP Tool 4	19.52	21.08	0.5708	Force Y
MAP Tool 6	30.78	28.06	0.5539	Force Y
Average	23.06 ± 6.70	24.95 ± 3.55		
MAP-beta Tool 1	17.42	17.74	0.3251	Force Y
MAP-beta Tool 4	19.15	21.14	0.3171	Force Y
MAP-beta Tool 6	52.93	88.16	0.3022	Force Y
Average	29.83 ± 20.02	42.35 ± 39.71		
VB Tool 1	22.47	23.59	0.9102	Vibration Y
VB Tool 4	17.01	18.50	0.9093	Force X
VB Tool 6	22.79	22.47	0.8960	Vibration Y
Average	20.77 ± 3.25	21.52 ± 2.67		

TABLE 3.8: Results for 10 state PHM experiments. MAP and MAP-beta consistently removes force in the Y direction, which is the more expensive sensor. VB and ML, which do not consider cost, select varying sensors for removal. The average RMSE and ± 1 standard deviation are given for each formulation.

is removed during feature selection. Further, both MAP algorithms consistently remove the same sensor when the training set is changed. The removed sensor when using the ML and VB algorithms vary when the number of states and the training set are changed. However, the removed sensor does stabilize for ML and VB when the number of states is reduced to 5. MAP and MAP-beta consistently yield a reduced model with the lowest total cost of sensors. The ML and VB algorithms do not give a clear indication of which sensor should be purchased, while MAP consistently indicates that Force in the Y direction can be eliminated from the sensor setup. The ML and VB algorithms suggests three different sensor can be removed depending on the number of states and training sets.

As the number of states decreases, the RMSE generally increases indicating that the models with 20 states fit the data better. We increase the number of states from 5 to 20 in order to demonstrate the effect of more states on ρ. The estimated saliencies increase with the number of states, because, in general, the probability of a series of observations coming from a multi-modal Gaussian distribution (the state-dependent distribution) is greater than the probability of them coming from a single Gaussian (the state-independent distribution). The priors force ρ_l towards

Algorithm and Tool	RMSE Full Model (μm)	RMSE Reduced Model (μm)	Smallest $\bar{\rho}$	Removed Sensor
ML Tool 1	20.44	20.45	0.7774	Vibration Y
ML Tool 4	29.07	27.64	0.8499	Vibration Y
ML Tool 6	55.14	38.75	0.8379	Vibration Y
Average	34.88 ± 18.07	28.95 ± 9.22		
MAP Tool 1	17.73	17.93	0.2037	Force Y
MAP Tool 4	18.14	18.84	0.1947	Force Y
MAP Tool 6	36.33	41.37	0.3273	Force Y
Average	24.07 ± 10.62	26.05 ± 13.28		
MAP-beta Tool 1	24.87	22.21	0.2434	Force Y
MAP-beta Tool 4	24.58	18.92	0.1961	Force Y
MAP-beta Tool 6	32.14	32.64	0.1953	Force Y
Average	27.20 ± 4.28	24.59 ± 7.16		
VB Tool 1	29.11	29.34	0.8601	Vibration Y
VB Tool 4	27.52	26.95	0.8642	Vibration Y
VB Tool 6	48.07	37.42	0.8941	Vibration Y
Average	34.90 ± 11.43	31.24 ± 5.49		

TABLE 3.9: Results for 5 state PHM experiments. MAP and MAP-beta consistently removes force in the Y direction, which is the more expensive sensor. ML and VB, which do not consider cost, remove a less expensive sensor, vibration in the Y direction. The average RMSE and ± 1 standard deviation are given for each formulation.

0 and help discriminate between relevant and irrelevant features. This is further support for using a MAP formulation over ML or VB when modeling HMMs with a larger state space.

The VB algorithm does not estimate a left-to-right Markov chain. During testing, the VB models can predict decreasing wear estimates (see Figures 3.4 to 3.6). A proposed advantage of VB is the ability to estimate the number of states. In this experiment, the estimated number of states by VB is the same as the initial number of states.

3.5.1 PHM Data using Full Conditional with Viterbi

As discussed earlier, there are two possible choices for $p(y_t|x_t = i, \Lambda)$ during prediction. In this section, results using the full conditional distribution are given. The training procedure is the same as in the previous section, the only difference is how the conditional likelihood is calculated for the Viterbi algorithm.

Algorithm and Tool	RMSE Full Model (μm)	RMSE Reduced Model (μm)	Smallest $\bar{\rho}$	Removed Feature
ML Tool 1	13.02	12.88	0.9700	Force Y
ML Tool 4	22.85	23.34	0.9676	Vibration Y
ML Tool 6	31.78	34.28	0.9672	Force X
Average	22.55 ± 9.38	23.50 ± 10.70		
MAP Tool 1	14.32	14.19	0.6482	Force Y
MAP Tool 4	23.66	23.68	0.6708	Force Y
MAP Tool 6	31.67	32.04	0.5919	Force Y
Average	23.22 ± 8.68	23.30 ± 8.93		
MAP-beta Tool 1	12.32	11.95	0.6482	Force Y
MAP-beta Tool 4	23.54	23.65	0.6708	Force Y
MAP-beta Tool 6	34.34	34.33	0.5919	Force Y
Average	23.40 ± 11.01	23.31 ± 11.19		
VB Tool 1	35.91	33.11	0.9844	Force Y
VB Tool 4	19.82	20.22	0.9815	Force X
VB Tool 6	21.01	33.77	0.9727	Vibration Y
Average	25.58 ± 8.97	29.03 ± 7.64		

TABLE 3.10: Results for 20 state PHM experiments using the full conditional distribution. MAP and MAP-beta consistently removes force in the Y direction, which is the more expensive sensor. VB and ML, which do not consider cost, select varying sensors for removal. The average RMSE and ± 1 standard deviation are given for each formulation.

Tables 3.10 to 3.12 contain the testing results when the full conditional distribution is used for calculating $p(y_t|x_t = i, \Lambda)$ in the Viterbi algorithm.

For the 20 state models, the results for the VB algorithm drastically improve due to a decrease in RMSE for Tool 6. The mean RMSE for ML and MAP increase, but only by a small margin. The RMSE for the full MAP-beta model increases, while the RMSE for the reduced model decreases. For ML, MAP and VB, the full models outperform the reduced models, while the reduced model for MAP-beta slightly outperforms the full model.

For the 10 state models, the mean of the RMSE over all test tools and algorithms improves. The mean RMSE for MAP-beta improves significantly. The full models outperform the reduced models but by a small margin.

For the 5 state models, the VB algorithm performs better than when using the reduced conditional distribution. The MAP and MAP-beta algorithms perform worse than when the reduced conditional distribution is used. The ML algorithm performs better on the full model but worse on the reduced model. The reduced

Algorithm and Tool	RMSE Full Model (μm)	RMSE Reduced Model (μm)	Smallest $\bar{\rho}$	Removed Feature
ML Tool 1	15.28	15.27	0.9483	Vibration Y
ML Tool 4	21.16	21.15	0.9376	Vibration Y
ML Tool 6	30.03	29.80	0.9410	Force Y
Average	22.16 ± 7.43	22.07 ± 7.31		
MAP Tool 1	15.28	14.11	0.4761	Force Y
MAP Tool 4	21.16	21.09	0.5708	Force Y
MAP Tool 6	30.03	31.65	0.5539	Force Y
Average	22.16 ± 7.43	22.28 ± 8.83		
MAP-beta Tool 1	13.66	14.07	0.6482	Force Y
MAP-beta Tool 4	21.16	21.33	0.6708	Force Y
MAP-beta Tool 6	34.72	34.51	0.5919	Force Y
Average	23.18 ± 10.67	23.30 ± 10.36		
VB Tool 1	17.56	17.60	0.9102	Vibration Y
VB Tool 4	17.01	18.50	0.9093	Force X
VB Tool 6	25.46	24.74	0.8960	Vibration Y
Average	20.01 ± 4.73	20.28 ± 3.89		

TABLE 3.11: Results for 10 state PHM experiments using the full conditional distribution. MAP and MAP-beta consistently removes force in the Y direction, which is the more expensive sensor. VB and ML, which do not consider cost, select varying sensors for removal. The average RMSE and ± 1 standard deviation are given for each formulation.

models for ML, MAP and VB perform better than the full model, while the full model for MAP-beta outperforms the reduced model.

After examining these results, it is unclear which conditional distribution is preferable. The reduced conditional offers the advantage of less computation during prediction and fewer parameter need to be stored. This benefit could be useful if the FSHMM was used in a commercial product. We will continue using the reduced conditional for testing, but concede that the full conditional could yield better predictive accuracy in cases where ρ is not close to 1 for features included in the reduced model.

As expected, the difference between the two conditional distributions is most apparent in the 5 state models due to the smaller values for ρ.

Algorithm and Tool	RMSE Full Model (μm)	RMSE Reduced Model (μm)	Smallest $\bar{\rho}$	Removed Feature
ML Tool 1	20.54	20.46	0.7774	Vibration Y
ML Tool 4	28.23	27.64	0.8499	Vibration Y
ML Tool 6	40.63	36.31	0.8379	Vibration Y
Average	29.80 ± 10.14	28.14 ± 7.94		
MAP Tool 1	17.88	17.88	0.2037	Force Y
MAP Tool 4	19.54	19.54	0.1947	Force Y
MAP Tool 6	45.82	44.76	0.3273	Force Y
Average	27.74 ± 15.67	27.39 ± 15.06		
MAP-beta Tool 1	21.18	21.18	0.6482	Force Y
MAP-beta Tool 4	22.16	22.16	0.6708	Force Y
MAP-beta Tool 6	40.20	40.48	0.5919	Force Y
Average	27.85 ± 10.71	27.94 ± 10.87		
VB Tool 1	28.56	28.98	0.8601	Vibration Y
VB Tool 4	27.51	26.14	0.8642	Vibration Y
VB Tool 6	33.65	33.97	0.8941	Vibration Y
Average	29.91 ± 3.28	29.70 ± 3.96		

TABLE 3.12: Results for 5 state PHM experiments using the full conditional distribution. MAP and MAP-beta consistently removes force in the Y direction, which is the more expensive sensor. VB and ML, which do not consider cost, select vibration in the Y direction for removal. The average RMSE and ± 1 standard deviation are given for each formulation.

3.6 Microsoft Kinect Data

Experiments are also performed on a data set containing Microsoft Kinect data [2]. This data was collected by observing a worker engaged in a painting process in a manufacturing setting. For more details on the experimental setup and data collection, see [101]. This painting process can be described as a sequence of six tasks, which are labeled as 'Fetch', 'Paint', 'Dry', 'Load', 'Walk' and 'Other'. These tasks are performed repeatedly by the worker. The objective is to infer the task (the hidden state) that the worker is engaged in at each time step from body position observations collected by the Kinect. As observable features, the Kinect records the X, Y, and Z coordinates of ten upper body joints. Specifically, the Kinect records the positions of ten joints on the body labeled 'Head', 'Shoulder Center', 'Shoulder Left', 'Elbow Left', 'Wrist Left', 'Hand Left', 'Shoulder Right', 'Elbow Right', 'Wrist Right', and 'Hand Right'. The set is composed of observations collected over the course of one hour. The first two-thirds of the data are used for training the models and the last third is reserved for testing the accuracy of our

[2] http://people.virginia.edu/ djr7m/incom2015/

estimated models. The first 2000 observations of the training set are are used to calculate initial parameter estimates. The task being performed by the worker at each time step is actually known, but is only used for calculating the initial parameter estimates and validation. The true tasks are not used when training the models using EM or VB.

All four algorithms are initialized with the same values. The state-dependent parameters $p(\cdot|\cdot)$ are initialized by calculating the state-dependent mean and standard deviation from the initialization set. The parameters for $q(\cdot|\cdot)$ are the means and standard deviations of the training data. The transition probabilities are initialized by counting the number of transition in the initialization set. The MAP hyperparameters are $\bar{a} = 1, \bar{p} = 1, m = \mu_{init}, s = 0.25, \zeta = 0.25, \eta = 1, b = \epsilon_{init}, c = 0.5, \nu = 0.25, \psi = 0.5, k = 15,000$. The MAP-beta hyperparameters are the same as MAP, except $k_l = 1$ and $\kappa_l = 15,000$. The cost of collecting each feature is the same for this data set. Therefore, the hyperparameters on ρ in MAP and MAP-beta are used to penalize larger feature subsets. The VB parameters are the same used in the previous section. The convergence threshold is 10^{-6}.

The hyperparameters for the feature saliencies are chosen using the $T/4$ heuristic. Informative priors are also used for μ and ϵ. The mean of the prior on μ is set to the initial value of μ, which is calculated from the supervised initialization set. It is logical to assume that the estimated values for μ should be close to the value calculated on a small supervised set of the data. The mean of the prior on ϵ is set to the mean of all the data. Irrelevant features should follow a Gaussian distribution with a mean close to the grand mean of the data. Similar intuition is used for selecting hyperparameters for all Kinect data experiments.

Three validation experiments are performed. In each experiment, the four algorithms are used to estimate model parameters. In the first experiment, all 30 features are treated as relevant, and the classification accuracy of the four algorithms is calculated on a test set. In the second experiment, features with estimated ρ below a given threshold are removed, and the classification accuracy on the test set is calculated for each reduced model. In the third experiment, we sequentially remove features based on the estimated ρ, and calculate classification accuracy for these reduced models. The test set is composed of approximately 20 minutes worth of data sampled at a rate of 30 frames per second, resulting in 30,102 time steps. Using the models obtained by each of the algorithms, the task being performed in each time step is inferred, and compared with the known task in that

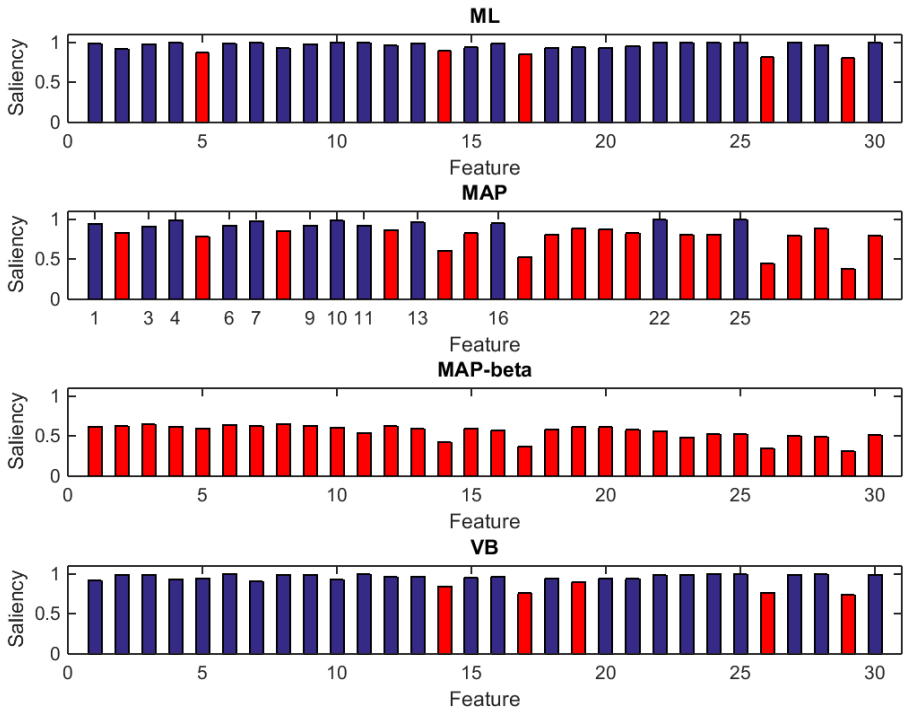

FIGURE 3.7: Blue bars indicate saliences ≥ 0.9 and red bars indicate saliences < 0.9. Features with ρ less than 0.9 are removed from the model during feature selection.

time step. The Viterbi algorithm using Equation 3.36 is implemented for task prediction. The fraction of correctly classified states is used to measure accuracy. This testing procedure only yields a point estimate for the accuracy of the model.

In the first experiment, the fractions of correctly classified states on the test set for ML, MAP, MAP-beta and VB are 0.7572, 0.7473, 0.7590 and 0.6415. ML, MAP and MAP-beta outperform VB by more than 10%. The EM formulations perform similarly in terms of classification accuracy.

In the second experiment, a removal threshold of 0.9 is chosen. Then features with estimated ρ below 0.9 are removed from the model during feature selection. Figure 3.7 displays plots of ρ, where features below the threshold are marked in red. Using a truncated exponential prior with support on $[0, 1]$ in the MAP formulation forces ρ estimates towards zero, and allows k to be chosen so the features can distinguish themselves between relevant and irrelevant. All estimated saliencies using MAP-beta are below the 0.9 threshold. ML and VB tend to assign higher ρ making it more difficult to perform feature selection. The fraction

of correctly classified states for the reduced ML model is 0.7600, the reduced MAP model is 0.7606, and the reduced VB model is 0.6561. The fraction of correctly classified states for the reduced MAP-beta model is 0.4964 which is to be expected as no features are used in the estimation. The model predicts every observation as 'Paint' and the classification accuracy reflects the proportion of 'Paint' in the test set. For a better comparison, we remove features with estimated ρ below 0.5 and retest MAP-beta. This yields a fraction of correctly classified states of 0.7480. The largest estimated ρ for MAP-beta is 0.6525, while the smallest for VB and ML are 0.7402 and 0.8068. Therefore, we cannot select a single removal threshold that is acceptable for every model.

The reduced ML and VB models each remove 5 features, while the reduced MAP model removes 18 features. The reduced models for theses three algorithms have improved accuracy over their corresponding full model. When the removal threshold is lowered to 0.5 for MAP-beta, only 6 features are removed and the accuracy decreased compared to the full model.

In the third experiment, features are removed sequentially based on the estimated saliencies. More specifically, for the single removed feature models, the feature with the lowest estimated ρ is removed and tested. For the models removing two features, the two features with the lowest estimated saliencies are removed and the models are tested. The process continues until all features have been removed and tested resulting in 30 reduced models per algorithm. The results are displayed in Figure 3.8.

For this third experiment, we see that the EM based algorithms dominate the VB algorithm until over 25 features are removed from the model. At this point, the algorithms' accuracies converge and begin to drop dramatically. When focusing on the EM algorithms, ML has a higher accuracy than MAP and MAP-beta until more than 25 features are removed. MAP and MAP-beta yield similar results with MAP-beta outperforming MAP in 16 of the 30 reduced models. However, starting at removing 24 features, the accuracy of MAP-beta drops to around 66%. At this point, under estimating the relevance of relevant features starts to affect the performance of the algorithm. ML and MAP do not have this problem and continue with performance measurements above 70% until a few more features are removed.

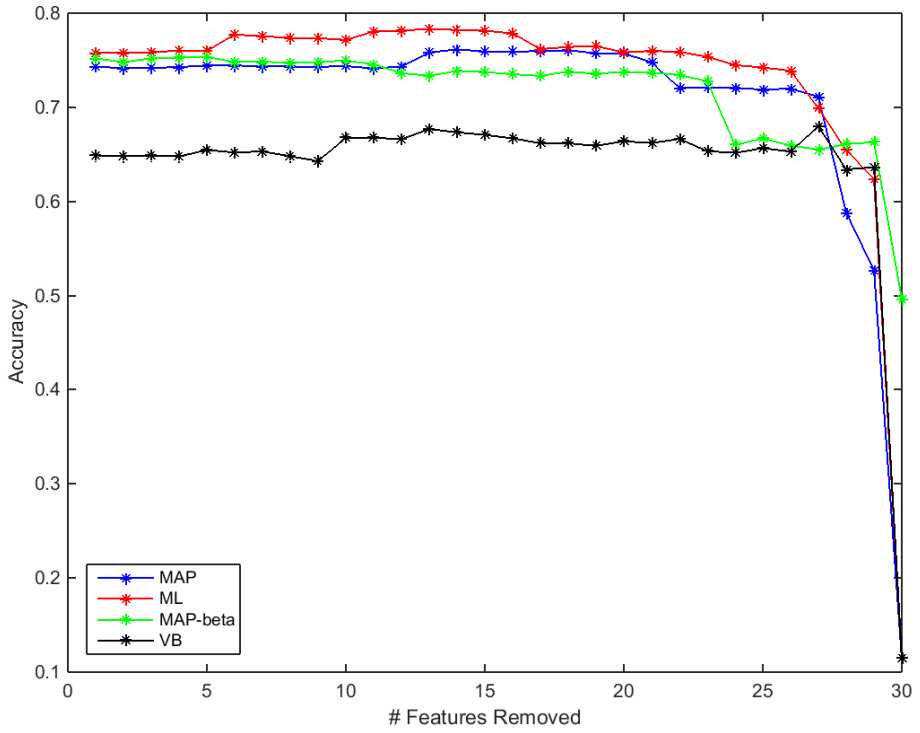

FIGURE 3.8: Sequential feature removal results for ML, MAP, MAP-beta and VB algorithms. The EM based methods have a higher accuracy than the VB method until more than 25 features are removed.

Table 3.13 contains the order the features are removed for each algorithm. It is clear that each algorithm removes features in a different order; however, the first feature removed is always the 'Right Hand' in the Y direction. Further, when the same feature subset is used for testing, as in the first model and the modeling removing 3 features, the EM based algorithms outperform the VB algorithm. This indicates that the EM algorithms in general give better parameter estimates than the VB algorithm for this data set.

Again, the VB algorithm does not allows for certain transition of the Markov chain to be 0. For this implementation, specific transition between tasks could be impossible. For instance, a worker might never go from 'Paint' to 'Load'. The VB algorithm could allow for this transition, but if this transition never happens in the training set, the ML and MAP algorithms will set this transition to 0. It should be noted that MAP only estimates transitions that appear in the training set because the hyperparameter is set to 1. If $\bar{a}_{ij}$ is set greater than 1 for all possible transitions, the algorithm will estimate a small transition probability for transitions that do not occur in the training set.

# Removed	ML	MAP	MAP-beta	VB
1	Right Hand Y	Right Hand Y	Right Hand Y	Right Hand Y
2	Right Wrist Y	Right Wrist Y	Right Wrist Y	Left Hand Y
3	Left Hand Y	Left Hand Y	Left Hand Y	Right Wrist Y
4	Center Shoulder Y	Left Wrist Y	Left Wrist Y	Left Wrist Y
5	Left Wrist Y	Center Shoulder Y	Right Elbow Y	Right Shoulder X
6	Head Y	Right Wrist Z	Hand Right X	Left Shoulder X
7	Left Shoulder Y	Right Hand Z	Right Wrist Z	Head X
8	Left Hand Z	Right Elbow Y	Right Hand Z	Left Elbow X
9	Right Shoulder Y	Left Hand Z	Right Elbow Z	Center Shoulder X
10	Left Wrist Z	Right Elbow Z	Right Wrist X	Right Shoulder Z
11	Right Shoulder X	Left Wrist Z	Left Elbow Y	Left Hand Z
12	Right Shoulder Z	Right Shoulder Z	Right Elbow X	Right Shoulder Y
13	Left Elbow Z	Head Y	Left Hand X	Center Shoulder Y
14	Right Hand X	Left Shoulder Y	Right Shoulder Z	Left Wrist Z
15	Head Z	Left Elbow Z	Left Hand Z	Left Elbow Z
16	Left Shoulder Z	Right Shoulder Y	Left Wrist X	Left Wrist X
17	Center Shoulder Z	Right Hand X	Left Wrist Z	Left Hand X
18	Head X	Right Shoulder X	Center Shoulder Y	Right Elbow X
19	Left Hand X	Head Z	Left Elbow X	Right Hand Z
20	Left Wrist X	Left Elbow Y	Right Shoulder Y	Head Z
21	Left Shoulder X	Left Shoulder Z	Right Shoulder X	Head Y
22	Right Elbow Y	Center Shoulder Z	Center Shoulder X	Right Elbow Y
23	Left Elbow Y	Head X	Head X	Left Shoulder Z
24	Center Shoulder X	Left Hand X	Shoulder Left X	Left Shoulder Y
25	Right Hand Z	Left Wrist X	Left Elbow Z	Right Wrist Z
26	Right Elbow Z	Left Shoulder X	Head Y	Right Wrist X
27	Left Elbow X	Left Elbow X	Left Shoulder Z	Left Elbow Y
28	Right Wrist Z	Center Shoulder X	Center Shoulder Z	Right Hand X
29	Right Elbow X	Right Elbow X	Head Z	Right Elbow Z
30	Right Wrist X	Right Wrist X	Left Shoulder Y	Center Shoulder Z

TABLE 3.13: Order of features removed during sequential reduced model testing. Each formulation removes features in a different order.

3.7 Highly Correlated Features

The FSHMM assumes that each feature is independent. In practice, it is highly unlikely that there is zero correlation between all features. Removing redundant features is typically a desirable property of a feature selection algorithm. Once the information is given to the classifier by a single relevant feature, the redundant features are not needed and do not provide a benefit. In this section, we give an argument for including redundant features when using HMMs. Redundant features actually improve the accuracy of an HMM. Then it is shown how the FSHMM performs in the presence of redundant features.

This section gives an example specific to HMMs; however, Guyon and Elisseeff [44] give an example for redundant features improving the accuracy for general classifiers.

3.7.1 Redundant Features and HMMs

First, we demonstrate how the conditional likelihood $p(y_t|x_t = i)$ is affected by redundant features. The third sequence of the synthetic data from Section 3.4 is used in these experiments.

Observation	True State	$p(y_t\|x_t = 1)$	$p(y_t\|x_t = 2)$	$p(y_t\|x_t = 1)/$ $p(y_t\|x_t = 2)$
1	1	0.0772	9.3654×10^{-6}	8.2482×10^{3}
2	1	0.0779	6.2922×10^{-5}	1.2378×10^{3}
3	1	0.0460	0.0010	44.7527
4	1	0.0702	1.7752×10^{-4}	395.6814
5	2	4.5821×10^{-4}	0.0585	0.0078
6	1	0.0721	4.0135×10^{-6}	1.7973×10^{4}
7	2	1.2691×10^{-4}	0.0734	0.0017
8	2	1.0721×10^{-7}	0.0388	2.7606×10^{-6}
9	2	1.9549×10^{-5}	0.0795	2.4575×10^{-4}
10	2	3.5797×10^{-5}	0.0796	4.4985×10^{-4}

TABLE 3.14: Conditional likelihoods and ratios for a single feature. Only first 10 observations are displayed.

Observation	True State	$p(y_t\|x_t = 1)$	$p(y_t\|x_t = 2)$	$p(y_t\|x_t = 1)/$ $p(y_t\|x_t = 2)$
1	1	0.0060	8.7710×10^{-11}	6.8032×10^{7}
2	1	0.0061	3.9592×10^{-9}	1.5322×10^{6}
3	1	0.0021	1.0561×10^{-6}	2.0028×10^{3}
4	1	0.0049	3.1514^{-8}	1.5656×10^{5}
5	2	2.0996×10^{-7}	0.0034	6.1326×10^{-5}
6	1	0.0052	1.6108×10^{-11}	3.2301×10^{8}
7	2	1.6106×10^{-8}	0.0054	2.9933×10^{-6}
8	2	1.1494×10^{-14}	0.0015	7.6207×10^{-12}
9	2	3.8217×10^{-10}	0.0063	6.0395×10^{-8}
10	2	1.2814×10^{-9}	0.0063	2.0237×10^{-7}

TABLE 3.15: Conditional likelihoods and ratios for two redundant features. Only first 10 observations are displayed. Two redundant features decrease the likelihood but increase the ratio between the two states.

The conditional likelihood using the first feature is calculated for each state. The true parameter values are used for these calculations. The first ten observations are in Table 3.14, and these results are consistent when all 500 observations are examined.

As one would expect, the conditional likelihood is larger for the true state. To create redundant features, the first feature is copied and used as the second and third features. The conditional likelihoods for two and three redundant features are in Tables 3.15 and 3.16.

Because the conditional likelihood is a product when features are considered independent, the conditional likelihood has an exponential relationship with the

Observation	True State	$p(y_t\|x_t = 1)$	$p(y_t\|x_t = 2)$	$p(y_t\|x_t = 1)/$ $p(y_t\|x_t = 2)$
1	1	4.6094×10^{-4}	8.2144×10^{-16}	5.6114×10^{11}
2	1	4.7246×10^{-4}	2.4912×10^{-13}	1.8965×10^{9}
3	1	9.7275×10^{-5}	1.0853×10^{-9}	8.9631×10^{4}
4	1	3.4657×10^{-4}	5.5943×10^{-12}	6.1949×10^{7}
5	2	9.6206×10^{-11}	2.0033×10^{-4}	4.8025×10^{-7}
6	1	3.7532×10^{-4}	6.4650×10^{-17}	5.8054×10^{12}
7	2	2.0440×10^{-12}	3.9469×10^{-4}	5.1788×10^{-9}
8	2	1.2323×10^{-21}	5.8575×10^{-5}	2.1037×10^{-17}
9	2	7.4710×10^{-15}	5.0336×10^{-4}	1.4842×10^{-11}
10	2	4.5870×10^{-14}	5.0387×10^{-4}	9.1037×10^{-11}

TABLE 3.16: Conditional likelihoods and ratios for three redundant features. Only first 10 observations are displayed. Three redundant features decrease the likelihood but increase the ratio between the two states.

likelihood of each redundant feature $p(y_t|x_t = i) = p(y_{lt}|x_t = i)^n$, where n is the number of redundant features. The ratios between the conditional likelihood for each state are given in the final column of Tables 3.14, 3.15, and 3.16. As the number of redundant features increases, the ratios between the conditional likelihoods increase when the true state is 1 and decrease when the true state is 2. This indicates that the discriminating power increases as more redundant features are added to the model.

The posterior distribution $p(x_t|\mathbf{y}) = \gamma_t(i)$ (Equation 2.12) can be used to further illustrate this point. As above, the true parameters are used in the calculations. The posterior probabilities for the first 10 observations of the first sequences are displayed in Tables 3.17, 3.18, and 3.19. The probability for the correct state increases as the number of redundant features increases. We can conclude that redundant relevant features lead to more accurate parameter estimates during training and more accurate state estimates during prediction.

A logical conclusion from this example is that adding more features, not necessarily relevant features, will improve the model. We test this theory by performing the posterior distribution experiment using the first relevant feature from the synthetic data set and the last irrelevant feature (random noise) from the synthetic data set. For the irrelevant feature, we use the following parameters: $\mu_1 = -1, \mu_2 = 1, \sigma_1 = \sigma_2 = 0.5$. The results are in Table 3.20.

When an irrelevant feature is used, the predictions are not as accurate as when only relevant features are used. For the tests using a single relevant features, state

Observation	True State	$p(x_t = 1\|y)$	$p(x_t = 2\|y)$
1	1	0.9999	9.7031×10^{-5}
2	1	0.9999	1.4565×10^{-4}
3	1	0.9960	0.0040
4	1	0.9980	0.0020
5	2	0.0421	0.9579
6	1	0.9998	1.9338×10^{-4}
7	2	0.0022	0.9978
8	2	7.7280×10^{-7}	1.0000
9	2	6.8290×10^{-5}	0.9999
10	2	1.2501×10^{-4}	0.9999

TABLE 3.17: Posterior distribution using a single feature. Only first 10 observations are displayed.

Observation	True State	$p(x_t = 1\|y)$	$p(x_t = 2\|y)$
1	1	1.0000	1.1759×10^{-8}
2	1	1.0000	1.1607×10^{-7}
3	1	0.9999	8.8758×10^{-5}
4	1	1.0000	5.1099×10^{-6}
5	2	3.4484×10^{-4}	0.9997
6	1	1.0000	1.1142×10^{-8}
7	2	3.7417×10^{-6}	1.0000
8	2	2.1169×10^{-12}	1.0000
9	2	1.6776×10^{-8}	1.0000
10	2	5.6214×10^{-8}	1.0000

TABLE 3.18: Posterior distribution using two redundant features. Only first 10 observations are displayed. Two redundant features increase the posterior probability for the correct state.

Observation	True State	$p(x_t = 1\|y)$	$p(x_t = 2\|y)$
1	1	1.0000	1.4257×10^{-12}
2	1	1.0000	9.3741×10^{-11}
3	1	1.0000	1.9834×10^{-6}
4	1	1.0000	1.2914×10^{-8}
5	2	2.7014×10^{-6}	1.0000
6	1	1.0000	6.2011×10^{-13}
7	2	6.4736×10^{-9}	1.0000
8	2	5.8437×10^{-18}	1.0000
9	2	4.1228×10^{-12}	1.0000
10	2	2.5288×10^{-11}	1.0000

TABLE 3.19: Posterior distribution using three redundant features. Only first 10 observations are displayed. Three redundant features increase the posterior probability for the correct state.

Observation	True State	$p(x_t = 1\|y)$	$p(x_t = 2\|y)$
1	1	0.9826	0.0174
2	1	0.7079	0.2921
3	**1**	**0.2057**	**0.7943**
4	**1**	**0.3086**	**0.6914**
5	2	2.5048×10^{-6}	1.0000
6	1	1.0000	1.3744×10^{-8}
7	2	1.5026×10^{-6}	1.0000
8	2	0.4939	0.5061
9	2	1.2148×10^{-4}	0.9999
10	2	9.1461×10^{-6}	1.0000

TABLE 3.20: Posterior distribution using one relevant feature and one irrelevant feature. Only first 10 observations are displayed. Bold indicates incorrect state prediction based on posterior probability. Irrelevant features increase the number of incorrectly classified states.

predictions based on the posterior probabilities yield an error rate of 0.0160, while adding an irrelevant feature increases the error to 0.1680.

From this experiment, we can conclude that more features are not always better. Relevant redundant features can increase the accuracy of the HMM, but irrelevant features can significantly decrease the accuracy. For HMMs, a feature selection method that eliminates redundant features is not required; however, in some cases, removing redundant features could be a desirable property.

3.7.2 FSHMM and Redundant Features

For this set of experiments, the FSHMM is applied to highly correlated synthetic data. The data is the same synthetic data used in all previous experiments. In these experiments, only the estimated feature saliencies are discussed.

For the first experiment, the first feature of the first sequence is replicated and used for the second and third features. ML and MAP with $k_l = 50$ are run on the redundant data. ML produces a saliency of 0.9868 for all three features and MAP a saliency of 0.9882 for all three features. When k_l is increased to 500 for the priors on the second and third feature ($k = [50\ 500\ 500]$),the estimated saliencies are [0.8752 0.1910 0.1910]. This indicates that k can be used to exclude redundant features, if it is known that the features are redundant before modeling.

For a second experiment, the second and third features are scaled by factors of 2 and 3. This changes the feature values but keeps the correlation between all features 1. ML and MAP are tested on the scaled data, and the MAP algorithm is run with $k_l = 50$. The saliency estimates for this experiment are [0.9579 0.9349 0.9315] for MAP and [1.0000 0.9868 1.0000] for ML. The MAP algorithm assigns lower saliencies to the features with larger range. The scaling has little effect on the ML algorithm.

For a final experiment, MAP is performed on the redundant data set with three identical features with random noise added to k. The expectation of this experiment is that the feature with the smallest weight will have the highest saliency estimate. k_l is drawn from a Gaussian distribution with a mean of 100 and a variance of 10. For this experiment, k = [87.9251 107.1724 116.3024] and the saliency estimates are [0.9378 0.9303 0.9264].

Even though we establish in the previous section that redundant features should not necessarily be removed during feature selection, this set of experiments demonstrates that the MAP formulation can deal with redundant features by adjusting k, if redundancy of features is known before modeling.

3.8 Conventional Feature Selection Comparison

In this section, we test conventional feature selection methods on the PHM and Kinect data sets. We evaluate sequential forward search (SFS), sequential backward search (SBS), an unsupervised feature similarity method and principal components analysis (PCA). For the sequential searches, both supervised and unsupervised evaluation functions are tested. PCA is not a FS technique, but rather a feature reduction or feature extraction method. The goal is to contrast widely used wrappers (SFS and SBS), filters (feature similarity and PCA) and embedded (FSHMM). It is demonstrated that these methods do not compare well to FSHMM on these data sets. The wrappers are time consuming to implement, require the choice of an evaluation function, do not perform well with unsupervised data, and can select different feature subsets based on the choice of evaluation function. The unsupervised filtering method does not allow for control over the size of the selected feature subset. PCA is an easier method to implement and less time consuming, but does not produce models as accurate as FSHMM. In the case

of the PHM data, PCA does not give insight into eliminating sensors, therefore the cost of the sensor setup cannot be reduced.

3.8.1 Sequential Methods

For the PHM data, a 5 state model is used. Twelve experiments are conducted using supervised SFS and SBS. Table 3.21 contains a summary for each experiment. In each experiment, a supervised tool is withheld for testing on the final model. In Experiment 1, the model is trained on the unsupervised tools and evaluated using the two remaining supervised tools. The evaluation function is the RMSE between the wear value and the median of the predicted state. A single feature is either added or removed at each step. In Experiment 2, the set of unsupervised tools and one of the remaining supervised tools is used for training the model at each step. The other remaining supervised tool is used for evaluation. The RMSE is again used as the evaluation function and a single feature is either added or removed at each step. Experiments 3 and 4 use the same setup as Experiments 1 and 2, but remove the three features corresponding to a single sensor at each step instead of a single feature. Experiment 5, 6, 7, and 8 have the same setup as the first four experiments, but uses the RMSE between the wear state and the predicted state. Experiment 9, 10, 11, and 12 have the same setup as the first four experiments, but use percent of correctly classified states as the evaluation function. These experiments demonstrate that the training set and the evaluation function can affect the accuracy of the final model and the selected feature subset.

Once a feature subset has been selected, a model for testing must be trained. We construct two possible models for the testing step. The first only uses the tools in the training set and does not use the tools in the evaluation set. The second model uses the tools in the training and evaluation set. For example, in Exp. 1, the first model would only use the unsupervised tools for training. The second model would use all the tools except the one withheld for testing.

When there are models at each step that produce the same value for the evaluation function, all subsequent models are explored. For instance, if removing feature 1 and feature 3 both yield the same RMSE, both subsets are explored.

Experiment	Training Set	Features Removed	Evaluation Function
1	U	Feature	RMSE wear value
2	U & S	Feature	RMSE wear value
3	U	Sensor	RMSE wear value
4	U &	Sensor	RMSE wear value
5	U	Feature	RMSE wear state
6	U & S	Feature	RMSE wear state
7	U	Sensor	RMSE wear state
8	U & S	Sensor	RMSE wear state
9	U	Feature	Percent correctly classified
10	U & S	Feature	Percent correctly classified
11	U	Sensor	Percent correctly classified
12	U & S	Sensor	Percent correctly classified

TABLE 3.21: Summary of each experiment for the supervised sequential search methods. The training set column indicates which tools are used for training the models. "U" indicates all unsupervised tools are used for training and the remaining supervised tools are used for evaluation. "S" indicates one supervised tool is used for training and one is used for evaluation. The feature removed column indicates if individual features or entire sensors are removed at each step. The evaluation function column displays the evaluation function used.

Tables B.1 to B.12 in Appendix B contain the results for the twelve experiments for SFS. The results for SBS are in Tables B.13 to B.24 in Appendix B. The first column labeled "Test" contains the tool used for testing. The second column labeled "Eval" contains the tools used for evaluation. The third column labeled "Train" contains the tools used for training the model and "U" designates all unsupervised tools (Tools 2, 3, and 5). The column labeled either "Added" or "Removed" contains a list of the features or sensors added or removed from the model in the order they were added or removed from the model. The final two columns contain the results of testing the selected features on the test tool. The evaluation function used during the evaluation step is used as the performance metric. The performance metric followed by a "1" is the model using only the training data when building the model for testing. The performance metric followed by a "2" is the model using the training and evaluation data for testing.

Before the sequential methods are compared with FSHMM, some general issues with the sequential methods are outlined. First, there are several decisions to be made before the sequential methods can be implemented. One must choose how to split the data set into training, evaluation, and testing sets. The experiments using sequential methods demonstrate that the selected feature subset and the performance of the final model are sensitive to the tools used in the training and

evaluation sets. For illustration, examine Tables B.15 and B.16. When testing on Tool 1, if Tools 4 and 6 are used for evaluation, SBS removes VY and FX. However, if Tool 6 is used for evaluation, SBS removes VZ and the RMSE drops from 25.95 to 17.27. Guyon and Elisseeff [44] state that variance in feature subset selection is an open problem that needs to be addressed in future research.

The results of the sequential methods are also sensitive to the evaluation function. For instance, in Table B.17 when Tool 1 is used for testing and the evaluation function is RMSE, the removed features are VZ_me, FX_me, VY_me, and VX_me. If we compare this with the testing of Tool 1 in Table B.21, where the data set is divided in the same fashion and the only difference is that accuracy is used as the evaluation function, FX_rms and FX_sle are removed.

If the same evaluation function is used but calculated in a slightly different way, the selected feature subsets can change. The results can be different if the RMSE between the predicted wear median and the true wear value is used as the evaluation function instead of the RMSE between the predicted wear state and the true wear state. In Table B.15 when Tool 1 is used as the test tool, the sensors VY and FX are removed. In Table B.19 with the same same tools used for training, evaluation, and testing, the sensors FZ and VZ are removed. Again, different feature subsets are selected when the only change is the way RMSE is calculated.

For each of these issues, only one instance in the set of experiments is highlighted; however, they can be found throughout the experiments. In conclusion, choices made about the supervised sequential methods before the process is implemented can have significant affect on the outcome.

When the sequential methods are compared with FSHMM, we see further drawbacks. First, as implemented above, the sequential methods require supervised data while the FSHMM does not. Later in this section, unsupervised sequential methods are tested and compared to the FSHMM.

The amount of computation for the sequential methods is much larger than for FSHMM. When using FSHMM, only one model is trained. Feature selection is conducted based on the results of parameter estimation. Then a reduced model is constructed from the estimated parameters. For sequential methods, numerous HMMs must be trained. In SBS, initially the full model is constructed, then L models, each removing a single different feature from the feature set, are trained. In the case of the PHM data, the first step in SBS requires 18 models to be trained.

Test	Eval	Train	Removed	RMSE1	RMSE2
1	4	U 6	FX	27.20	27.41
1	6	U 4	VZ	17.28	24.74
4	1	U 6	VX	23.18	21.61
4	6	U 1	FZ	25.08	26.50
6	1	U 4	VZ	14.72	32.27
6	4	U 1		34.75	34.76

TABLE 3.22: Experiment 4 SBS. The RMSE and the removed sensor are sensitive to the training and evaluation sets.

In each subsequent step, one less model is trained. The number of models in total for the sequential methods is dependent upon the number of features and the number of steps in the selection process.

The sequential methods must address the issue of ties when evaluating the removal or addition of features. In Table B.22, when Tool 1 is used for testing and Tool 6 is used for evaluation, there are seven different feature subsets that can be chosen due to ties in the evaluation function. There is one subset that is selected twice just in a different order. The results and the selected feature subsets are dependent upon which feature is chosen when a tie occurs. We have previously discussed how changes in the division of the data set and evaluation function can give different results and subsets. In this case, the division of the data and the evaluation function are the same. There is a significant spread in the accuracy depending on how the tie is broken. The accuracy ranges from 0.23 to 0.68. The FSHMM could also produce feature saliency estimates with the same value. However, the two features will either be included or excluded. The tie will not result in significantly different feature subsets and results as in the sequential methods.

When comparing the results of the sequential methods with those of the FSHMM, we must compare the RMSE2 in Table B.16. For ease of comparison, this table is reprinted in this chapter in Table 3.22. FSHMM performs better on Tools 1 and 4, but worse on Tool 6. Depending on how the data is divided into testing, training, and evaluation, SBS selects different sensors for removal. For three divisions of the data, a vibration sensor is removed. There is one instance when no sensor is selected for removal. These four scenarios result in a larger total sensor cost than FSHMM. When a force sensor is selected for removal, SBS cannot agree on which direction. In one scenario, FX is removed and in the other FZ is removed. FSHMM always chooses FY for removal.

Testing Tool	Resulting Model	RMSE
1	VX_rms VZ_rms VZ_me	27.67
4	VY_rms VZ_rms VZ_me	31.71
6	VX_rms VY_rms VZ_rms VZ_me	40.95
Average		33.44 ± 6.81

TABLE 3.23: Sequential search results for AIC and BIC when adding or removing a feature. Varying feature sets are selected depending on the training set. The average RMSE and ± 1 standard deviation are given.

Finally, the sequential methods cannot include the cost of the sensor into the feature selection process. There is no process built into the sequential methods for giving one feature preference over another feature. Each feature is treated the same, and feature selection is purely based on changes in the evaluation function.

The previous experiments involving sequential search required supervised data for the evaluation function. The FSHMM is an unsupervised method, so for a better comparison, sequential search methods using unsupervised evaluation functions are evaluated. The Akaike information criteria (AIC) and the Bayesian information criteria (BIC) are used as unsupervised evaluation functions [80]. AIC and BIC for HMMs are

$$AIC = 2((I-1) + I(I-1) + 2IL) - 2P(Y|\theta), \tag{3.37}$$

and

$$BIC = 2P(Y|\theta) - ((I-1) + I(I-1) + 2IL)\log(NT). \tag{3.38}$$

The quantity $((I-1) + I(I-1) + 2IL)$ is the number of free parameters and $P(Y|\theta)$ is the likelihood. AIC is minimized, while BIC is maximized at each step. Both evaluation functions and both search directions result in the same model. Tables 3.23 and 3.24 contain the resulting models when individual features and sensors are removed.

For a better comparison with the MAP FSHMM, unsupervised SBS is used to remove a single sensor. Both AIC and BIC result in the same sensor being removed. The results are displayed in Table 3.25. Testing on Tools 1 and 4 removes FY, the same sensor removed by MAP FSHMM. Testing on Tool 6 removes FX. All three testing tools remove a force sensor resulting in the same sensor expense as

Testing Tool	Resulting Model	RMSE
1	VZ	26.79
4	VZ	19.75
6	VZ	26.54
Average		24.36 ± 3.99

TABLE 3.24: Sequential search results for AIC and BIC when adding or removing a sensor. These models result in a single sensor in the selected feature subset. The average RMSE and ± 1 standard deviation are given.

Testing Tool	Removed Sensor	RMSE
1	FY	24.45
4	FY	25.12
6	FX	40.63
Average		30.07 ± 9.15

TABLE 3.25: Sequential search results for AIC and BIC when removing a single sensor. Varying force sensors are removed depending on the training set. The average RMSE and ± 1 standard deviation are given.

FSHMM. However, changing the training data result in a different selected feature subset. Further, the MAP FSHMM reduced models perform better than the SBS unsupervised models on two out of the three tools.

The unsupervised methods are also tested on the Kinect data set. The initialization, training, and testing sets are divided as in the experiments using the FSHMM. Six hidden states are assumed. For SFS, both AIC and BIC yield the same final model and add 16 features: HeadY, HeadZ, ShoulderCenterY, ShoulderCenterZ, ShoulderLeftY, ShoulderLeftZ, ElbowLeftY, ElbowLeftZ, WristLeftY, WristLeftZ, HandLeftY, HandLeftZ, ShoulderRightY, ShoulderRightZ, ElbowRightY, and ElbowRightZ. The accuracy for this model is surprisingly low at 0.0549. On the test set, the model predicts the first time step correctly then transitions to the 'Other' state and never transitions out of that state. The SFS method selects a larger feature subset, but includes several features associated with the Y and Z directions and excludes features associated with X. For comparison, the MAP FSHMM removes features in the Y direction and prefers features associated with X and Z.

For SBS, again AIC and BIC yield the same feature subset. WristRightX, HandRightX, and HandRightZ are removed during feature selection. The final model produces an accuracy of 0.5714 on the test set, while MAP FSHMM has an accuracy above 0.7.

Tool	K	Sensor Removed	RMSE (μm)
Tool 1	3	VZ_rms VZ_sle FX_me VX_me	17.72
Tool 4	3	VX_sle VY_sle FZ_me	23.19
Tool 6	3	VX_sle VY_sle FZ_me	34.45
Average			25.12 $\pm$ 8.53
Tool 1	4	VZ_rms VY_sle VZ_sle FX_me VX_me	20.23
Tool 4	4	FY_sle FZ_sle VX_sle VY_sle FZ_me	22.08
Tool 6	4	FY_sle FZ_sle VX_sle VY_sle FZ_me	37.90
Average			26.74 $\pm$ 9.71

TABLE 3.26: Results for feature similarity method on PHM data. Feature similarity cannot remove a single sensor. The average RMSE and $\pm$ 1 standard deviation are given.

From the experiments on the Kinect data, we can see that SFS and SBS using unsupervised methods selects features that increase the likelihood but not features that help the model accurately distinguish between states.

3.8.2 Filters

The feature similarity method [77] removes features that are similar to a specific feature by some measure. The idea is that removing these features will not significantly affect the predictive ability of the classifier. This method is similar to a K nearest neighbor classifier, and an initial value for K, the number of similar features to evaluate, must be selected for the algorithm. Mitra, Murthy, and Pal [77] claim that a good initial value for K is the number of features in the original data set minus the number of features desired in the reduced set. However, this relationship does not always hold.

For the PHM data, K = 3 and K = 4 were tested. The similarity method only produces a reduced set and does not rank features; therefore, the user has little control over the selection process. A single sensor was not able to be removed when using this method. The removed features and the RMSE for both values of K are displayed in Table 3.26.

When K = 3, four features are removed for Tool 1, and 3 features are removed for Tools 4 and 6. When K = 4, five features are removed for each tool. The former has a lower RMSE than the reduced MAP FSHMM model, while the latter has a higher RMSE. It is difficult to compare the similarity method to the FSHMM because FSHMM is required to remove the sensor with the lowest average feature

saliency. This type of testing is not possible with the feature similarity method and is one drawback to the method.

For the kinect data set, K = 10 and K = 15. The first setting removes 13 features and produces a test set accuracy of 0.6679. The other setting for K removes 21 features and has a test set accuracy of 0.6647. Neither is as accurate as the reduced MAP FSHMM model. Again, a direct comparison is difficult because the number of features removed is different from the FSHMM. This reiterates the drawback that it is difficult to control the size of the reduced feature subset when using feature similarity.

3.8.3 Principal Component Analysis

Principal component analysis (PCA) [80] is also used to reduce the number of features, even though it is not a true feature selection method. New features are constructed from the whole feature set instead of reducing the original set of features to a subset. In the case of the PHM problem, where we try to eliminate sensor to reduce the total cost of the sensor setup, PCA would not be acceptable because all the sensors would be required to construct components on new data. It is an unsupervised method of feature reduction, so it will be tested and compared with FSHMM.

The principal components (PCs) are calculated on all six tools. The percent of variance explained in the first five components is displayed in Figure 3.9 by the bars, and the cumulative sum of the variance is the blue line. The first four principal components account for 94% of the variance, the first five account for 96% and the first six for 98%. As before, LOOCV is used as the testing methodology. PCA is performed again on the training data with the testing tool removed. The testing tool is converted to PCs using the coefficients calculated on only the training data. HMMs are trained using Baum Welch on the first four, five, and six components, then tested using Viterbi. A five state model is assumed.

The RMSE for each testing tool using the first four, five, and six PCs are in Table 3.27. The MAP formulation of the FSHMM outperform PCA on Tools 1 and 4 but not on Tool 6. However, as previously discussed, no sensors are removed. All information for the sensors is needed to create the PCs.

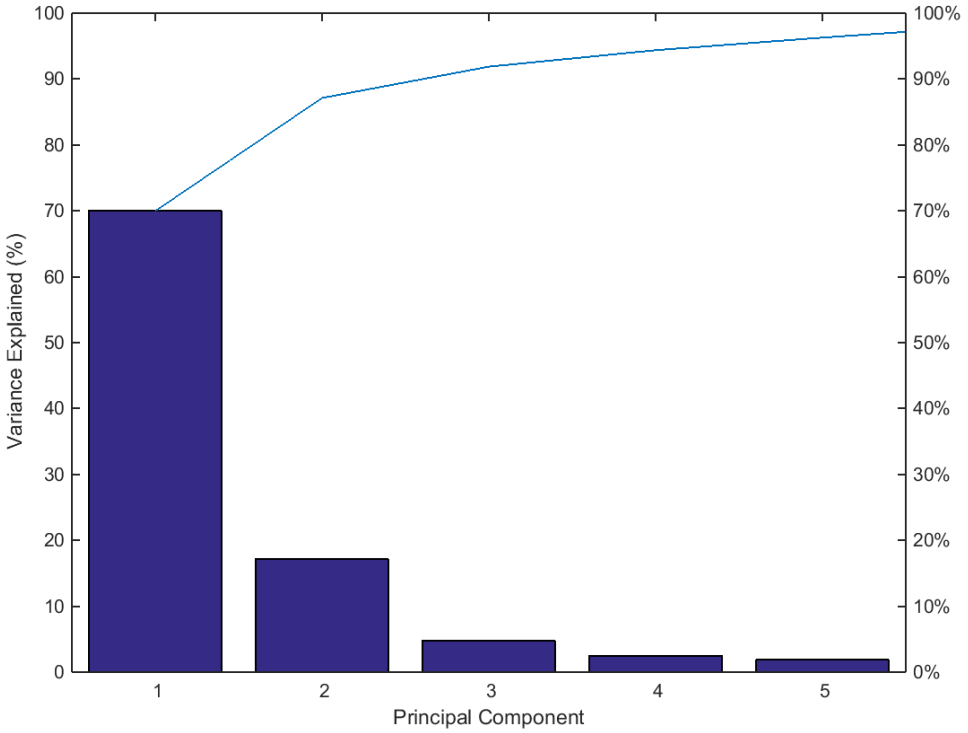

FIGURE 3.9: Scree plot for principal components. Bars indicate percent of variance explained by each PC and the solid line is the cumulative variance.

Tool	4 PCs	5 PCs	6 PCs
Tool 1	43.36	39.15	37.43
Tool 4	26.77	27.04	29.52
Tool 6	31.27	30.45	25.23
Average	33.80 ± 8.58	32.21 ± 6.24	30.73 ± 6.19

TABLE 3.27: RMSE for HMMs using first 4, 5 and 6 principal components. The average RMSE and ± 1 standard deviation are given.

A similar PCA is performed on the Kinect data set. The percent of variance explained by each PC is calculated using the entire data set, then PC coefficients are calculated on the training set. The PCs for the test set are calculated using the PC coefficients from the training set. The first two thirds of the noon hour are used for training and the last third is reserved for testing. The first two PCs account for 95% of the variance, the first four account for 98% and the first seven account for 99%. The test set accuracy for HMMs trained using the first two, four and seven PCs are 0.4998, 0.7165 and 0.6129. The ML and MAP reduced models for the FSHMM both outperform the best PC model tested. Further, as discussed previously, PC is not a FS technique so all the data from the Kinect is needed to calculate the PCs.

3.9 Chapter Conclusions

Conclusions from this chapter are listed below:

- The FSHMM can be used as a simultaneous feature selection and parameter estimation algorithm.
- The MAP formulation outperforms ML, MAP-beta, and VB in terms of feature subset selection size, feature subset selection consistency, and accuracy.
- $T/4$ is a heuristic for choosing the hyperparameters for the priors on ρ, given no other knowledge about the system or cost of features.
- VB does not perform as well as the EM based methods, and the estimated feature saliencies can be affected by the number of observations.
- ML overestimates the relevance of irrelevant features.
- MAP-beta underestimates the relevance of relevant features.
- Methods using priors on ρ can stabilize parameter estimates as the number of states increases. This is demonstrated in the PHM experiments.
- Redundant features do not necessarily need to be removed during feature selection for HMMs.
- MAP outperforms conventional feature selection techniques.

Chapter 4

Other Conditional Feature Distributions

The FSHMM outlined in the previous chapter assumes the relevant features follow a single Gaussian distribution. In this chapter, other conditional feature distributions are explored. Parameter estimates are derived, and the algorithms are tested on synthetic data. When applicable, the models are tested on the PHM, Kinect, or an event detection data set.

4.1 Gaussian Mixture Model

In this section, it is assumed that the relevant features follow a Gaussian mixture model (GMM) (see [8] and [95] for information on standard HMMs using GMM emissions). As in the previous chapter, consider an HMM with I states and L features. Let $\mathbf{y} = \{y_1, y_2, ..., y_T\}$ be the sequence of observed data where the observation for the l-th feature at time t, which is represented by the l-th component of y_t, is denoted by y_{lt}. Let $\mathbf{x} = \{x_1, x_2, ..., x_T\}$ be the unobserved state sequence. Let $\mathbf{z} = \{z_1, \ldots, z_L\}$ be a set of binary variables indicating the relevancy of each feature. If $z_l = 1$, then the l-th feature is relevant. Otherwise, if $z_l = 0$ the l-th feature is irrelevant.

Assume the relevant feature distribution is composed of a mixture of M Gaussian distributions. Let $\Phi = \{\phi_{1t}, ..., \phi_{Mt}\}$ be the hidden variable indicating the mixture component, where $\phi_{mt} = 1$ if observation t comes from the m^{th} mixture and

$\phi_{mt} = 0$ otherwise. Let ω_{im} be the probability that the observation comes from the m^{th} mixture, given the state. This model formulation has 3 hidden variables ($\mathbf{x}$, $\mathbf{z}$, and Φ) where the standard FSHMM only has 2 ($\mathbf{x}$ and $\mathbf{z}$). The set of model parameters Λ is now $\{\pi, A, \mu, \sigma, \rho, \epsilon, \tau, \omega\}$.

The marginal probability of ϕ_t is

$$P(\Phi|\Lambda) = \prod_{m=1}^{M} \omega_{im}^{\phi_{mt}}. \tag{4.1}$$

The marginal probability of $\mathbf{z}$ is the same as in Equation 3.2. Assuming the features are conditionally independent given the state, the conditional distribution of y_t given $\mathbf{z}$, $\mathbf{x}$ and Φ can be written as

$$P(y_t|\Phi, \mathbf{z}, x_t = i, \Lambda) = \prod_{l=1}^{L} [p(y_{lt}|\mu_{ilm}, \sigma_{ilm}^2)_l^z q(y_{lt}|\epsilon_l, \tau_l^2)^{1-z_l}]^{\phi_{mt}}. \tag{4.2}$$

The joint distribution of y_t, Φ, and $\mathbf{z}$ given $\mathbf{x}$ is

$$\begin{aligned} P(y_t, \Phi,\mathbf{z}|x_t = i, \Lambda) &= P(y_t|\Phi, \mathbf{z}, x_t = i, \Lambda) P(\Phi|\Lambda) P(\mathbf{z}|\Lambda) \\ &= \prod_{m=1}^{M} \left[\omega_{im} \prod_{l=1}^{L} [\rho_l p(y_{lt}|\mu_{ilm}, \sigma_{ilm}^2)]^{z_l} [(1-\rho_l) q(y_{lt}|\epsilon_l, \tau_l^2)]^{1-z_l} \right]^{\phi_{mt}}. \end{aligned} \tag{4.3}$$

The marginal distribution for y_t given $\mathbf{x}$ can be found by summing 4.3 over $\mathbf{z}$ and Φ.

$$\begin{aligned} f_{x_t}(y_t) &= P(y_t|x_t = i, \Lambda) \\ &= \sum_{m=1}^{M} \omega_{im} \prod_{l=1}^{L} \left(\rho_l p(y_{lt}|\mu_{ilm}, \sigma_{ilm}^2) + (1-\rho_l) q(y_{lt}|\epsilon_l, \tau_l^2) \right). \end{aligned} \tag{4.4}$$

The complete data likelihood for the FSHMM with GMM emissions is

$$P(\mathbf{x}, \mathbf{y}, \mathbf{z}, \Phi|\Lambda) = \pi_{x_1} P(y_1, \Phi, \mathbf{z}|x_1, \Lambda) \prod_{t=2}^{T} a_{x_{t-1}, x_t} P(y_t, \Phi, \mathbf{z}|x_t, \Lambda). \tag{4.5}$$

Derivations for the $\mathcal{Q}$ function and the ML and MAP parameter update equations are given in Appendix A Section A.5.

4.1.1 E-Step

The E-step is augmented from the standard FSHMM to include Φ. The $\mathcal{Q}$ function is

$$\begin{aligned}\mathcal{Q}(\Lambda, \Lambda') &= \mathbb{E}[\log P(\mathbf{x}, \mathbf{y}, \mathbf{z}, \Phi|\Lambda)|\mathbf{y}, \Lambda'] \\ &= \sum_{\mathbf{x},\mathbf{z},\Phi} \log(P(\mathbf{x}, \mathbf{y}, \mathbf{z}, \Phi|\Lambda))P(\mathbf{x}, \mathbf{z}, \Phi|\mathbf{y}, \Lambda').\end{aligned} \tag{4.6}$$

The conditional state probabilities $\gamma_t(i)$ and the conditional transition probabilities $\xi_t(i, j)$ are the same as in Equations 2.12 and 2.13.

The E-step probabilities are now

$$\begin{aligned}e_{ilmt} &= P(y_{lt}, z_l = 1|\phi_{mt} = 1, x_t = i, \Lambda') \\ &= \rho_l p(y_{lt}|\mu_{ilm}, \sigma^2_{ilm}),\end{aligned} \tag{4.7}$$

$$\begin{aligned}h_{ilmt} &= P(y_{lt}, z_l = 0|\phi_{mt} = 1, x_t = i, \Lambda') \\ &= (1 - \rho_l)q(y_{lt}|\epsilon_l, \tau_l^2),\end{aligned} \tag{4.8}$$

$$\begin{aligned}g_{ilmt} &= P(y_{lt}|\phi_{mt} = 1, x_t = i, \Lambda') \\ &= e_{ilmt} + h_{ilmt}\end{aligned} \tag{4.9}$$

$$\begin{aligned}u_{ilmt} &= P(z_l = 1, x_t = i, \phi_{mt} = 1|\mathbf{y}, \Lambda') \\ &= \gamma_t(i) \left(\frac{e_{ilmt}}{g_{ilmt}}\right) \left(\frac{\omega_{im} \prod_{l=1}^{L} g_{ilmt}}{\sum_{m=1}^{M} \omega_{im} \prod_{l=1}^{L} g_{ilmt}}\right),\end{aligned} \tag{4.10}$$

and

$$
\begin{aligned}
v_{ilmt} &= \mathbb{P}(z_l = 0, x_t = i, \phi_{mt} = 1 | \mathbf{y}, \Lambda') \\
&= \gamma_t(i) \left(\frac{h_{ilmt}}{g_{ilmt}} \right) \left(\frac{\omega_{im} \prod_{l=1}^{L} g_{ilmt}}{\sum_{m=1}^{M} \omega_{im} \prod_{l=1}^{L} g_{ilmt}} \right).
\end{aligned} \tag{4.11}
$$

4.1.2 M-Step ML

The M-step update parameters are

$$\hat{\pi}_i = \gamma_t(i), \tag{4.12}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \tag{4.13}$$

$$\hat{\mu}_{ilm} = \frac{\sum_{t=1}^{T} u_{ilmt} y_{lt}}{\sum_{t=1}^{T} u_{ilmt}}, \tag{4.14}$$

$$\hat{\sigma}^2_{ilm} = \frac{\sum_{t=1}^{T} u_{ilmt} (y_{lt} - \mu_{ilm})^2}{\sum_{t=1}^{T} u_{ilmt}}, \tag{4.15}$$

$$\hat{\epsilon}_l = \frac{\sum_{t=1}^{T} \big(\sum_{i=1}^{I} \sum_{m=1}^{M} v_{ilmt} \big) y_{lt}}{\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{m=1}^{M} v_{ilmt}}, \tag{4.16}$$

$$\hat{\tau}^2_l = \frac{\sum_{t=1}^{T} \big(\sum_{i=1}^{I} \sum_{m=1}^{M} v_{ilmt} \big) (y_{lt} - \epsilon_l)^2}{\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{m=1}^{M} v_{ilmt}}, \tag{4.17}$$

$$\hat{\omega}_{im} = \frac{\sum_{t=1}^{T} \sum_{l=1}^{L} u_{ilmt}}{\sum_{t=1}^{T} \sum_{l=1}^{L} \sum_{m=1}^{M} u_{ilmt}}, \tag{4.18}$$

and

$$
\begin{aligned}
\hat{\rho}_l &= \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{m=1}^{M} u_{ilmt}}{\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{m=1}^{M} u_{ilmt} + \sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{m=1}^{M} v_{ilmt}} \\
&= \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{m=1}^{M} u_{ilmt}}{T}.
\end{aligned} \tag{4.19}
$$

4.1.3 M-Step MAP

The priors used for MAP estimation are listed below. Dir is the Dirichlet distribution, $\mathcal{N}$ is the Gaussian distribution, IG is the inverse gamma distribution, and A_i is row i of the transition matrix.

$$\pi \sim \mathrm{Dir}(\pi|\bar{\boldsymbol{p}}), \tag{4.20}$$

$$A_i \sim \mathrm{Dir}(A_i|\bar{\boldsymbol{a}}_i), \tag{4.21}$$

$$\mu_{il} \sim \mathcal{N}(\mu_{il}|m_{il}, s_{il}^2), \tag{4.22}$$

$$\sigma_{il}^2 \sim \mathrm{IG}(\sigma_{il}^2|\zeta_{il}, \eta_{il}), \tag{4.23}$$

$$\epsilon_l \sim \mathcal{N}(\epsilon_l|b_l, c_l^2), \tag{4.24}$$

$$\tau_l^2 \sim \mathrm{IG}(\tau_l|\nu_l, \psi_l), \tag{4.25}$$

$$\boldsymbol{\omega}_i \sim \mathrm{Dir}(\boldsymbol{\omega}_i|\mathbf{w}_i), \tag{4.26}$$

$$\rho_l \sim \frac{1}{Z} e^{-k_l \rho_l}, \tag{4.27}$$

where Z is the normalizing constant, $\boldsymbol{\omega}_i$ represents the vector of mixture component for state i, and $\mathbf{w}_i$ represents the corresponding hyperparameter vector. The parameter update equations are

$$\hat{\pi}_i = \frac{\gamma_1(i) + \bar{p}_i - 1}{\sum_{i=1}^{I} \left(\gamma_1(i) + \bar{p}_i - 1\right)}, \tag{4.28}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j) + \bar{a}_{ij} - 1}{\sum_{j=1}^{I} \left(\sum_{t=1}^{T-1} \xi_t(i,j) + \bar{a}_{ij} - 1\right)}, \tag{4.29}$$

$$\hat{\mu}_{ilm} = \frac{s_{il}^2 \sum_{t=1}^{T} u_{iltm} y_{lt} + \sigma_{ilm}^2 m_{ilm}}{s_{ilm}^2 \sum_{t=1}^{T} u_{ilmt} + \sigma_{ilm}^2}, \tag{4.30}$$

$$\hat{\sigma}_{ilm}^2 = \frac{\sum_{t=1}^{T} u_{ilmt}(y_{lt} - \mu_{ilm})^2 + 2\eta_{ilm}}{\sum_{t=1}^{T} u_{ilmt} + 2(\zeta_{ilm} + 1)}, \tag{4.31}$$

$$\hat{\epsilon}_l = \frac{c_l^2 \sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilmt}\right) y_{lt} + \tau_l^2 b_l}{c_l^2 \sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilmt}\right) + \tau_l^2}, \tag{4.32}$$

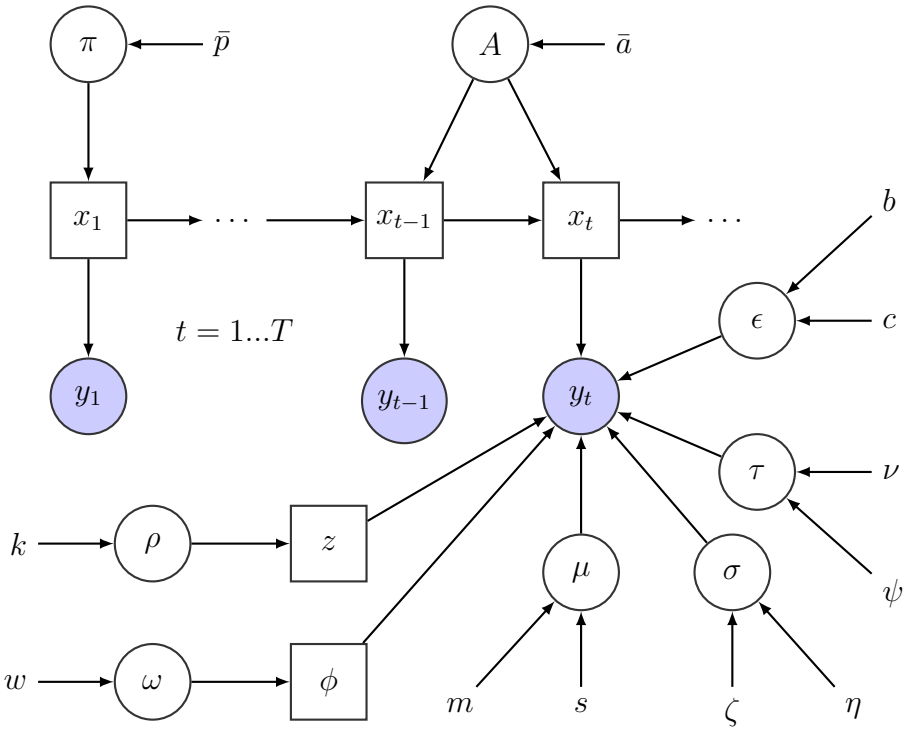

FIGURE 4.1: Graphical model for MAP GMM FSHMM using exponential prior. Squares represent hidden variables. Filled circles are observable variables. Open circles are model parameters.

$$\hat{\tau}_l^2 = \frac{\sum_{t=1}^{T}\left(\sum_{i=1}^{I}\sum_{m=1}^{M} v_{ilmt}\right)(y_{lt}-\epsilon_l)^2 + 2\psi_l}{\sum_{t=1}^{T}\left(\sum_{i=1}^{I}\sum_{m=1}^{M} v_{ilmt}\right) + 2(\nu_l+1)}, \tag{4.33}$$

$$\hat{\omega}_{im} = \frac{\sum_{t=1}^{T}\sum_{l=1}^{L} u_{ilmt} + w_{im} - 1}{\sum_{m=1}^{M}\left(\sum_{t=1}^{T}\sum_{l=1}^{L} u_{ilmt} + w_{im} - 1\right)}, \tag{4.34}$$

and

$$\hat{\rho}_l = \frac{T + k_l - \sqrt{(T+k_l)^2 - 4k_l(\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{m=1}^{M} u_{ilmt})}}{2k_l}. \tag{4.35}$$

The graphical model for the MAP GMM FSHMM using an exponential prior is displayed in Figure 4.1.

The beta prior $\mathcal{B}(\rho_l|k_l,\kappa_l)$ can also be used for feature saliencies when the emissions are assumed to be a GMM. The parameter update equation is

$$\rho_l = \frac{\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{m=1}^{M} u_{ilmt} + k_l - 1}{T + k_l + \kappa_l - 2}. \tag{4.36}$$

Algorithm	$\hat{\pi}_1$	$\hat{\pi}_2$	$\hat{a}_{11}$	$\hat{a}_{12}$	$\hat{a}_{21}$	$\hat{a}_{22}$
True Value	1	0	0.75	0.25	0.4	0.6
ML	1	0	0.7630	0.2370	0.4174	0.5826
MAP	0.8	0.2	0.7621	0.2379	0.4180	0.5820
MAP-beta	0.8	0.2	0.7625	0.2375	0.4177	0.5823
VB	1	0	0.7638	0.2362	0.4190	0.5810

TABLE 4.1: GMM parameter estimates for initial distribution and transition matrix. The priors on MAP and MAP-beta affect the estimates for the initial distribution. All transition probabilities are within 0.02 units of the true probability.

4.1.4 Synthetic Data Experiments

Three observation sequences, each with 500 time steps, are generated from a two state HMM. There are two relevant features and a GMM with two mixtures is used for the state conditional distribution for these features. Three irrelevant features of random noise, generated from $\mathcal{N}(0, I)$, are added to the data, resulting in a model with five features in total. The parameters for the model are listed in the tables with the estimated parameters.

The hyperparameters for the priors in the MAP and formulation are: $\bar{p}_i = \bar{a}_{ij} = 2, s_{il} = 0.5, \zeta_{il} = \eta_{il} = \nu_l = \psi_l = c_{il} = 1, k_l = 50$. b_l is the mean of the observations for the l^{th} feature. For the first relevant feature ($l = 1$), m is 1 minus the true value for $i = 1$ for each mixture and 1 plus the true value for $i = 2$ for each mixture. For the second feature ($l = 2$), m is 1 plus the true value for $i = 1$ for each mixture and 1 minus the true value for $i = 2$ for each mixture. MAP-beta uses the same hyperparmeters, except $k_l = 1$ and $\kappa_l = 50$. The hyperparmeters for the VB formulation are the same as described in [137].

The algorithms are initialized using equal initial state, transition, and mixture probabilities. μ is set to the prior in the MAP formulation m, $\sigma = 1, \epsilon = b, \tau$ is the standard deviation of the data, and $\rho_l = 0.5$. The algorithms are run for a maximum of 500 iterations. Convergence is tested by calculating the absolute percent change in the likelihood for ML, the posterior probability for MAP and MAP-beta, and the lower bound for VB. The convergence threshold is 10^{-6}.

The parameter estimates for the synthetic data are in Tables 4.1 to 4.7. The ML formulation converges in 37 iterations, the MAP formulation converges in 189 iterations, the MAP-beta formulation converges in 169 iterations and the VB formulation converges in 375 iterations. The conclusions from the synthetic data

Algorithm	$\hat{w}_{11}$	$\hat{w}_{21}$	$\hat{w}_{12}$	$\hat{w}_{22}$
True Value	0.75	0.25	0.4	0.6
ML	0.7438	0.2562	0.4304	0.5696
MAP	0.7446	0.2554	0.4306	0.5694
MAP-beta	0.7501	0.2499	0.4181	0.5819
VB	0.7411	0.2589	0.4235	0.5765

TABLE 4.2: GMM parameter estimates for mixture probabilities. All mixture probabilities are within 0.04 units of the true probability.

Algorithm	$\hat{\mu}_{111}$	$\hat{\mu}_{112}$	$\hat{\mu}_{121}$	$\hat{\mu}_{122}$	$\hat{\mu}_{211}$	$\hat{\mu}_{212}$	$\hat{\mu}_{221}$	$\hat{\mu}_{222}$
True Value	10	20	40	50	30	40	80	100
ML	9.84	19.82	39.92	49.94	29.99	40.13	80.37	100.13
MAP	9.78	19.77	40.05	50.03	30.24	40.18	80.00	100.08
MAP-beta	9.31	19.77	40.04	50.08	30.60	40.17	80.07	100.08
VB	9.85	19.82	39.91	49.93	30.13	40.13	80.59	100.13

TABLE 4.3: GMM parameter estimates for μ for relevant features. All parameters are within 1 unit of the true value.

Algorithm	$\hat{\sigma}_{111}$	$\hat{\sigma}_{112}$	$\hat{\sigma}_{121}$	$\hat{\sigma}_{122}$	$\hat{\sigma}_{211}$	$\hat{\sigma}_{212}$	$\hat{\sigma}_{221}$	$\hat{\sigma}_{222}$
True Value	5	2	5	2	5	2	5	2
ML	4.79	2.09	4.90	2.05	4.45	2.10	4.79	2.01
MAP	4.81	2.08	4.88	2.01	4.40	2.09	4.75	2.00
MAP-beta	4.46	1.97	4.77	1.88	3.69	2.08	4.62	2.00
VB	4.75	2.12	4.84	2.04	4.01	2.09	4.65	2.01

TABLE 4.4: GMM parameter estimates for σ for relevant features. All parameters are within 0.5 units of the true value.

Algorithm	$\hat{\epsilon}_3$	$\hat{\epsilon}_4$	$\hat{\epsilon}_5$
True Value	0	0	0
ML	0.02	-0.03	0.05
MAP	-0.02	-0.003	0.003
MAP-beta	-0.02	-0.003	0.003
VB	-0.02	-0.003	0.003

TABLE 4.5: GMM parameter estimates for ϵ for irrelevant features. All parameters are within 0.1 units of the true value.

Algorithm	$\hat{\tau}_3$	$\hat{\tau}_4$	$\hat{\tau}_5$
True Value	1	1	1
ML	0.95	1.00	0.98
MAP	0.95	1.02	1.02
MAP-beta	0.95	1.02	1.02
VB	1.09	0.97	0.96

TABLE 4.6: GMM parameter estimates for τ for irrelevant features. All parameters are within 0.1 units of the true value.

Algorithm	$\hat{\rho}_1$	$\hat{\rho}_2$	$\hat{\rho}_3$	$\hat{\rho}_4$	$\hat{\rho}_5$
True Value	1	1	0	0	0
ML	0.9978	0.9993	0.5020	0.5053	0.4962
MAP	1.0000	0.9973	0.0466	0.0046	0.0023
MAP-beta	0.9147	0.9472	0.0396	0.0051	0.0022
VB	0.9911	0.9896	10^{-9}	10^{-9}	10^{-9}

TABLE 4.7: GMM parameter estimates for feature saliencies of all features. ML overestimates the relevance of the irrelevant features, while MAP, MAP-beta, and VB estimate ρ below 0.05 for the irrelevant features. MAP-beta underestimates the relevance of the relevant features.

experiments are the same as in the single Gaussian case. The MAP and MAP-beta formulations produce skewed initial state distribution parameters due to the prior distribution. The ML formulation results in higher estimates for feature saliencies on the irrelevant features than the other three algorithms. MAP-beta slightly underestimates the relevance of relevant features; however, this trait is less pronounced than in the single Gaussian case. The VB formulation gives the smallest estimates for the irrelevant feature saliencies. All four algorithms produce accurate estimates for all the other parameters. MAP-beta and VB perform better on the GMM feature model than the single Gaussian model.

4.1.5 PHM Data

The GMM FSHMM is also tested on the PHM data set and the ML, MAP, MAP-beta and VB formulations are compared. We assume there are 5 states and two mixtures per state. The training and testing procedure is the same as in the previous chapter.

The ML, MAP and MAP-beta algorithms are initialized with the same values. The initial self transition a_{ii} is 0.9, the transition to the next state $a_{i,i+1}$ is 0.1, and $\pi_1 = 1$. The mixture components for each state are the inverse of the number of components $\omega_{im} = M^{-1}$. The state-dependent means for the first mixture μ_{il1} are equally spaced between -2 and 2. The initial value for the state-dependent mean of the second mixture is randomly chosen by adding a small amount of random noise to the value of the first mixture ($\mu_{il1} + \mathcal{N}(0, 0.25^2)$. The state-dependent standard deviation σ_{il} is 1 for all states, mixtures and features. The state-independent parameters are calculated from the training data. For MAP, the prior parameters are $\bar{a}_{ii} = \bar{a}_{i,i+1} = 2$, $\bar{a}_{ij} = 1$ for $j \neq i$ and $j \neq i+1$, $\bar{p}_1 = 2$, $\bar{p}_{i\neq 1} = 1$, $w_{im} = 2$,

Algorithm and Tool	RMSE Full Model (μm)	RMSE Reduced Model (μm)	Smallest $\bar{\rho}$	Removed Sensor
ML Tool 1	18.21	15.45	0.9520	Vibration Y
ML Tool 4	22.41	22.78	0.9663	Force X
ML Tool 6	32.28	28.90	0.9745	Vibration Y
Average	24.30 ± 7.22	22.38 ± 6.73		
MAP Tool 1	17.77	18.54	0.5383	Force Y
MAP Tool 4	22.50	13.13	0.5875	Force Y
MAP Tool 6	25.33	24.17	0.5757	Force Y
Average	21.87 ± 3.82	21.94 ± 5.52		
MAP-beta Tool 1	19.98	20.34	0.3290	Force Y
MAP-beta Tool 4	19.89	21.17	0.3270	Force Y
MAP-beta Tool 6	45.91	38.06	0.3250	Force Y
Average	28.59 ± 15.00	26.52 ± 10.00		
VB Tool 1	29.37	29.34	0.8939	Vibration Y
VB Tool 4	27.52	25.53	0.8593	Vibration Y
VB Tool 6	48.07	37.42	0.8524	Vibration Y
Average	34.99 ± 11.37	30.77 ± 6.07		

TABLE 4.8: Results for 5 state PHM experiments using 2 mixtures per state. MAP and MAP-beta, which incorporate the cost of features, consistently selects force in the Y direction for removal. VB selects vibration in the Y direction for removal, which is less expensive than the force sensor. ML removes varying sensors. The average RMSE and ± 1 standard deviation are given for each formulation.

$m = \mu_{init}$, $s = 0.5$, $\zeta = \eta = \nu = \psi = 0.5$, $b = 0$, and $c = 1$. Half of the assumed cost of each sensor is used for k_l ($k_l = 1200$ for the force features and $k_l = 600$ for the vibration features). For MAP-beta, the hyperparameters are the same as for MAP, except $k_l = 1$ and κ_l is half the assumed cost of the sensor. The hyperparameters for VB are the same as in [137]. The initial parameter values for the VB algorithm are set as close as possible to the initial values of the EM methods. The convergence threshold for this experiment is lowered to 10^{-6} for all four algorithms.

The ML and MAP formulations for the GMM FSHMM both have improved average RMSE over the single Gaussian case. The MAP-beta formulation using the GMM FSHMM does not perform as well as the single Gaussian case. The VB formulation shows improvement for the reduced model but not the full model. As expected, the estimated saliencies for the GMM FSHMM are closer to 1 than those in the single Gaussian case. While the difference in accuracy of the formulations might not be significant due to the small sample size, the formulations incorporating cost distinguish themselves during feature selection by consistently selecting

the more expensive sensor for removal.

The 10 and 20 state models are not tested in this analysis. As the number of states and mixtures grows, more data is needed to accurately estimate model parameters. Sparsity of the data with respect to state and mixture becomes an issue for the ML formulation. This causes estimates for some components of τ to go to 0. The MAP, MAP-beta and VB formulations do not run into this problem because the priors keep the estimates from going to 0.

4.2 Exponential Distribution

Consider an HMM with I states and L features. The notation for $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ is the same as in previous sections. Assume the relevant features have a state-dependent exponential distribution and the irrelevant features have a state-independent Gaussian distribution. The conditional distribution for y_t given $x_t = i$ and $\mathbf{z}$ is

$$P(y_t|\mathbf{z}, x_t = i, \Lambda) = \prod_{l=1}^{L} p(y_{lt}|\mu_{il})^{z_l} q(y_{lt}|\epsilon_l, \tau_l^2)^{1-z_l}, \tag{4.37}$$

where

$$p(y_{lt}|\mu_{il}) = \mu_{il} e^{-\mu_{il} y_{lt}}. \tag{4.38}$$

The marginal distribution for $\mathbf{z}$ is the same as Equation 3.2. The joint distribution of y_t and $\mathbf{z}$, given $x_t = i$, is the same as in Equation 3.3 with Equation 4.38 used for $p(\cdot|\cdot)$ and the joint distribution of $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$ is the same as in Equation 3.5 with Equation 4.37 used for $P(y_t|\mathbf{z}, x_t = i, \Lambda)$.

The state-independent distribution $q(\cdot|\cdot)$ is a Gaussian for the exponential features. This is due to the difficulty in distinguishing exponential distributions with different parameters. Figure 4.2 displays histograms of random samples drawn from exponential distributions with $\mu = 20$ and $\mu = 100$. The distribution with $\mu = 100$ is completely contained within the distribution with $\mu = 20$, making distinguishing between the two distributions difficult. The state-independent distribution is left as a Guassian to alleviate this issue.

The E-step probabilities are the same as in Equations 3.8 - 3.12 with the appropriate substitutions and $\gamma_t(i)$ and $\xi_t(i, j)$ calculated using the forward-backward

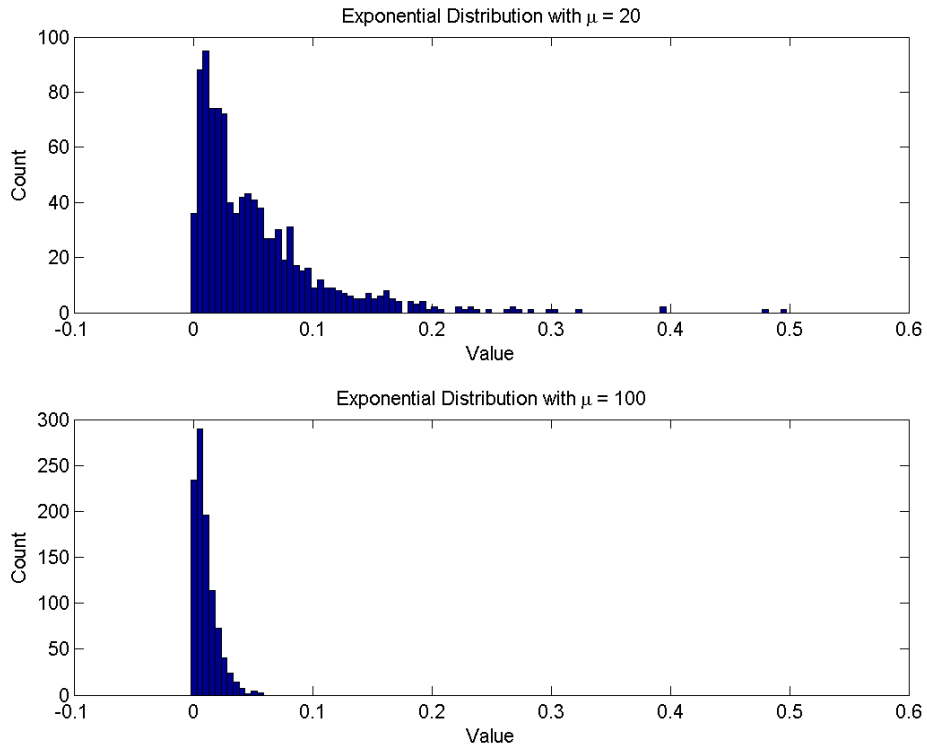

FIGURE 4.2: Top: Histogram of 1000 random samples from an exponential distribution with $\mu = 20$. Bottom: Histogram of 1000 random samples from an exponential distribution with $\mu = 100$. It is difficult for an algorithm to distinguish between these two exponential distributions.

algorithm (Equations 2.12 and 2.13). Derivations for the ML and MAP M-step parameter update equations are in Appendix A Section A.6.

4.2.1 M-Step ML

The M-step update parameters are

$$\hat{\pi}_i = \gamma_1(i), \tag{4.39}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \tag{4.40}$$

$$\hat{\mu}_{il} = \frac{\sum_{t=1}^{T} u_{ilt}}{\sum_{t=1}^{T} u_{ilt} y_{lt}}, \tag{4.41}$$

$$\hat{\epsilon}_l = \frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) y_{lt}}{\sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}}, \tag{4.42}$$

$$\hat{\tau}_l^2 = \frac{\sum_{t=11}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) (y_{lt} - \epsilon_l)^2}{\sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}}, \tag{4.43}$$

and

$$\begin{aligned} \hat{\rho}_l &= \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt}}{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt} + \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}} \\ &= \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{i,l,t}}{T}. \end{aligned} \tag{4.44}$$

4.2.2 M-Step MAP

The priors used for MAP estimation are listed below. Dir is the Dirichlet distribution, $\mathcal{N}$ is the Gaussian distribution, IG is the inverse gamma distribution, G is the Gamma distribution using shape and rate as hyperparameters, and A_i is row i of the transition matrix.

$$\pi \sim \text{Dir}(\pi | \bar{\boldsymbol{p}}), \tag{4.45}$$

$$A_i \sim \text{Dir}(A_i | \bar{\boldsymbol{a}}_i), \tag{4.46}$$

$$\mu_{il} \sim \mathcal{G}(\mu_{il} | m_{il}, s_{il}), \tag{4.47}$$

$$\epsilon_l \sim \mathcal{N}(\epsilon_l | b_l, c_l), \tag{4.48}$$

$$\tau_l^2 \sim \text{IG}(\tau_l | \nu_l, \psi_l), \tag{4.49}$$

$$\rho_l \sim \frac{1}{Z} e^{-k_l \rho_l}, \tag{4.50}$$

where Z is the normalizing constant. The parameter update equations are

$$\hat{\pi}_i = \frac{\gamma_1(i) + \bar{p}_i - 1}{\sum_{i=1}^{I} (\gamma_1(i) + \bar{p}_i - 1)}, \tag{4.51}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j) + \bar{a}_{ij} - 1}{\sum_{j=1}^{I} \left(\sum_{t=1}^{T-1} \xi_t(i,j) + \bar{a}_{ij} - 1 \right)}, \tag{4.52}$$

$$\hat{\mu}_{il} = \frac{\sum_{t=1}^{T} u_{ilt} + m_{il} - 1}{\sum_{t=1}^{T} u_{ilt} y_{lt} + s_{il}}, \tag{4.53}$$

$$\hat{\epsilon}_l = \frac{c_l^2 \sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) y_{lt} + \tau_l^2 b_l}{c_l^2 \sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) + \tau_l^2}, \tag{4.54}$$

$$\hat{\tau}_l^2 = \frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) (y_{lt} - \epsilon_l)^2 + 2\psi_l}{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) + 2(\nu_l + 1)}, \tag{4.55}$$

and

$$\hat{\rho}_l = \frac{T + k_l - \sqrt{(T + k_l)^2 - 4k_l(\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt})}}{2k_l}. \tag{4.56}$$

4.2.3 Synthetic Data

As in previous sections, the exponential formulation of the FSHMM is tested on synthetic data. However, there is no published VB formulation using an exponential distribution, so only the ML and MAP formulations are tested. MAP-beta is also excluded because the advantages and disadvantages of this prior have already been explored.

Three observation sequences of 500 time steps are generated from a two state HMM. There are two relevant features with state-dependent exponential distributions. Three irrelevant features of random noise generated from $\mathcal{N}(10, I)$ are added to the data, resulting in a model with five features in total. The mean of the irrelevant features is increased to 10 so all the observations are greater than or equal to 0. If the observations contained a negative emission, an exponential distribution would not be considered as an appropriate state-dependent distribution. The model parameters are

$$\mu_1 = \begin{bmatrix} 10 & 20 \end{bmatrix}, \quad \mu_2 = \begin{bmatrix} 50 & 100 \end{bmatrix}, \quad A = \begin{bmatrix} 0.75 & 0.25 \\ 0.4 & 0.6 \end{bmatrix}, \quad \pi = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

The hyperparameters for the priors in the MAP formulation are: $\bar{p}_i = \bar{a}_{ij} = 2, s_{il} = \nu_l = \psi_l = 1, c_l = 1$. b_l is the mean of the observations for the l^{th} feature. m_{il} for the relevant features is 1 minus the true value. m_{il} for the irrelevant features is 1 minus the true value for state 1 and 1 plus the true value for state 2. For the feature saliencies, the weight parameter k_l is set to 50. ML and MAP are initialized

Algorithm	$\hat{\pi}_1$	$\hat{\pi}_2$	$\hat{a}_{11}$	$\hat{a}_{12}$	$\hat{a}_{21}$	$\hat{a}_{22}$
ML	1	0	0.74	0.26	0.34	0.66
MAP	0.78	0.22	0.75	0.25	0.37	0.63

TABLE 4.9: Exponential feature parameter estimates for initial distribution and transition matrix. The prior used in MAP affects the estimates for the initial distribution. All transition probabilities are within 0.02 units of the true probability.

Algorithm	$\hat{\mu}_{11}$	$\hat{\mu}_{12}$	$\hat{\mu}_{21}$	$\hat{\mu}_{22}$
ML	9.35	19.16	48.21	86.10
MAP	9.72	19.23	49.65	88.05

TABLE 4.10: Exponential feature parameter estimates for μ for relevant features. The estimates for μ_{22} are more than 10% from the true value. The other estimates are within 2 units of the true value.

Algorithm	$\hat{\epsilon}_3$	$\hat{\epsilon}_4$	$\hat{\epsilon}_5$	$\hat{\tau}_3$	$\hat{\tau}_4$	$\hat{\tau}_5$
ML	10.03	9.96	9.98	0.99	1.01	1.00
MAP	10.03	9.96	9.98	0.99	1.01	1.00

TABLE 4.11: Exponential feature parameter estimates for ϵ and τ for irrelevant features. All parameter estimates are within 0.1 units of the true value.

Algorithm	$\hat{\rho}_1$	$\hat{\rho}_2$	$\hat{\rho}_3$	$\hat{\rho}_4$	$\hat{\rho}_5$
ML	0.9902	0.9644	5.937×10^{-9}	6.0405×10^{-9}	6.1995×10^{-9}
MAP	1.0000	1.0000	5.9445×10^{-9}	6.0489×10^{-9}	6.2079×10^{-9}

TABLE 4.12: Exponential feature parameter estimates for ρ of all features. For the exponential feature FSHMM, the ML formulation does not overestimate the relevance of irrelevant features.

with the same values: equal initial state probabilities and transition probabilities, $\mu = m, \epsilon = b, \rho = 0.5$, and τ is the standard deviation of the data. The maximum number of iterations is 500 and the convergence threshold is 10^{-6}.

The estimated parameters are in Tables 4.9 to 4.12. In the training data, all three of the state sequences start in state 1. The ML algorithm gives initial state distribution estimates that match the true value exactly. The prior on the initial state distribution parameters for MAP results in skewed estimates. Both algorithm give accurate parameter estimates for the transition probabilities. The MAP formulation gives parameter estimates for μ that are marginally closer to the true value than the ML formulation. Both formulations produce estimates for μ_{22} that are more than 10% from the true value. This is due to the difficulty of distinguishing the difference between lower valued observations drawn from exponential distributions. For instance, when the distribution is exponential, $P(0.01|\mu = 20) = 16.38$

while $P(0.01|\mu = 100) = 36.79$. This point is further illustrated in Figure 4.2. Observations on the lower end could belong to either distribution.

The estimates for ρ are accurate for both formulations. This is due to the algorithm easily distinguishing between features that come from an exponential distribution and features generated by a Gaussian distribution. The advantage of the MAP formulation is the ability to include the test cost of features.

4.3 Gamma Distribution

Consider an HMM with I states and L features. The notation for $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ is the same as in previous sections. Assume the relevant features have a state-dependent Gamma distribution and the irrelevant features have a state-independent Gamma distribution. The conditional distribution of y_t given $\mathbf{z}$ and $\mathbf{x}$ is

$$P(y_t|\mathbf{z}, x_t = i, \Lambda) = \prod_{l=1}^{L} p(y_{lt}|\mu_{il}, \sigma_{il})^{z_l} q(y_{lt}|\epsilon_l, \tau_l^2)^{1-z_l}, \tag{4.57}$$

where

$$p(y_{lt}|\mu_{il}, \sigma_{il}) = \frac{\sigma_{il}^{\mu_{il}}}{\Gamma(\mu_{il})} y_{lt}^{\mu_{il}-1} e^{-\sigma_{il} y_{lt}}, \tag{4.58}$$

and $q(y_{lt}|\epsilon_l, \tau_l^2)$ follows a similar Gamma distribution. The marginal distribution for $\mathbf{z}$ is the same as Equation 3.2. The joint distribution of y_t and $\mathbf{z}$ given $x_t = i$ is the same as in Equation 3.3 with Equation 4.58 used for $p(\cdot|\cdot)$ and $q(\cdot|\cdot)$, and the joint distribution of $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ is the same as in Equation 3.5 with Equation 4.57 used for $P(y_t|\mathbf{z}, x_t = i, \Lambda)$.

The E-step probabilities are the same as in Equations 3.8 - 3.12 with the appropriate substitutions, and $\gamma_t(i)$ and $\xi_t(i, j)$ are calculated using the forward-backward algorithm (Equations 2.12 and 2.13). Derivations for the ML and MAP M-step parameter update equations are in Appendix A Section A.7.

4.3.1 ML M-Step

There are no closed form solutions for μ_{il} and ϵ_l, so an iterative optimization routine must be used. Almhana et al. [2] use a gradient based method to update

the shape parameter for mixtures of Gamma distributions; however, we outline a Newton's method for the Gamma feature parameters, which is presented in the following section.

The M-step parameter update equations for the parameters with closed form solutions are

$$\hat{\pi}_i = \gamma_1(i), \tag{4.59}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \tag{4.60}$$

$$\hat{\sigma}_{il} = \frac{\mu_{il} \sum_{t=1}^{T} u_{ilt}}{\sum_{t=1}^{T} u_{ilt} y_{lt}}, \tag{4.61}$$

$$\hat{\tau}_l = \frac{\epsilon_l \sum_{t=1}^{T} \big(\sum_{i=1}^{I} v_{ilt} \big)}{\sum_{t=1}^{T} \big(\sum_{i=1}^{I} v_{ilt} \big) y_{lt}}, \tag{4.62}$$

and

$$\begin{aligned} \hat{\rho}_l &= \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt}}{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt} + \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}} \\ &= \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{i,l,t}}{T}. \end{aligned} \tag{4.63}$$

4.3.2 MAP M-step

The priors used for MAP estimation are listed below. Dir is the Dirichlet distribution, $\mathcal{N}$ is the Gaussian distribution, G is the Gamma distribution using shape and rate as hyperparameters, and A_i is row i of the transition matrix.

$$\pi \sim \mathrm{Dir}(\pi|\bar{\boldsymbol{p}}), \tag{4.64}$$

$$A_i \sim \mathrm{Dir}(A_i|\bar{\boldsymbol{a}}_i), \tag{4.65}$$

$$\mu_{il} \sim \mathrm{G}(\mu_{il}|m_{il}, s_{il}), \tag{4.66}$$

$$\sigma_l \sim \mathrm{G}(\sigma_{il}|\zeta_{il}, \eta_{il}), \tag{4.67}$$

$$\epsilon_l \sim \mathrm{G}(\epsilon_l|b_l, c_l), \tag{4.68}$$

$$\tau_l \sim \mathrm{G}(\tau_l|\nu_l, \psi_l), \tag{4.69}$$

$$\rho_l \sim \frac{1}{Z} e^{-k_l \rho_l}, \tag{4.70}$$

where the Zs are the normalizing constants. Again, there are no closed form solutions for μ_{il} and ϵ_l. The parameter update equations for the parameters with closed form solutions are

$$\hat{\pi}_i = \frac{\gamma_1(i) + \bar{p} - 1}{\sum_{i=1}^{I} \left(\gamma_1(i) + \bar{p}_i - 1\right)}, \tag{4.71}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j) + \bar{a}_{ij} - 1}{\sum_{j=1}^{I} \left(\sum_{t=1}^{T-1} \xi_t(i,j) + \bar{a}_{ij} - 1\right)}, \tag{4.72}$$

$$\hat{\sigma}_{il} = \frac{\mu_{il} \sum_{t=1}^{T} u_{ilt} + \zeta_{il} - 1}{\sum_{t=1}^{T} u_{ilt} y_{lt} + \eta_{il}}, \tag{4.73}$$

$$\hat{\tau}_l = \frac{\epsilon_l \sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) + \nu_l - 1}{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) y_{lt} + \psi_l}, \tag{4.74}$$

and

$$\hat{\rho}_l = \frac{T + k_l - \sqrt{(T + k_l)^2 - 4k_l(\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt})}}{2k_l}. \tag{4.75}$$

4.3.3 Newton's Method for Gamma Features

Even though there is a closed form solution for the rate parameter, we optimize both the shape and rate simultaneously using Newton's method. (The state-dependent and state-independent parameters can be treated as two separate optimization problems.) Further, both of these parameters must be strictly greater

than zero, therefore, a barrier function is used. The optimization problem for the relevant distribution using ML is defined as follows

$$\begin{aligned} \underset{\mu_{il},\sigma_{il}}{\text{minimize}} \quad & -\sum_{t=1}^{T} u_{ilt} \log p(y_{lt}|\mu_{il},\sigma_{il}) \\ \text{subject to} \quad & \mu_{il} > 0, i = 1...I, l = 1...L, \\ & \sigma_{il} > 0, i = 1...I, l = 1...L. \end{aligned} \tag{4.76}$$

The optimization problem for the irrelevant distribution using ML is defined as follows

$$\begin{aligned} \underset{\epsilon_l,\tau_l}{\text{minimize}} \quad & -\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt} \log q(y_{lt}|\epsilon_l,\tau_l) \\ \text{subject to} \quad & \epsilon_l > 0, l = 1...L, \\ & \tau_l > 0, l = 1...L. \end{aligned} \tag{4.77}$$

For MAP estimation, the logarithm of the priors is added to the objective function

$$\begin{aligned} \underset{\mu_{il},\sigma_{il}}{\text{minimize}} \quad & -\sum_{t=1}^{T} u_{ilt} \log p(y_{lt}|\mu_{il},\sigma_{il}) - \log(G(\mu_{il})) - \log(G(\sigma_{il})) \\ \text{subject to} \quad & \mu_{il} > 0, i = 1...I, l = 1...L, \\ & \sigma_{il} > 0, i = 1...I, l = 1...L. \end{aligned} \tag{4.78}$$

and

$$\begin{aligned} \underset{\epsilon_l,\tau_l}{\text{minimize}} \quad & -\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt} \log q(y_{lt}|\epsilon_l,\tau_l) - \log(G(\epsilon_l)) - \log(G(\tau_l)) \\ \text{subject to} \quad & \epsilon_l > 0, l = 1...L, \\ & \tau_l > 0, l = 1...L. \end{aligned} \tag{4.79}$$

To ensure the constraints are followed, a log barrier function is added to the objective. The ML optimization problems are now

$$\underset{\mu_{il},\sigma_{il}}{\text{minimize}} \quad -\sum_{t=1}^{T} u_{ilt} \log p(y_{lt}|\mu_{il},\sigma_{il}) - \frac{1}{c_k}\log(\mu_{il}) - \frac{1}{c_k}\log(\sigma_{il}), \tag{4.80}$$

and

$$\underset{\epsilon_l,\tau_l}{\text{minimize}} \quad -\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}\log q(y_{lt}|\epsilon_l,\tau_l) - \frac{1}{c_k}\log(\epsilon_l) - \frac{1}{c_k}\log(\tau_l). \tag{4.81}$$

The MAP optimization problems with the barrier functions are

$$\begin{aligned}\underset{\mu_{il},\sigma_{il}}{\text{minimize}} \quad & -\sum_{t=1}^{T} u_{ilt}\log p(y_{lt}|\mu_{il},\sigma_{il}) - \log(G(\mu_{il})) - \log(G(\sigma_{il})) \\ & - \frac{1}{c_k}\log(\mu_{il}) - \frac{1}{c_k}\log(\sigma_{il}),\end{aligned} \tag{4.82}$$

and

$$\begin{aligned}\underset{\epsilon_l,\tau_l}{\text{minimize}} \quad & -\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}\log q(y_{lt}|\epsilon_l,\tau_l) - \log(G(\epsilon_l)) - \log(G(\tau_l)) \\ & - \frac{1}{c_k}\log(\epsilon_l) - \frac{1}{c_k}\log(\tau_l),\end{aligned} \tag{4.83}$$

where c_k is the number of iterations at iteration k of the optimization. The barrier function for the rate parameter is not always necessary, and can be omitted in some implementations.

Newton's method [80] is an iterative optimization technique using the gradient of the objective function and the Hessian matrix. Let $\mathbf{x}$ be the set of parameters one wishes to minimize. Newtons method is

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha\mathbf{H}^{-1}\mathbf{G}, \tag{4.84}$$

where k is the iteration, α is the step size, $\mathbf{H}$ is the Hessian, and $\mathbf{G}$ is the gradient. The optimization stops after a specified number of iterations or $\mathbf{H}^{-1}\mathbf{G}$ becomes sufficiently small.

Let the objective function be represented by F. The components of the gradient and the Hessian matrix needed for Newton's method for the ML formulation are

$$\frac{\partial F}{\partial \mu_{il}} = -\log(\sigma_{il})\sum_{t=1}^{T} u_{ilt} + \Psi(\mu_{il})\sum_{t=1}^{T} u_{ilt} + \sum_{t=1}^{T} u_{ilt}\log(y_{lt}) - \frac{1}{c_k\mu_{il}}, \tag{4.85}$$

$$\frac{\partial F}{\partial \sigma_{il}} = -\frac{\mu_{il}}{\sigma_{il}} \sum_{t=1}^{T} u_{ilt} + \sum_{t=1}^{T} u_{ilt} y_{lt} - \frac{1}{c_k \sigma_{il}}, \tag{4.86}$$

$$\frac{\partial^2 F}{\partial \mu_{il}^2} = \frac{\partial \Psi(\mu_{il})}{\partial \mu_{il}} \sum_{t=1}^{T} u_{ilt} + \frac{1}{c_k \mu_{il}^2}, \tag{4.87}$$

$$\frac{\partial^2 F}{\partial \mu_{il} \partial \sigma_{il}} = -\frac{1}{\sigma_{il}} \sum_{t=1}^{T} u_{ilt}, \tag{4.88}$$

$$\frac{\partial^2 F}{\partial \sigma_{il}^2} = \frac{\mu_{il}}{\sigma_{il}^2} \sum_{t=1}^{T} u_{ilt} + \frac{1}{c_k \sigma_{il}^2}, \tag{4.89}$$

$$\frac{\partial F}{\partial \epsilon_l} = -\log(\tau_l) \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} + \Psi(\epsilon_l) \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} + \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} \log(y_{lt}) - \frac{1}{c_k \epsilon_l}, \tag{4.90}$$

$$\frac{\partial F}{\partial \tau_l} = -\frac{\epsilon_l}{\tau_l} \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} + \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} y_{lt} - \frac{1}{c_k \tau_l}, \tag{4.91}$$

$$\frac{\partial^2 F}{\partial \epsilon_l^2} = \frac{\partial \Psi(\epsilon_l)}{\partial \epsilon_l} \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} + \frac{1}{c_k \epsilon_l^2}, \tag{4.92}$$

$$\frac{\partial^2 F}{\partial \epsilon_l \partial \tau_l} = -\frac{1}{\tau_l} \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}, \tag{4.93}$$

and

$$\frac{\partial^2 F}{\partial \tau_l^2} = \frac{\epsilon_l}{\tau_l^2} \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} + \frac{1}{c_k \tau_l^2}, \tag{4.94}$$

where $\Psi(\mu_{il}) = \frac{\partial \log(\Gamma(\mu_{il}))}{\partial \mu_{il}}$ and $\Psi(\epsilon_l) = \frac{\partial \log(\Gamma(\epsilon_l))}{\partial \mu_{il}}$. $\Psi(\cdot)$ is referred to as the digamma function.

For MAP, the components of the gradient and Hessian are

$$\begin{aligned} \frac{\partial F}{\partial \mu_{il}} = &-\log(\sigma_{il}) \sum_{t=1}^{T} u_{ilt} + \Psi(\mu_{il}) \sum_{t=1}^{T} u_{ilt} + \sum_{t=1}^{T} u_{ilt} \log(y_{lt}) \\ &- \frac{m_{il} - 1}{\mu_{il}} + s_{il} - \frac{1}{c_k \mu_{il}}, \end{aligned} \tag{4.95}$$

$$\frac{\partial F}{\partial \sigma_{il}} = -\frac{\mu_{il}}{\sigma_{il}} \sum_{t=1}^{T} u_{ilt} + \sum_{t=1}^{T} u_{ilt} y_{lt} - \frac{\zeta_{il} - 1}{\sigma_{il}} + \eta_{il} - \frac{1}{c_k \sigma_{il}}, \tag{4.96}$$

$$\frac{\partial^2 F}{\partial \mu_{il}^2} = \frac{\partial \Psi(\mu_{il})}{\partial \mu_{il}} \sum_{t=1}^{T} u_{ilt} + \frac{m_{il} - 1}{\mu_{il}^2} + \frac{1}{c_k \mu_{il}^2}, \tag{4.97}$$

$$\frac{\partial^2 F}{\partial \mu_{il} \partial \sigma_{il}} = -\frac{1}{\sigma_{il}} \sum_{t=1}^{T} u_{ilt}, \tag{4.98}$$

$$\frac{\partial^2 F}{\partial \sigma_{il}^2} = \frac{\mu_{il}}{\sigma_{il}^2} \sum_{t=1}^{T} u_{ilt} + \frac{\zeta_{il} - 1}{\sigma_{il}^2} + \frac{1}{c_k \sigma_{il}^2}, \tag{4.99}$$

$$\begin{aligned} \frac{\partial F}{\partial \epsilon_l} = &- \log(\tau_l) \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} + \Psi(\epsilon_l) \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} + \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} \log(y_{lt}) \\ &- \frac{b_l - 1}{\epsilon_l} + c_l - \frac{1}{c_k \epsilon_l}, \end{aligned} \tag{4.100}$$

$$\frac{\partial F}{\partial \tau_l} = -\frac{\epsilon_l}{\tau_l} \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} + \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} y_{lt} - \frac{\nu_l - 1}{\tau_l} + \psi_l - \frac{1}{c_k \tau_l}, \tag{4.101}$$

$$\frac{\partial^2 F}{\partial \epsilon_l^2} = \frac{\partial \Psi(\epsilon_l)}{\partial \epsilon_l} \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} + \frac{b_l - 1}{\epsilon_l^2} + \frac{1}{c_k \epsilon_l^2}, \tag{4.102}$$

$$\frac{\partial^2 F}{\partial \epsilon_l \partial \tau_l} = -\frac{1}{\tau_l} \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}, \tag{4.103}$$

and

$$\frac{\partial^2 F}{\partial \tau_l^2} = \frac{\epsilon_l}{\tau_l^2} \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} + \frac{\nu_l - 1}{\tau_l^2} + \frac{1}{c_k \tau_l^2}. \tag{4.104}$$

4.3.4 Synthetic Data

For the synthetic data tests, an observation sequence of 2000 time steps is generated from a two state HMM. There are two relevant features with state-dependent Gamma distributions. Three irrelevant features are generated from a single Gamma distribution, and are added to the data, resulting in a model with five features in total. The model parameters are

Algorithm	$\hat{\pi}_1$	$\hat{\pi}_2$	$\hat{a}_{11}$	$\hat{a}_{12}$	$\hat{a}_{21}$	$\hat{a}_{22}$
ML	1	0	0.74	0.26	0.40	0.60
MAP	0.67	0.33	0.74	0.26	0.40	0.60

TABLE 4.13: Gamma parameter estimates for initial distribution and transition matrix. The prior in MAP affects the initial distribution. The estimates for the transition probabilities are within 0.01 units of the true probability.

Algorithm	$\hat{\mu}_{11}$	$\hat{\mu}_{12}$	$\hat{\mu}_{21}$	$\hat{\mu}_{22}$	$\hat{\sigma}_{11}$	$\hat{\sigma}_{12}$	$\hat{\sigma}_{21}$	$\hat{\sigma}_{22}$
ML	10.40	21.04	49.18	112.74	1.04	1.05	0.98	1.13
MAP	10.21	20.90	48.49	107.19	1.01	1.04	0.96	1.07

TABLE 4.14: Gamma parameter estimates for μ and σ for relevant features. The estimates for μ_{22} are more than 5% from the true value. The rest of the parameter estimates are within 2 units of the true value.

Algorithm	$\hat{\epsilon}_3$	$\hat{\epsilon}_4$	$\hat{\epsilon}_5$	$\hat{\tau}_3$	$\hat{\tau}_4$	$\hat{\tau}_5$
ML	32.54	29.78	32.29	1.08	0.99	1.07
MAP	32.01	29.23	31.86	1.07	0.97	1.06

TABLE 4.15: Gamma parameter estimates for ϵ and τ for irrelevant features. The estimates for ϵ are within 3 units of the true value. The estimates for τ 0.1 units of the true value.

$$\mu_1 = \begin{bmatrix} 10 & 20 \end{bmatrix}, \quad \mu_2 = \begin{bmatrix} 50 & 100 \end{bmatrix}, \sigma_1 = \begin{bmatrix} 1 & 2 \end{bmatrix}, \quad \sigma_2 = \begin{bmatrix} 1 & 1 \end{bmatrix},$$

$$A = \begin{bmatrix} 0.75 & 0.25 \\ 0.4 & 0.6 \end{bmatrix}, \quad \pi = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

The shape parameter for the single Gamma is 30 and the rate is 1.

The hyperparameters for the priors in the MAP formulation are: $\bar{p}_i = \bar{a}_{ij} = 2, s_{il} = \zeta_{il} = \eta_{il} = \nu_l = \psi_l = c_l = 1$. b_l is the mean of the observations for the l^{th} feature; and $m_{1l} = [8\ 18]$ and $m_{2l} = [45\ 90]$ for the relevant features. m_{il} for the irrelevant features is 1. For the feature saliencies, the weight parameter k_l is set to 50. ML and MAP are initialized with the same values: equal initial state probabilities and transition probabilities, $\mu_{init} = m, \sigma_{init} = 0.9, \epsilon_{init} = b, \tau_{init} = 0.9$, and $\rho_{init} = 0.5$,. The maximum number of iterations is 500 and the convergence threshold is 10^{-6}.

The estimated parameters are in Tables 4.13 to 4.16. As expected, the parameter estimates for the initial state distribution for the MAP formulation are skewed due to the prior. The initial state distribution for ML matches the true parameters

Algorithm	$\hat{\rho}_1$	$\hat{\rho}_2$	$\hat{\rho}_3$	$\hat{\rho}_4$	$\hat{\rho}_5$
ML	0.9990	0.9987	0.0011	0.0019	9.9996×10^{-4}
MAP	1	1	0	0	0

TABLE 4.16: Gamma parameter estimates for ρ of all features. Both formulations identify the relevant and irrelevant features.

exactly. Both algorithms give the same accurate parameter estimates for the transition probabilities. The parameter estimates for the shape and rate parameters for both the relevant and irrelevant features are within 20% of the true values. For the feature saliency parameter estimates, both algorithms accurately identify the relevant and irrelevant features with the MAP algorithm picking the true values exactly. The MAP algorithm converges in 18 iterations, while ML converges in 35. In conclusion, ML and MAP are both viable options for the FSHMM with Gamma features. On the synthetic data, neither algorithm significantly distinguishes itself from the other in terms of parameter estimation accuracy but MAP does converge faster. MAP also has the ability to include the cost of features. These tests on synthetic data demonstrate that Newton's method can be used for estimating parameters of an HMM with Gamma emissions.

4.4 Poisson Features

Consider an HMM with I states and L features. The notation for $\mathbf{x}$ and $\mathbf{z}$ is the same as in previous sections; however, Y_{lt} is used to denote a discrete observation. Assume the relevant features have a state-dependent Poisson distribution and the irrelevant features have a state-independent Poisson distribution. The conditional distribution for Y_t given $x_t = i$ and $\mathbf{z}$ is

$$P(y_t|\mathbf{z}, x_t = i, \Lambda) = \prod_{l=1}^{L} p(y_{lt}|\mu_{il}, \sigma_{il}^2)^{z_l} q(y_{lt}|\epsilon_l, \tau_l^2)^{1-z_l}, \tag{4.105}$$

where

$$p(Y_{lt}|\mu_{il}) = \frac{\mu_{il}^{Y_{lt}} e^{-\mu_{il}}}{Y_{lt}!}. \tag{4.106}$$

The marginal distribution for $\mathbf{z}$ is the same as Equation 3.2. The joint distribution of y_t and $\mathbf{z}$ given $x_t = i$ is the same as in Equation 3.3 with Equation 4.106 used

for $p(\cdot|\cdot)$ and $q(\cdot|\cdot)$. The joint distribution of $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ is the same as in Equation 3.5 with Equation 4.105 used for $P(y_t|\mathbf{z}, x_t = i, \Lambda)$.

The E-step probabilities are the same as in Equations 3.8 - 3.12 with the appropriate substitutions and $\gamma_t(i)$ and $\xi_t(i,j)$ calculated using the forward-backward algorithm (Equations 2.12 and 2.13). Derivations for the ML and MAP M-step parameter update equations are in Appendix A Section A.8.

4.4.1 M-Step ML

The M-step update parameters are

$$\hat{\pi}_i = \gamma_1(i), \tag{4.107}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \tag{4.108}$$

$$\hat{\mu}_{il} = \frac{\sum_{t=1}^{T} u_{ilt} Y_{lt}}{\sum_{t=1}^{T} u_{ilt}}, \tag{4.109}$$

$$\hat{\epsilon}_l = \frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) Y_{lt}}{\sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}}, \tag{4.110}$$

and

$$\begin{aligned} \hat{\rho}_l &= \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt}}{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt} + \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}} \\ &= \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{i,l,t}}{T}. \end{aligned} \tag{4.111}$$

4.4.2 M-Step MAP

The priors used for MAP estimation are listed below. Dir is the Dirichlet distribution, $\mathcal{N}$ is the Gaussian distribution, G is the Gamma distribution using shape and rate as hyperparameters, and A_i is row i of the transition matrix.

$$\pi \sim \mathrm{Dir}(\pi|\bar{\boldsymbol{p}}), \tag{4.112}$$

$$A_i \sim \mathrm{Dir}(A_i|\bar{\boldsymbol{a}}_i), \tag{4.113}$$

$$\mu_{il} \sim \mathrm{G}(\mu_{il}|m_{il}, s_{il}), \tag{4.114}$$

$$\epsilon_l \sim \mathrm{G}(\epsilon_l|b_l, c_l), \tag{4.115}$$

$$\rho_l \sim \frac{1}{Z} e^{-k_l \rho_l}, \tag{4.116}$$

where Z is the normalizing constant. The parameter update equations are

$$\hat{\pi}_i = \frac{\gamma_1(i) + \bar{p}_i - 1}{\sum_{i=1}^{I} (\gamma_1(i) + \bar{p}_i - 1)}, \tag{4.117}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j) + \bar{a}_{ij} - 1}{\sum_{j=1}^{I} \left(\sum_{t=1}^{T-1} \xi_t(i,j) + \bar{a}_{ij} - 1 \right)}, \tag{4.118}$$

$$\hat{\mu}_{il} = \frac{\sum_{t=1}^{T} u_{ilt} Y_{lt} + m_{il} - 1}{\sum_{t=1}^{T} u_{ilt} + s_{il}}, \tag{4.119}$$

$$\hat{\epsilon}_l = \frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) Y_{lt} + b_l - 1}{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) + c_l}, \tag{4.120}$$

and

$$\hat{\rho}_l = \frac{T + k_l - \sqrt{(T + k_l)^2 - 4k_l (\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt})}}{2k_l}. \tag{4.121}$$

4.4.3 Synthetic Data

As in previous sections, the Poisson FSHMM is tested on synthetic data. However, there is no published VB formulation so only the ML and MAP formulations are tested. In a later section, the Poisson FSHMM will be tested on a real data set along with the discrete non-parametric FSHMM. Again, the MAP-beta formulation is excluded as it has already been extensively compared to the MAP formulation.

Algorithm	$\hat{\pi}_1$	$\hat{\pi}_2$	$\hat{a}_{11}$	$\hat{a}_{12}$	$\hat{a}_{21}$	$\hat{a}_{22}$
ML	0	1	0.78	0.22	0.41	0.59
MAP	0.33	0.67	0.78	0.22	0.41	0.59

TABLE 4.17: Poisson parameter estimates for initial distribution and transition matrix. The prior in MAP affects the estimates for the initial distribution. The estimates for the transition probabilities are within 0.03 units of the true probability.

Algorithm	$\hat{\mu}_{11}$	$\hat{\mu}_{12}$	$\hat{\mu}_{21}$	$\hat{\mu}_{22}$
ML	10.00	20.04	49.77	99.86
MAP	9.99	20.04	49.76	99.84

TABLE 4.18: Poisson parameter estimates for μ for relevant features. All estimates are within 0.5 units of the true value.

Algorithm	$\hat{\epsilon}_3$	$\hat{\epsilon}_4$	$\hat{\epsilon}_5$
ML	30.28	29.78	30.04
MAP	29.95	29.82	30.11

TABLE 4.19: Poisson parameter estimates for ϵ for irrelevant features. All parameters are within 0.5 units of the true value.

An observation sequence of 2000 time steps is generated from a two state HMM. There are two relevant features with state-dependent Poisson distributions. Three irrelevant features generated from a single Poisson distribution with an expected value of 30 are added to the data, resulting in a model with five features in total. The model parameters are:

$$\mu_1 = \begin{bmatrix} 10 & 20 \end{bmatrix}, \quad \mu_2 = \begin{bmatrix} 50 & 100 \end{bmatrix}, \quad A = \begin{bmatrix} 0.75 & 0.25 \\ 0.4 & 0.6 \end{bmatrix}, \quad \pi = \begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix}.$$

The hyperparameters for the priors in the MAP formulation are: $\bar{p}_i = \bar{a}_{ij} = 2, s_{il} = c_l = 1$. b_l is the mean of the observations for the l^{th} feature, $m_{1l} = [5\ 15\ 25\ 30\ 35]$, and $m_{2l} = [40\ 90\ 35\ 30\ 25]$. For the feature saliencies, the weight parameter k_l is set to 50. ML and MAP are initialized with the same values: equal initial state probabilities and transition probabilities, $\mu_{init} = m, \epsilon_{init} = b$, and $\rho_{init} = 0.5$. The maximum number of iterations is 500 and the convergence threshold is 10^{-6}.

The ML method converged in 21 iterations while MAP converged in 184. The estimated parameters are in Tables 4.17 to 4.20. ML gives initial distribution estimates that match the training data. In spite of the training data starting in state 2, MAP gives initial distribution estimates that more closely reflect the true

Algorithm	$\hat{\rho}_1$	$\hat{\rho}_2$	$\hat{\rho}_3$	$\hat{\rho}_4$	$\hat{\rho}_5$
ML	1.0000	1.0000	0.3847	0.4999	0.3895
MAP	1.0000	1	0.0050	0.0105	0.0037

TABLE 4.20: Poisson parameter estimates for ρ of all features. ML overestimates the relevance of the irrelevant features, while MAP successfully identifies the relevant and irrelevant features.

parameters due to the prior distribution. Both algorithms give accurate estimates for the transition probabilities and the emission distribution parameters. As seen in previous formulations, ML overestimates ρ for irrelevant features, while the prior on the feature saliencies force the MAP estimates towards 0. Both formulations give accurate estimates for the feature saliencies of the relevant features.

4.5 Discrete Non-Parametric Features

Consider an HMM with I states and L features. For the discrete non-parametric features, let $P_{il}(\mathcal{Y})$ be the probability that the observation symbol from the l^{th} feature is from the set of possible observation symbols $\mathcal{Y}$, given $x = i$. The total number of possible observation symbols for the l^{th} feature is $\mathcal{P}_l$.

There are two possibilities for modeling the state-independent distribution. The state-independent distribution can be modeled as nonuniform and estimated by the algorithm; or it can be assumed that the state-independent distribution is uniform and set to a fixed value a priori. For the former, $q(Y_{lt}) = P_l(\mathcal{Y})$ and for the latter $q(Y_{lt}) = \mathcal{P}_l^{-1}$ (note that the conditional parameters have been dropped from the notation as these distributions are non-parametric).

The notation for $\mathbf{x}$ and $\mathbf{z}$ is the same as in previous sections. The conditional distribution for Y_t given $x_t = i$ and $\mathbf{z}$ is

$$P(y_t|\mathbf{z}, x_t = i, \Lambda) = \prod_{l=1}^{L} P_{il}(Y_{lt})^{z_l} q(Y_{lt})^{1-z_l}, \tag{4.122}$$

The marginal distribution for $\mathbf{z}$ is the same as Equation 3.2. The joint distribution of y_t and $\mathbf{z}$ given $x_t = i$ is the same as in Equation 3.3 with the appropriate substitutions used for $p(\cdot|\cdot)$ and $q(\cdot|\cdot)$. The joint distribution of $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ is the same as in Equation 3.5 with Equation 4.122 used for $P(y_t|\mathbf{z}, x_t = i, \Lambda)$.

The E-step probabilities are the same as in Equations 3.8 - 3.12 with the appropriate substitutions and $\gamma_t(i)$ and $\xi_t(i,j)$ calculated using the forward-backward algorithm (Equations 2.12 and 2.13). Derivations for the ML and MAP M-step parameter update equations are in Appendix A Section A.9.

4.5.1 M-Step ML

When $q(Y_{lt})$ is estimated from data, the ML M-step updates are

$$\hat{\pi}_i = \gamma_1(i), \tag{4.123}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \tag{4.124}$$

$$\hat{P}_{il}(\mathcal{Y}) = \frac{\sum_{t=1}^{T} u_{ilt} \mathbb{I}(\mathcal{Y} = Y_{lt})}{\sum_{t=1}^{T} u_{ilt}}, \tag{4.125}$$

$$\hat{P}_{l}(\mathcal{Y}) = \frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) \mathbb{I}(\mathcal{Y} = Y_{lt})}{\sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}}, \tag{4.126}$$

and

$$\begin{aligned} \hat{\rho}_l &= \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt}}{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt} + \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}} \\ &= \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} u_{i,l,t}}{T}, \end{aligned} \tag{4.127}$$

where $\mathbb{I}(\mathcal{Y} = Y_{lt}) = 1$ if $\mathcal{Y} = Y_{lt}$ and 0 otherwise. When the state-independent distribution is assumed to be uniform and not estimated, $\hat{P}_l(\mathcal{Y}) = \mathcal{P}_l^{-1}$. The other parameter update equations do not change and are the same as above.

4.5.2 M-Step MAP

The priors used for MAP estimation are listed below. Dir is the Dirichlet distribution, A_i is row i of the transition matrix and $\mathbf{P}_{il}$ is a vector representing every possible observation symbol for state i and the l^{th} feature.

$$\pi \sim \mathrm{Dir}(\pi|\bar{\boldsymbol{p}}), \tag{4.128}$$

$$A_i \sim \mathrm{Dir}(A_i|\bar{\boldsymbol{a}}_i), \tag{4.129}$$

$$\mathbf{P}_{il} \sim \mathrm{Dir}(\mathbf{P}_{il}|\mathbf{m}_{il}), \tag{4.130}$$

$$\mathbf{P}_i \sim \mathrm{Dir}(\mathbf{P}_l|\mathbf{b}_l), \tag{4.131}$$

$$\rho_l \sim \frac{1}{Z} e^{-k_l \rho_l}, \tag{4.132}$$

where Z is the normalizing constant. The parameter update equations are

$$\hat{\pi}_i = \frac{\gamma_1(i) + \bar{p}_i - 1}{\sum_{i=1}^{I} (\gamma_1(i) + \bar{p}_i - 1)}, \tag{4.133}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j) + \bar{a}_{ij} - 1}{\sum_{j=1}^{I} \left(\sum_{t=1}^{T-1} \xi_t(i,j) + \bar{a}_{ij} - 1 \right)}, \tag{4.134}$$

$$\hat{P}_{il}(\mathcal{Y}) = \frac{\sum_{t=1}^{T} u_{ilt} \mathbb{I}(\mathcal{Y} = Y_{lt}) + m_{il}(\mathcal{Y}) - 1}{\sum_{\mathcal{Y}} \left(\sum_{t=1}^{T} u_{ilt} \mathbb{I}(\mathcal{Y} = Y_{lt}) + m_{il}(\mathcal{Y}) - 1 \right)}, \tag{4.135}$$

$$\hat{P}_{l}(\mathcal{Y}) = \frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) \mathbb{I}(\mathcal{Y} = Y_{lt}) + b_l(\mathcal{Y}) - 1}{\sum_{\mathcal{Y}} \left(\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) \mathbb{I}(\mathcal{Y} = Y_{lt}) + b_l(\mathcal{Y}) - 1 \right)}, \tag{4.136}$$

and

$$\hat{\rho}_l = \frac{T + k_l - \sqrt{(T + k_l)^2 - 4k_l (\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt})}}{2k_l}. \tag{4.137}$$

As in the ML case, when the state-independent distribution is assumed to be uniform and not estimated, $\hat{P}_l(\mathcal{Y}) = \mathcal{P}_l^{-1}$. The prior on the state-independent distribution is omitted, but the rest of the parameter update equations are the same as above.

4.5.3 Synthetic Data

As in previous sections, the discrete non-parametric FSHMM is tested on synthetic data. Again, there is no published VB formulation so only the ML and MAP formulations are tested. For comparison, both the model estimating $P_l(\mathcal{Y})$ and the model treating it as uniform are evaluated.

An observation sequence of 2000 time steps is generated from a three state HMM. There are two relevant features with state-dependent discrete non-parametric distributions. Three irrelevant features generated from a uniform distribution are added to the data, resulting in a model with five features in total. Both the relevant and irrelevant features have 3 possible observation symbols. The model parameters are:

$$
\begin{aligned}
P_{11}(\mathcal{Y}) &= [0.95\ 0.025\ 0.025], & P_{12}(\mathcal{Y}) &= [0.99\ 0.005\ 0.005],\\
P_{21}(\mathcal{Y}) &= [0.01\ 0.98\ 0.01], & P_{22}(\mathcal{Y}) &= [0.025\ 0.95\ 0.025],\\
P_{31}(\mathcal{Y}) &= [0.025\ 0.025\ 0.95], & P_{32}(\mathcal{Y}) &= [0.01\ 0.01\ 0.98],
\end{aligned}
$$

$$
A = \begin{bmatrix} 0.55 & 0.25 & 0.2 \\ 0.3 & 0.5 & 0.2 \\ 0.15 & 0.25 & 0.6 \end{bmatrix}, \quad \pi = \begin{bmatrix} 0.4 \\ 0.4 \\ 0.2 \end{bmatrix}.
$$

The hyperparameters for the priors in the MAP formulation are $\beta_i = \alpha_{ij} = m_{il}(\mathcal{Y}) = b_l(\mathcal{Y}) = 2$. For the feature saliencies, the weight parameter k_l is set to 50. ML and MAP are initialized with the same values: equal initial state probabilities and transition probabilities are used. For the initial values of $P_{il}(\mathcal{Y})$, the symbol with the largest true probability for state i and the l^{th} feature is set to 0.9 and the other two symbols are set to 0.05. For the initial value of $P_l(\mathcal{Y})$ (when being estimated and not assumed to be uniform), the sum of the symbols divided by T is used. The maximum number of iterations is 500 and the convergence threshold is 10^{-6}.

The ML method converged in 53 iterations and ML assuming a uniform distribution for the state-independent distribution (designated ML-U) converged in 53 iterations. MAP converged in 310 iterations and MAP-U converged in 322 iterations. The estimated parameters are in Tables 4.21 to 4.25. There appears to be

Algorithm	$\hat{\pi}_1$	$\hat{\pi}_2$	$\hat{\pi}_3$
ML	0	0	1
ML-U	0	0	1
MAP	0.25	0.25	0.5
MAP-U	0.25	0.25	0.5

TABLE 4.21: Estimates for non-parametric feature FSHMM initial distribution. The prior in MAP affects the estimates of the initial distribution.

Algorithm	$\hat{a}_{11}$	$\hat{a}_{12}$	$\hat{a}_{13}$	$\hat{a}_{21}$	$\hat{a}_{22}$	$\hat{a}_{23}$	$\hat{a}_{31}$	$\hat{a}_{32}$	$\hat{a}_{33}$
ML	0.58	0.24	0.18	0.31	0.48	0.21	0.15	0.22	0.63
ML-U	0.58	0.24	0.18	0.30	0.49	0.21	0.15	0.22	0.63
MAP	0.58	0.24	0.18	0.31	0.48	0.21	0.15	0.22	0.63
MAP-U	0.58	0.24	0.18	0.31	0.48	0.21	0.15	0.22	0.63

TABLE 4.22: Estimates for non-parametric feature FSHMM transition matrix. All estiamtes are within 0.03 units of the true probability.

Algorithm	$\hat{P}_{1l}(1)$	$\hat{P}_{1l}(2)$	$\hat{P}_{1l}(3)$	$\hat{P}_{2l}(1)$	$\hat{P}_{2l}(2)$	$\hat{P}_{2l}(3)$	$\hat{P}_{3l}(1)$	$\hat{P}_{3l}(2)$	$\hat{P}_{3l}(3)$
ML F1	0.99	0	0.01	0	1.00	0	0.01	0	0.99
ML F2	1.00	0	0	0.03	0.95	0.02	0	0	1.00
ML-U F1	0.98	0.01	0.01	0	1.00	0	0.01	0.01	0.98
ML-U F2	1.00	0	0	0.03	0.94	0.02	0	0	1.00
MAP F1	0.95	0.03	0.02	0.01	0.98	0.01	0.02	0.03	0.95
MAP F2	0.98	0.01	0.01	0.04	0.94	0.02	0.01	0.01	0.98
MAP-U F1	0.95	0.03	0.02	0.01	0.98	0.01	0.02	0.03	0.95
MAP-U F2	0.98	0.01	0.01	0.04	0.94	0.02	0.01	0.01	0.98

TABLE 4.23: Estimates for non-parametric FSHMM $P_{il}(\mathcal{Y})$. F1 is feature 1 and F2 is feature 2. The estimates for the algorithm that assumes the state independent distribution is uniform are similar to the estimates for the algorithm that estimates the states independent distribution. The estimated probabilities are within 0.05 units of the true probabilities.

little difference between the methods that estimate $P_l(\mathcal{Y})$ and those that assume it is uniform. Further, the estimates for $P_l(\mathcal{Y})$ are close to a uniform distribution. These results are most likely biased by the fact that the irrelevant features are drawn from a uniform distribution.

ML and MAP give similar parameter estimates for $\mathbf{A}$, but the priors skew the results for π. For the relevant feature parameters, ML tends to overestimate some of the probabilities, while the use of priors in MAP prevents this. As seen in previous formulations, ML overestimates ρ for irrelevant features, while the prior on the feature saliencies force the MAP estimates towards 0. Both formulations give accurate estimates for the feature saliencies of the relevant features.

Algorithm	$\hat{P}_1(1)$	$\hat{P}_1(2)$	$\hat{P}_1(3)$	$\hat{P}_2(1)$	$\hat{P}_2(2)$	$\hat{P}_2(3)$	$\hat{P}_3(1)$	$\hat{P}_3(2)$	$\hat{P}_3(3)$
ML	0.34	0.33	0.33	0.35	0.32	0.33	0.34	0.32	0.34
MAP	0.34	0.33	0.33	0.34	0.32	0.34	0.33	0.33	0.34

TABLE 4.24: Estimates for non-parametric FSHMM $P_l(\mathcal{Y})$, the state-independent parameters. The estimates are within 0.02 units of the true value.

Algorithm	$\hat{\rho}_1$	$\hat{\rho}_2$	$\hat{\rho}_3$	$\hat{\rho}_4$	$\hat{\rho}_5$
ML	0.9785	0.9755	0.5004	0.5014	0.5012
ML-U	0.9662	0.9722	0.5004	0.5012	0.5010
MAP	1.0000	1.0000	8.8006×10^{-4}	0.0050	0.0030
MAP-U	1.0000	1.0000	6.7891×10^{-4}	0.0072	0.0025

TABLE 4.25: Estimates for non-parametric FSHMM feature saliencies. The ML formulation overestimates the relevance of the irrelevant features, while MAP successfully identifies the relevant and irrelevant features.

4.5.4 Cal IT Data

To test the discrete feature FSHMM formulation, the CalIT2 data set publicly available on UCI Machine Learning Repository [66] is modeled. The CalIT2 data set is comprised of counts for people entering and exiting a building through the main entrance over a 15 week period. Each count is aggregated over a half hour period, and each half hour is labeled with a time and date. The data includes a list of scheduled events with their dates and times. The objective for this data set is to detect the known events.

Ihler, Hutchins, and Smyth [46] model the CalIT2 data set using a joint Markov and Poisson process; where the day of the week and time of day are modeled as effects on the Poisson rate parameter, and the observations are the counts. An FSHMM is trained assuming the entering and exiting counts have a Poisson distribution, and the day of the week and time of day have discrete non-parametric distributions. As in [46], a two state Markov chain represents the presence of an event.

The method presented in [46] demonstrates remarkable accuracy at detecting the presence of an event, as well as having the ability to estimate the event attendance. The goal of this analysis is not to suggest the FSHMM is a better method for modeling this data set, but to demonstrate that discrete feature formulation of the FSHMM is possible and that features assuming different distributions can be modeled together. The FSHMM assumes that state 1 is the normal operation or no event, and state 2 is an event.

Ihler et al. [46] do not split the data into training and testing sets, and we will follow this procedure so our analysis is comparable to theirs. The models are trained on the entire data set and then Viterbi is used to detect the presence of events on the same data. The VB method in [137] only uses Gaussian distributions so we will omit the VB method, and only compare the ML, MAP and MAP-beta formulations.

The three formulations are initialized with the same values, where $\pi = [1; 0]$; the self transition of $\mathbf{A}$ is 0.9, and the transition to the other state is 0.1; and the Poisson parameters for the state representing no event are set to 1, and the parameters for an event are set to 20. The non-parametric probabilities are initialized so there is a larger probability of an event on weekdays, and from 8 o'clock in the morning to 8 in the evening:

$$P_{day} = \begin{bmatrix} 0.25 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.25 \\ 0.025 & 0.19 & 0.19 & 0.19 & 0.19 & 0.19 & 0.025 \end{bmatrix},$$

$$P_{time} = \begin{bmatrix} 0.0278 & \dots & 0.0139 & \dots & 0.0278 \\ 0.0139 & \dots & 0.0278 & \dots & 0.0139 \end{bmatrix}.$$

The state-independent Poisson parameters are set to the mean of the data and, as previously stated, the state-independent non-parametric probabilities are fixed to the inverse of the number of possible symbols (7^{-1} and 48^{-1}). As always, the initial feature saliencies are 0.5. For the MAP formulations, $\bar{a} = 2, \bar{p} = 1, \mathfrak{m} = \lambda_{init}, \mathfrak{s} = 1, \hat{p} = 2, \mathfrak{b} = \varepsilon_{init}$, and $\mathfrak{c} = 1$. For MAP using an exponential prior, k is equal to a quarter of the number of observations $k = 1260$. For MAP-beta, $k = 1$ and $\kappa = 1260$.

The estimated saliencies for the three features are displayed in Table 4.26. Reduced models were constructed by removing the feature with the lowest estimated ρ. For ML, the feature representing count entering the building was removed, while for MAP and MAP-beta the feature representing the day was removed.

The ML full and reduced models successfully detected 28 out of the 29 scheduled events within the time period the event occurred. The full and reduced models for both MAP formulations detected a schedule event in the time period the event occurred with 100% accuracy. There were several unscheduled events detected

Algorithm	Count In	Count Out	Day	Time
ML	0.7895	0.8335	0.9952	0.9993
MAP	0.6876	0.7650	0.0848	0.5372
MAP-beta	0.5083	0.5903	0.0782	0.3256

TABLE 4.26: Estimated feature saliencies for ML, MAP and MAP-beta. ML removes the count entering the building, while MAP and MAP-beta remove the day of the week.

with no method for validation. Ihler et al. [46] were also able to achieve 100% scheduled event detection using their most liberal choice of prior hyperparameters. Their method also detected numerous unscheduled events.

4.6 Chapter Conclusions

The primary conclusions from this chapter are listed below.

- Distributions other than a Gaussian can be used as state-dependent feature distributions for the FSHMM. The conditional feature distributions tested in this chapter are GMMs, exponential, gamma, Poisson and discrete non-parametric features.
- As the number of states and mixtures increase, sparsity of the data becomes an issue, which can lead to parameter estimates for the variance parameters of 0. Methods that use a prior on the variance are one solution to this issue.
- Newton's method with a barrier function can be used to solve for the parameters of the conditional gamma distribution used for the FSHMM.
- Models using different distributions for each feature can be trained. This is possible due to the assumption that each feature is independent.

Chapter 5

Feature Saliency Explicit Duration Hidden Markov Model

In this chapter, feature saliency is applied to a semi-Markov process; specifically, the explicit duration HMM outlined in Section 2.2.1. The emission distribution is assumed to be a single Gaussian, but any of the distributions outlined in Chapter 4 can be used. The steps of the EM algorithm for an ML and MAP formulation are derived, and the model is tested on synthetic data, the PHM data set, and the Kinect data set. This formulation is denoted as the feature saliency explicit duration hidden Markov model (FSEDHMM).

5.1 FSEDHMM

Consider a hidden semi-Markov model (HSMM) with I states. Let $\mathbf{y} = \{y_1, y_2, ..., y_T\}$ be the sequence of observed data, where each $y_t \in \mathbb{R}^L$. The observation for the l-th feature at time t, which is represented by the l-th component of y_t, is denoted by y_{lt}. Let $\mathbf{x} = \{x_1, x_2, ..., x_T\}$ be the unobserved state sequence. Assume that the hidden states follow a semi-Markov process, where the duration in a state is modeled as a truncated Poisson random variable with parameter λ. Let $\mathcal{D} = \{d_1, ..., d_{\mathcal{N}}\}$ be the sequence of duration random variables, where $\mathcal{N}$ is the number of sojourn periods or one plus the number of state transitions. Note that there are two sets of subscripts: t represents the time step, and n represents the n^{th} sojourn period. The duration of the n^{th} sojourn period can last for multiple time steps with a minimum value of d_{min} and a maximum value of d_{max}. The sum

of the individual duration random variables is equal to the total number of time steps $\sum_{n=1}^{\mathcal{N}} d_n = T$. The probability of duration d in state j is represented by $p_j(d)$. The explicit duration formulation assumes that state transition and duration are independent. The state transition for a semi-Markov model, as defined in Section 2.2, is split into separate components $a_{(i,d')(j,d)} = a_{i,j}p_j(d)$ when using the explicit duration assumption. The truncated Poisson distribution (Equation 2.69) is used for $p_j(d)$.

As with all saliency formulations, let $\mathbf{z} = \{z_1, \ldots, z_L\}$ be a set of binary variables indicating the relevancy of each feature. If $z_l = 1$, then the l-th feature is relevant. Otherwise, if $z_l = 0$ the l-th feature is irrelevant. The state-dependent and state-independent distributions are assumed to be Gaussian using the notation outlined in Section 3.1. The set of latent random variables for the FSEDHMM is $\{\mathbf{x},\mathbf{z},\mathcal{D}\}$, and the set of model parameters Λ is $\{\pi, A, \mu, \sigma, \rho, \epsilon, \tau, \omega\, \lambda\}$.

The marginal probability of $\mathbf{z}$ is the same as Equation 3.2, and the joint distribution of y_t and $\mathbf{z}$, given $\mathbf{x}$, $P(y_t, \mathbf{z}|x_t = i, \Lambda)$ is the same as Equation 3.3. The joint probability of all random variables is

$$P(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathcal{D}) = \pi_{x_1} p_{x_1}(d_1) \left[\prod_{\tau=1}^{d_1} P(y_\tau, \mathbf{z}|x_1 = i, \Lambda)\right] \left\{\prod_{n=2}^{\mathcal{N}} a_{x_{n-1},x_n} p_{x_n}(d_n) \left[\prod_{\tau=\hat{d}_n+1}^{\hat{d}_n+d_n} P(y_\tau, \mathbf{z}|x_n = i, \Lambda)\right]\right\}, \tag{5.1}$$

where

$$\hat{d}_n = \sum_{\hat{n}=1}^{n-1} d_{\hat{n}}. \tag{5.2}$$

5.2 EM Algorithm for FSEDHMM

The E-step is augmented from the standard FSHMM to include $\mathcal{D}$. The $\mathcal{Q}$ function is

$$\begin{aligned}\mathcal{Q}(\Lambda, \Lambda') &= \mathbb{E}[\log P(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathcal{D}|\Lambda)|\mathbf{y}, \Lambda'] \\ &= \sum_{\mathbf{x},\mathbf{z},\mathcal{D}} \log(P(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathcal{D}|\Lambda))P(\mathbf{x}, \mathbf{z}, \mathcal{D}|\mathbf{y}, \Lambda').\end{aligned} \tag{5.3}$$

The E-step probabilities are the same as Equations 3.8 through 3.12, with the Equation 2.37 used for $\gamma_t(j)$, and Equation 2.35 used for $\xi_t(i,j)$. In addition, $\eta_t(j,d)$ (Equation 2.34) must be calculated. The forward backward algorithm for an EDHMM is outlined in Section 2.2.1. Derivations for the $\mathcal{Q}$ function and ML and MAP M-step parameter update equations are in Appendix A Section A.10.

5.2.1 ML M-Step

The parameter estimates are the same as the single Gaussian formulation but include the duration parameter.

$$\hat{\pi}_i = \gamma_1(i), \tag{5.4}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \tag{5.5}$$

$$\hat{\mu}_{il} = \frac{\sum_{t=1}^{T} u_{ilt} y_{lt}}{\sum_{t=1}^{T} u_{ilt}}, \tag{5.6}$$

$$\hat{\sigma}_{il}^2 = \frac{\sum_{t=1}^{T} u_{ilt}(y_{lt} - \hat{\mu}_{il})^2}{\sum_{t=1}^{T} u_{ilt}}, \tag{5.7}$$

$$\hat{\epsilon}_l = \frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) y_{lt}}{\sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}}, \tag{5.8}$$

$$\hat{\tau}_l^2 = \frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right)(y_{lt} - \hat{\epsilon}_l)^2}{\sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}}, \tag{5.9}$$

$$\hat{\lambda}_i = \frac{\sum_{t=1}^{T} \sum_{d=d_{min}}^{D} \eta_t(i,d) d}{\sum_{t=1}^{T} \sum_{d=d_{min}}^{D} \eta_t(i,d)}, \tag{5.10}$$

and

$$\begin{aligned}\hat{\rho}_l &= \frac{\sum_{t=1}^{T}\sum_{i=1}^{I} u_{ilt}}{\sum_{t=1}^{T}\sum_{i=1}^{I} u_{ilt} + \sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}} \\ &= \frac{\sum_{t=1}^{T}\sum_{i=1}^{I} u_{i,l,t}}{T}.\end{aligned} \tag{5.11}$$

5.2.2 MAP M-step

The priors used for MAP estimation are listed below. Dir is the Dirichlet distribution, $\mathcal{N}$ is the Gaussian distribution, IG is the inverse gamma distribution, G is the gamma distribution and A_i is row i of the transition matrix.

$$\pi \sim \mathrm{Dir}(\pi|\bar{\boldsymbol{p}}), \tag{5.12}$$

$$A_i \sim \mathrm{Dir}(A_i|\bar{\boldsymbol{a}}_i), \tag{5.13}$$

$$\mu_{il} \sim \mathcal{N}(\mu_{il}|m_{il}, s_{il}^2), \tag{5.14}$$

$$\sigma_{il}^2 \sim \mathrm{IG}(\sigma_{il}^2|\zeta_{il}, \eta_{il}), \tag{5.15}$$

$$\epsilon_l \sim \mathcal{N}(\epsilon_l|b_l, c_l^2), \tag{5.16}$$

$$\tau_l^2 \sim \mathrm{IG}(\tau_l|\nu_l, \psi_l), \tag{5.17}$$

$$\lambda_i \sim \mathrm{G}(\lambda_i|o_i, \varpi_i), \tag{5.18}$$

$$\rho_l \sim \frac{1}{Z} e^{-k_l \rho_l}, \tag{5.19}$$

where Z is the normalizing constant.

The parameter update equations are

$$\hat{\pi}_i = \frac{\gamma_1(i) + \bar{p} - 1}{\sum_{i=1}^{I}\left(\gamma_1(i) + \bar{p}_i - 1\right)}, \tag{5.20}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1}\xi_t(i,j) + \bar{a}_{ij} - 1}{\sum_{j=1}^{I}\left(\sum_{t=1}^{T-1}\xi_t(i,j) + \bar{a}_{ij} - 1\right)}, \tag{5.21}$$

$$\hat{\mu}_{il} = \frac{s_{il}^2 \sum_{t=1}^{T} u_{ilt} y_{lt} + \hat{\sigma}_{il}^2 m_{il}}{s_{il}^2 \sum_{t=1}^{T} u_{ilt} + \hat{\sigma}_{il}^2}, \tag{5.22}$$

$$\hat{\sigma}_{il}^2 = \frac{\sum_{t=1}^{T} u_{ilt}(y_{lt} - \hat{\mu}_{il})^2 + 2\eta_{il}}{\sum_{t=1}^{T} u_{ilt} + 2(\zeta_{il} + 1)}, \tag{5.23}$$

$$\hat{\epsilon}_l = \frac{c_l^2 \sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) y_{lt} + \hat{\tau}_l^2 b_l}{c_l^2 \sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) + \hat{\tau}_l^2}, \tag{5.24}$$

$$\hat{\tau}_l^2 = \frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) (y_{lt} - \hat{\epsilon}_l)^2 + 2\psi_l}{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt}\right) + 2(\nu_l + 1)}, \tag{5.25}$$

$$\hat{\lambda}_i = \frac{\sum_{t=1}^{T} \sum_{d=d_{min}}^{D} \left(\eta_t(i,d)d\right) + o_i - 1}{\sum_{t=1}^{T} \sum_{d=d_{min}}^{D} \eta_t(i,d) + \varpi_i}, \tag{5.26}$$

and

$$\hat{\rho}_l = \frac{T + k_l - \sqrt{(T + k_l)^2 - 4k_l(\sum_{t=1}^{T} \sum_{i=1}^{I} u_{ilt})}}{2k_l}. \tag{5.27}$$

5.2.3 Implementation

HSMMs significantly increase the amount of computation needed for estimating model parameters and prediction compared to an HMM. The EDHMM assumption, that the state transition and duration are independent, decreases the amount of computation compared to the general HSMM. However, the computational load is still much greater than that of a standard HMM, the FSHMM, or some other formulations of the HSMM. For a comparison of the complexity and storage needs for different versions of HSMMs, see [133].

As with HMMs, multiplying long streams of probabilities quickly drive calculations below machine precision. For HMMs, this is addressed by scaling some probabilities. For HSMMs, this issue can be overcome for some data sets by using the predicted forward and backward probabilities (Equations 2.46 and 2.47). However, for larger data sets, the predicted probabilities do not completely alleviate the underflow problem. The E-step probabilities must be calculated in the log domain, and the log-sum-exponent trick must be used extensively.

The log-sum-exponent trick [80] is used for summing numerous small probabilities, where the individual probabilities could fall below machine precision, but the sum

is above machine precision. In generic terms, let $x_1, ..., x_N$ be a string of very small numbers, and let $z = \sum_{n=1}^{N} x_n$. Taking the log of both sides yields

$$\begin{aligned} \log z &= \log \left(\sum_{n=1}^{N} x_n \right) \\ &= \log \left(\sum_{n=1}^{N} e^{\log x_n} \right). \end{aligned} \tag{5.28}$$

Let $m = \max \log x_n$. The log of z can then be written as

$$\log z = m + \log \left(\sum_{n=1}^{N} e^{\log x_n - m} \right) \tag{5.29}$$

Equation 5.29 can be used to calculate the predicted probabilities without the individual probabilities dropping below machine precision and being rounded to 0.

5.3 Synthetic Data

Three observation sequences are produced by an EDHMM with two relevant features. The two dimensional vectors of relevant features are generated from $\mathcal{N}(\mu_i, \Sigma)$, and the duration periods are generated by $\text{Pois}(\lambda_i)$. The model has two states and the three sequences have 503, 505, and 502 time steps. The model parameters are:

$$\mu_1 = \begin{bmatrix} 10 & 20 \end{bmatrix}, \quad \mu_2 = \begin{bmatrix} 30 & 60 \end{bmatrix}, \quad \Sigma = \begin{bmatrix} 25 & 0 \\ 0 & 25 \end{bmatrix},$$

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \pi = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \lambda = \begin{bmatrix} 5 \\ 10 \end{bmatrix}.$$

Note that the transition matrix has no self transitions, which is a requirement of EDHMMs. Three irrelevant features of random noise, generated from $\mathcal{N}(0, I)$, are added to the data, resulting in a model with five features in total.

Algorithm	$\hat{\mu}_{11}$	$\hat{\mu}_{12}$	$\hat{\mu}_{21}$	$\hat{\mu}_{22}$
ML	10.0131	20.0315	30.1661	60.0931
MAP	10.0053	19.9857	30.0776	60.0755

TABLE 5.1: FSEDHMM parameter estimates for μ for relevant features. All estimates are within 0.2 units of the true value.

Algorithm	$\hat{\sigma}_{11}$	$\hat{\sigma}_{12}$	$\hat{\sigma}_{21}$	$\hat{\sigma}_{22}$
ML	4.6571	4.8262	4.8509	5.0238
MAP	4.6509	4.8040	4.9093	5.0000

TABLE 5.2: FSEDHMM parameter estimates for σ for relevant features. All estimates are within 0.5 units of the true value.

Algorithm	$\hat{\epsilon}_3$	$\hat{\epsilon}_4$	$\hat{\epsilon}_5$
ML	-0.0811	-0.0615	-0.0449
MAP	-0.0067	-0.0567	0.0001

TABLE 5.3: FSEDHMM parameter estimates for ϵ for irrelevant features. All estimates are within 0.1 units of the true value.

The hyperparameters for the priors in the MAP formulation are: $\bar{p}_i = \bar{a}_{ij} = 1$ and $s_{il} = \zeta_{il} = \eta_{il} = \nu_l = \psi_l = c_{il} = \varpi_i = 1$. b_l is the mean of the observations for the l^{th} feature. m_{il} is the true value minus 1. o_i for each state is equal to 6. For the feature saliencies, the weight parameter k_l is set to 50. The algorithms are initialized using the true values for π_i and a_{ij}. Previous tests on synthetic data have demonstrated that these parameters are easy to estimate, and the EDHMM formulation restricts the transition probabilities in a two state model. For initializing the other parameters, the following values are used: $\mu = m$, $\sigma = 4$, $\lambda = o_i$, $\epsilon = b$, and τ is the standard deviation of the data. The feature saliencies are always initially set to 0.5. The algorithms are run for a maximum of 500 iterations. Convergence is tested by calculating the absolute percent change in the likelihood for ML and the posterior probability for MAP. The convergence threshold is 10^{-6}. The maximum duration is 20 time steps and the minimum duration is 1 time step.

The model parameters estimated by ML and MAP are all listed in Tables 5.1 to 5.6. The estimates for the transition probabilities and the initial probabilities are omitted, because they are estimated exactly.

The ML and MAP formulations give similar parameter estimates that are close to the true values. As with the FSHMM, the primary difference between the two methods can be seen in the estimates for ρ. The relevant features all have

Algorithm	$\hat{\tau}_3$	$\hat{\tau}_4$	$\hat{\tau}_5$
ML	0.9653	0.9975	0.9453
MAP	1.0065	0.9860	0.9776

TABLE 5.4: FSEDHMM parameter estimates for τ for irrelevant features. All estimates are within 0.1 of the true value.

Algorithm	$\hat{\lambda}_1$	$\hat{\lambda}_2$
ML	5.1942	9.5588
MAP	5.1933	9.5136

TABLE 5.5: FSEDHMM parameter estimates for λ. All estimates are within 0.5 units of the true value.

Algorithm	$\hat{\rho}_1$	$\hat{\rho}_2$	$\hat{\rho}_3$	$\hat{\rho}_4$	$\hat{\rho}_5$
ML	0.9908	0.9984	0.2825	0.2826	0.2731
MAP	0.9953	0.9986	0.0022	0.0191	0.0100

TABLE 5.6: FSEDHMM parameter estimates for feature saliencies of all features. The ML formulation overestimates the relevance of the irrelevant features, while MAP successfully identifies the relevant and irrelevant features.

estimated ρ above 0.99. ML estimates higher ρ for the irrelevant features than MAP.

5.4 PHM Data

The ML and MAP formulations of the FSEDHMM are tested on the PHM data set. The experimental setup is same as described in Section 3.5. There is no version of the VB formulation for EDHMMs to test, and the MAP-beta formulation for the FSEDHMM is omitted. Only the 5 state model is evaluated.

The ML and MAP FSEDHMMs are initialized with the same values. The initial state probability and the transition matrices are not estimated. We assume a left-to-right model. For an EDHMM, a LTR model has a probability of 1 of transitioning to the next state, and a zero probability for all other state transitions. The initial distribution is also known to be 1 for the first state, and 0 for all other states, because we assume each tool is new and has no wear. The state-dependent means μ_{il} are equally spaced between -2 and 2. The state-dependent standard deviations σ_{il} is 1 for all states and features. The state-independent parameters are calculated from the training data. The initial values for λ_i are T/I. For MAP, the prior hyperparameters are $m = \mu_{init}$, $s = \zeta = \eta = \nu = \psi = 0.5$, $b = 0$, $c = 1$,

Algorithm and Tool	RMSE Full Model (μm)	RMSE Reduced Model (μm)	Smallest $\bar{\rho}$	Removed Feature
ML Tool 1	29.01	28.01	0.8366	Vibration Y
ML Tool 4	26.16	29.73	0.5762	Vibration Y
ML Tool 6	48.11	34.84	0.7700	Vibration Y
Average	34.43 $\pm$ 11.94	30.86 $\pm$ 3.55		
MAP Tool 1	17.83	17.83	0.3239	Force Y
MAP Tool 4	18.68	18.91	0.1768	Force Y
MAP Tool 6	33.51	33.94	0.2069	Force Y
Average	23.34 $\pm$ 8.82	23.56 $\pm$ 9.01		

TABLE 5.7: FSEDHMM results for 5 state PHM experiments. MAP removes force in the Y direction, while ML removes vibration in the Y direction which is less expensive than the force sensor. The average RMSE and $\pm$ 1 standard deviation are given for each formulation.

$o = \lambda_{init}$, and $\varpi = 1$. Half of the assumed cost of each sensor is assumed for k_l ($k_l = 1200$ for the force features and $k_l = 600$ for the vibration features). The convergence threshold for this experiment is 10^{-6}. The RMSE values for the full and reduced models are in Table 5.7 and plots of the estimated paths for the full and reduced models are in Figure 5.1.

The MAP formulation outperforms the ML formulation in terms of average RMSE and feature reduction. The ML formulation chooses vibration in the Y direction to be removed, and the MAP formulation chooses force in the Y direction. The average estimated ρ for the removed features are lower for MAP than for ML, which is to be expected due to the prior. The reduced model for ML performs better than the full while the full model slightly outperforms the reduced for MAP.

When comparing the FSEDHMM to the FSHMM results in Table 3.9, the ML FSEDHMM has a lower average RMSE for the full model, but a higher average RMSE for the reduced. The MAP FSEDHMM has a lower average RMSE for both the full and reduced models. The features selected for removal are the same for FSEDHMM and FSHMM. The EDHMM formulation improves on the single Gaussian HMM. However, the GMM emission formulation using MAP (Table 4.8) outperforms the MAP FSEDHMM. The predictive ability of the FSEDHMM on the PHM data could possibly be improved upon by using a GMM for the emission distribution, however, this would increase computation. The difference in the average RMSE of the FSEDHMM and FSHMM might not be significant; however, the MAP formulation continues to select the more expensive sensor for removal.

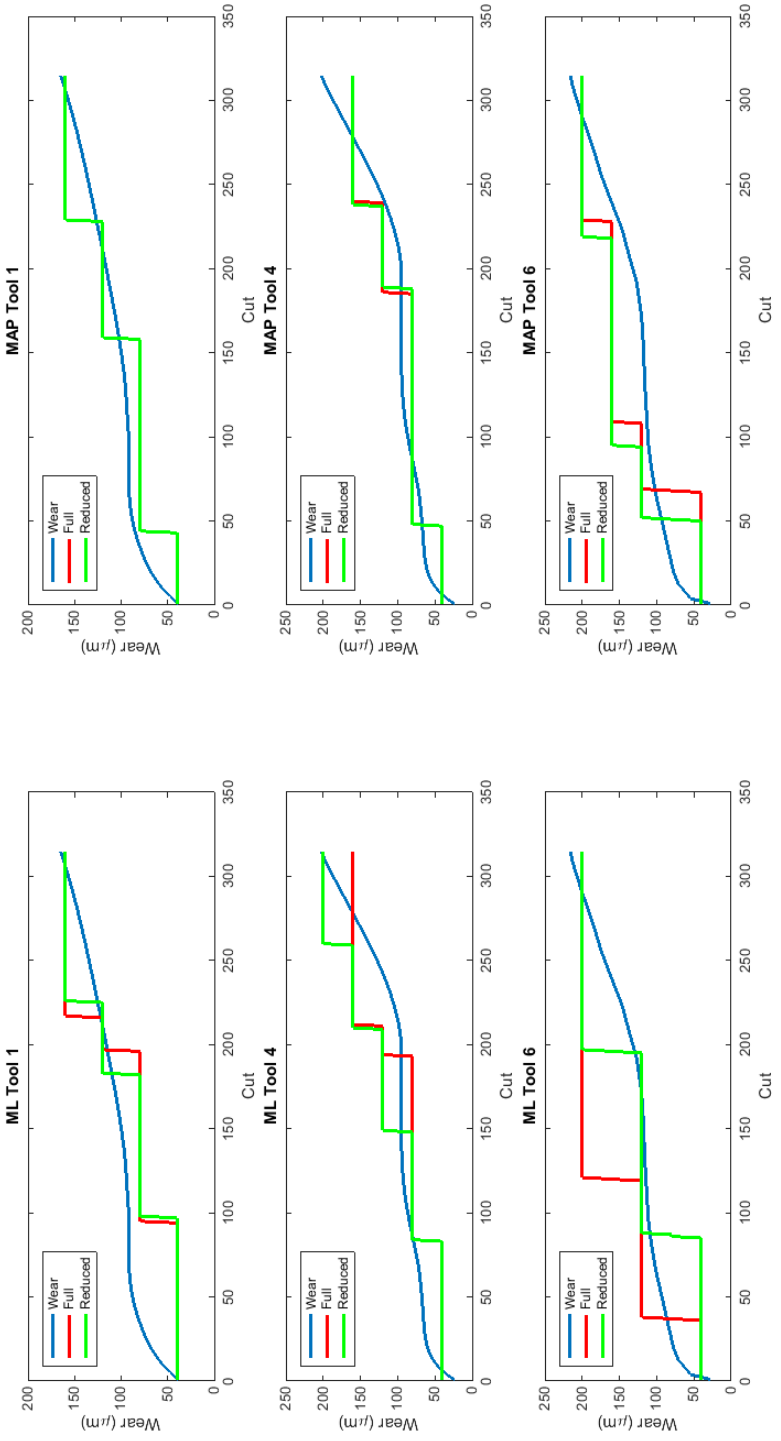

FIGURE 5.1: Plots for 5 state FSEDHMM model. Wear is the true wear measurement, Full is the full model using 18 features, Reduced is the reduced model using 15 features.

Algorithm and Tool	λ_1	λ_2	λ_3	λ_4	λ_5
ML 1 & 4	56.52	82.99	80.15	63.22	80.31
ML 1 & 6	68.79	75.84	61.20	48.04	61.19
ML 4 & 6	78.54	79.85	63.63	49.46	72.53
MAP 1 & 4	55.91	78.12	59.90	82.79	74.91
MAP 1 & 6	46.44	83.17	62.78	67.30	81.72
MAP 4 & 6	33.33	82.80	67.12	76.63	81.42

TABLE 5.8: FSEDHMM λ's for 5 state PHM experiments. In this table, the tools indicate the training set. These parameters can be interpreted as the average time in each state.

One advantage to the FSEDHMM is the ability to interpret the estimated values for λ as the average time in each state. For example, the estimated λ for ML using Tools 1 and 4 as the training data are [56.52, 82.99, 80.15, 63.22, 80.31]. On average, this model will spend 56 cuts in the first state. All values for the estimated λ are in Table 5.8. λ estimated by MAP more closely reflect a true wear curve than the ML estimates, because the average number of cuts spent in the first state, which can be interpreted as the initial wear-in state, is shorter than the average number of cuts in all other states.

5.5 Kinect Data

The FSEDHMM is also tested on the Kinect data set. The data are divided into training and testing sets as in all previous experiments: the first two thirds are used of training, the last third is reserved for testing, and the first 2000 observations of the training set are used for initializing the algorithm. This testing procedure only provides point estimates for the accuracy. Due to the significant increase in the computation required to train the FSEDHMM, the training set is broken into smaller sequences, so it can be processed in parallel. More specifically, the probabilities for each sequence in the E-step of the EM algorithm can be calculated independently and in parallel. Each sequence is approximately four minutes worth of data. Without parallel processing, the amount of time needed to train the FSEDHMM on the Kinect data set is on the order of months. However, there is a trade-off to segmenting the data sets in this fashion: it introduces discontinuities to the data. For the first observation in each of the smaller sequences, the algorithm has no knowledge of the data that came before it. For the last observation, there is no knowledge of the data that comes after it. As the number of sequences increases,

this affect on the data increases. We believe four minute sequences adequately balance the trade-off between the training time and adding discontinuities to the data.

ML and MAP are initialized with the same values. The state-dependent parameters $p(\cdot|\cdot)$ are initialized by calculating the state-dependent mean and standard deviation from the initialization set. The parameters for $q(\cdot|\cdot)$ are the means and standard deviations of the training data. The transition probabilities are initialized by counting the number of transition in the initialization set. The duration distribution parameters are calculated from the initialization set by averaging the time spent in each state before a transition. The MAP hyperparameters are $\bar{a} = 1, \bar{p} = 1, m = \mu_{init}, s = 0.25, \zeta = 0.25, \eta = 1, b = \epsilon_{init}, c = 0.5, \nu = 0.25, \psi = 0.5, o = \lambda_{init}, \varpi = 1$, and $k = 15,000$. In the MAP formulation, in addition to informative priors on μ, ϵ, and ρ, which are used in the FSHMM, we place an informative prior on λ. This is an informative prior because the hyperparameter o is calculated from the initialization set. The maximum duration is 2000, and the minimum duration is 30. These values are selected based on the range of times for each task. The maximum duration is just over one minute, the total work sequence should take around 30 seconds to complete so none of the primary tasks should last for one minute. The 'Other' state is the only possible task that can last longer than one minute but this rarely occurs. The minimum duration is one second, and each task should take at least this amount of time to complete. The convergence threshold for this experiment is increased to 10^{-4} in order to decrease training time.

The fraction of correctly classified observations, or the accuracy, for the ML full model is 0.8056, and the accuracy for the reduced model is 0.8067. The accuracies for the MAP full and reduced models are 0.8193 and 0.8037. The full MAP model gives the best accuracy, and reduced MAP gives the worst. The full and reduced models are less than 1% apart. Overall, the FSEDHMM performs better on the Kinect data than the FSHMM. ML removes 5 features, while MAP removes 19, one more feature than its FSHMM counterpart. This is possibly due to the higher convergence threshold. While the reduced MAP model does not perform as well as the full model or either ML model in terms of accuracy, it uses more than half the number of features with less than a 2% drop in accuracy.

As with the PHM data, we can examine the estimates for the duration parameters to gain understanding about the process. The parameter estimates for λ are

Algorithm	Fetch	Paint	Dry	Load	Walk	Other
ML	106.97	203.53	57.68	468.00	33.25	64.99
MAP	92.80	208.91	52.84	446.69	31.31	98.25

TABLE 5.9: FSEDHMM λ's for Kinect data. These parameters can be interpreted as the average time in each task.

displayed in Table 5.9. We see that on average 'Walk' takes the least amount of time. This makes sense the distance between the painting booth and the drying rack is about a step. On average, 'Load' takes the most time because the worker can remain at the loading rack for a significant amount of time performing several tasks.

5.6 Chapter Conclusions

The primary conclusions for this chapter are listed below.

- The feature saliency conditional distribution can be applied to HSMMs. This chapter specifically shows the implementation for an explicit duration model.
- The log-sum-exponent trick can be used to overcome precision issues when using the FSEDHMM. Further, predicted probabilities should be used for calculating the expectations in the E-step.
- The FSEDHMM performs better on both the PHM and Kinect data sets than the standard FSHMM using a single Gaussian distribution.
- As the number of states and the maximum duration increases, the computation required for training the FSEDHMM and predicting using Viterbi increases significantly. We address this issue by segmenting the Kinect training data set into smaller sequences, so that the E-step in EM can be performed in parallel.

Chapter 6

Conclusion and Future Work

In this chapter, conclusions about the model formulations and numerical experiments performed in Chapters 3, 4 and 5 are given, as well as ideas for future work stemming from the ideas proposed in this dissertation.

6.1 Conclusion

At the end of Chapters 3, 4 and 5 conclusions from the chapters are summarized as bullet points. In this section, they are restated and expanded upon.

6.1.1 FSHMM

In Chapter 3, the FSHMM is proposed, and a ML and two MAP formulations are derived. They are compared to a preexisting VB formulation and standard feature selection techniques. ML, MAP and MAP-beta use the expectation maximization algorithm to solve for model parameters. This is possible given that the number of hidden states in the Markov chain is finite and known a priori. Numerical experiments on synthetic data and two case study data sets demonstrate that the FSHMM can be used for simultaneous parameter estimation and feature selection for HMMs.

The primary difference between the four formulations is the use of informative priors. ML does not use priors on any model parameters. MAP and MAP-beta

use priors on all model parameters. The difference between these two formulations is the prior distribution used on the feature saliencies: MAP uses a truncated exponential while MAP-beta uses a beta distribution. VB classifies some parameters as random variables and assigns them prior distributions; but classifies others, including the feature saliencies, as model parameters and does not assign them prior distributions. We can now divided the four formulations into two classes: those that use informative priors on the feature saliencies (MAP and MAP-beta) and those that do not use priors on the feature saliencies (ML and VB).

The two FSHMM formulations using informative priors on the feature saliencies (MAP and MAP-beta) have the ability to incorporate the test cost of each feature into the feature selection process. There are two notions about test cost presented in this dissertation. The first assumes that the features have a financial cost associated with its use. For example, consider a system with two features where feature 1 costs twice as much as feature 2. When used as inputs to a classifier, if both features result in the same predictive ability of the classifier, the lower cost feature should be preferred. The MAP and MAP-beta formulations allow for the relative cost between features to be conveyed to the feature selection process through the informative prior distribution on the feature saliencies. In other systems, stakeholders might value a low cost sensor array over high predictive ability. This would establish a trade-off between these two factors. The priors could be used to convey the desired trade-off to the estimation algorithm.

The second notion about cost assumes that larger feature sets have a higher cost due to the increase in computation for prediction and storage requirements. This becomes crucial as the size of data sets, in terms of both the number of features and the number of observations, becomes large. Given similar predictive ability, a smaller feature set should be preferred. Again, the trade-off between accuracy and cost can be considered and conveyed to the learning algorithm through informative priors.

The synthetic data experiments demonstrate that ML tends to overestimate the relevance of irrelevant features, which leads to the inclusion of irrelevant features in a final feature subset. MAP-beta tends to underestimate the relevance of relevant features, which leads to the exclusion of relevant features from the final feature subset. VB provides accurate relevancy estimates for the features, but requires many more iterations in the training process and does not converge. Further, it is demonstrated that the feature saliency estimates for VB are affected by the

number of observations in the training set, i.e. a larger number of observations results in the over estimation of the relevance of irrelevant features.

When the four model formulations are compared on the PHM data, the MAP formulation outperforms ML, MAP-beta and VB in terms of accuracy and feature selection. MAP generally yields lower RMSE and consistently selects a lower cost feature subset. These experiments also demonstrate that the formulations that do not use priors on the feature saliencies give inconsistent feature subsets, which change with different training data sets. MAP and MAP-beta consistently select the same feature subset regardless of the training data. Further experiments on the PHM data demonstrate that a reduced conditional distribution, using only the parameters from the state-dependent distribution, can be used in place of the full conditional distribution.

The experiments on the Kinect data reiterate the results from the PHM experiments on a larger data set in terms of both the number of features and the number of observations. The reduced MAP model outperforms all other models and selects the smallest feature subset. MAP-beta's problem with underestimating relevant features is evident in this experiment, where all the features are removed from the model given a fixed removal threshold. This results in a significant drop in accuracy from the full model.

While this work does not provide a methodology for selecting hyperparameters (this is left to future work and expanded on in the next section) a heuristic for choosing the hyperparameters for the prior distribution on ρ is provided. Given no other information on the cost of features, $T/4$, where T is the number of observations in the training set, appears to be a good rule of thumb. The Kinect experiments use this rule, while the PHM experiments use the relative cost between features.

Generally, feature selection techniques attempt to remove redundant or highly correlated features. Synthetic data experiments demonstrate that redundant features can improve the predictive ability of an HMM; thus, a feature selection technique specifically for HMMs should not require the removal of redundant features. The FSHMM assumes that each feature is independent. In many applications, this assumption might not be reasonable. A methodology for dealing with correlated features for the FSHMM is still an open question and left to future work.

The MAP FSHMM also outperforms standard feature selection techniques. Supervised sequential searches are extremely sensitive to several factors, including the choice of the evaluation function and the training set, in terms of both accuracy and feature subset selection. The unsupervised sequential search methods are more robust, but do not perform as well and cannot incorporate cost into the feature selection process. The feature similarity method only produces a feature subset (it does not rank features so the user can control their removal) and does not incorporate cost. PCA is a feature extraction technique so it can reduce the dimension of the input, but all features must be collected in order to transform the data into the principal component space.

6.1.2 Other Feature Distributions

In Chapter 4, FSHMM formulations with feature distributions other than the standard Gaussian are derived and tested. The distributions explored in this chapter are mixtures of Gaussian distributions, exponential distributions, gamma distributions, Poisson distributions, and non-parametric discrete distributions. The GMM formulation is tested on synthetic data and the PHM data set. The numerical experiments demonstrate that the more complex model can improve the accuracy of the models over those explored in Section 3.5. However, as the number of states and mixtures increase, sparsity of data becomes an issue. This is illustrated by the ML formulation estimating the variance of some conditional distributions as 0. The models with priors do not have this problem, because the priors keep the estimate greater than 0.

The exponential and gamma FSHMMs are only tested on synthetic data. Testing these formulations on real world data is left to future work. There is no closed form solution for the shape parameter of the gamma distribution, so a Newton's method with a barrier function is implemented. The existing method outlined for mixtures of gammas use a simpler gradient descent technique.

Formulations using discrete features are derived for both a Poisson distribution and non-parametric distributions. An event detection data set is used for numerical experiments as well as synthetic data. For the event detection experiments, we demonstrate that the FSHMM can successfully model data with different assumed distributions. The Poisson distribution is used to model counts of people entering and exiting the building, and non-parametric distributions are used for features

representing the day of the week and time of day. The FSHMM achieves accuracy on par with the model previously published using the event detection data set.

6.1.3 FSEDHMM

In Chapter 5, a feature saliency model for a semi-Markov process is derived and tested. A semi-Markov process assumes that there is a sojourn or residence time to each state, meaning that state transitions do not need to occur every time step. There are several types of HSMMs, but the one studied in this dissertation is the explicit duration model. The EDHMM models the duration in each state as a random variable with a state-dependent distribution, and assumes that transitions and durations are independent. We use a truncated Poisson distribution to model the state duration. Semi-Markov processes generally perform well on data where the self-transition probabilities are high, indicating the process stays in the same state for a number of time steps. Further, the explicit duration assumption works well when the duration spent in each state is dependent on the current state.

Their are two primary implementation issues for the FSEDHMM, which is why semi-Markov models are not more widely used. First, numerical precision and underflow (probabilities that are non-zero being rounded to 0) are two issues that are not as straight forward to address as when using HMMs. The predicted probabilities can alleviate these issues when the number of states and the maximum duration are small – e.g. two states and a maximum duration of 3. When these become larger, the log-sum-exponent trick needs to be implemented. Second, the computational load for training an EDHMM is significant. We address this issue by segmenting the Kinect data set into smaller sequences, so they can be processed in parallel. This reduces the time for training the model from months to days. However, there is a trade-off to splitting the data into smaller segments: the introduction of discontinuities into the data. Dividing the data into more sequences allows for faster training time but can lead to less accurate parameter estimates. Another option is to increase the convergence threshold. Again, we trade faster training time for less accurate parameter estimates.

The MAP FSEDHMM performs better in terms of accuracy on both the PHM and Kinect data sets than the standard FSHMM. On the PHM data, the ML formulation removes a vibration sensor while MAP removes a force sensor. The FSEDHMM also removes one more feature on the Kinect data set than the FSHMM.

This is most likely due to the increase in the convergence threshold. In terms of feature subset selection, the FSEDHMM performs as good or better than the standard FSHMM. However, the increase in performance comes with a significant increase in training and prediction times.

6.2 Future Work

Future work concerning informative priors for HMMs and feature selection for HMMs can be broken into several categories outlined below.

- **Reduced Conditional Distribution** We outline two conditional distributions that can be used during prediction: the full conditional which uses the state-dependent distribution and the state-independent distribution, and the reduced conditional which uses only the state dependent distribution. We demonstrate on the PHM data that the reduced conditional can be used in place of the full conditional, reducing prediction times and parameter storage needs. However, it is not clear if the reduced conditional should be used in all cases or if it has significant benefits. Future work should assess the trade-off between these two conditional distributions, and explore this question on more data sets to investigate if a methodology for a picking a conditional distribution for prediction can be established.

- **Newton's Method for Gamma Features** In this work, a Newton's method for solving for the parameters of an FSHMM with an assumed gamma feature distribution is presented (Section 4.3.3). In the literature, there exists a gradient descent procedure for mixtures of gammas [2]. Future work would include a comparison of Newton's method to other optimization routines for solving for the model parameters of latent variable models with a gamma feature distribution. Comparisons should be made for mixtures of gammas, standard HMMs with gamma distributions, and the FSHMM with gamma features presented in thesis. Further, the numerical experiments in Section 4.3.4 are only performed on synthetic data. Future work should include experiments on real data sets.

- **Redundant Features** Even though we have demonstrated that redundant features are not necessarily a problem for HMMs, this is still a needed area

of research. The FSHMM assumes that each feature is independent. In practice, this is often not the case. We would expect features calculated from the same sensor or joints, such as the head and shoulders, to have non-zero correlations. By assuming the features in our test systems are independent, were are essentially acting contrary to some prior knowledge about the system. This is a modeling trade-off. We are neglecting some prior knowledge (feature correlation) to include other prior knowledge (feature selection that considers test cost). An embedded feature selection technique that can include cost and eliminate redundant features is an area of future research. This could be an addition to the FSHMM or a new method entirely.

- **Estimate Number of Hidden States** The work in this dissertation assumes that the number of hidden states in the model is known a priori. In some applications, this might not be possible. There are established techniques for estimating the number of states. However, most are similar to those used for greedy feature selection, where multiple models with different numbers of states are constructed, and an evaluation function is used to select the best model. The VB method in [137] can simultaneously estimate the number of hidden states or the number of mixtures. This option is lacking in the EM methods derived in this thesis. However, the VB method does not allow for the inclusion of cost and cannot estimate a LTR Markov chain. An FSHMM formulation that can include cost and estimate the number of states is an area of future work.

- **Hyperparmaeter Estimation and Selection** In this thesis, hyperparameters for all prior distributions are chosen using intuition and not a formal methodology. A heuristic for the saliency hyperparameter is proposed, but this is just a general starting point. Future work should develop a methodology for estimating the hyperparameters for all of the prior distributions used for the FSHMM. A significant portion of the future research should focus on converting cost, and other forms of prior knowledge, into a representative hyperparameter for the feature saliencies. Methods that can estimate the values of the hyperparameters from data should be explored.

- **Prior Distribution Selection Methodology** We study using an informative prior to incorporate cost into the feature selection process. Two priors on the feature saliencies, a beta distribution and an exponential distribution, are proposed and compared. Future work will investigate a methodology for

selecting the type of distribution used as a prior. The beta and exponential distributions are used to convey the cost of features. Other types of distributions could be used to convey different information such as physical properties or interactions. New priors on the other parameters in the model should be tested.

- **Modeling Trade-offs and Resource Allocation** Trade-offs when modeling systems with prior knowledge are an important consideration. Given a limited amount of time to study a system, the allocation of resources is important. For example, should one focus on learning as much as possible about the system before modeling begins, or should they focus on exploring all possible aspects of modeling. This is a trade-off between getting better priors and getting better models for the likelihood. Non-parametric Bayesian methods could provide better models for the likelihood. They have an infinite number of parameters, which significantly increases the amount of computation in the modeling step. Bayesian methods in general perform better when the priors give accurate information about the system. In a Bayesian setting, we now have two competing objectives: either make the priors as strong as possible or significantly increase the number of model parameters to better model the data. Non-parametric Bayesian methods with respect to informative priors is an area of future research.

Appendix A

Derivations

Appendix A outlines in detail the derivations discussed in the body of this thesis.

A.1 Derivations for the Joint and Marginal Distributions for FSHMM

Start with the conditional distribution of y_t given $\mathbf{z}$ and x_t, 3.1.

$$P(y_t|\mathbf{z}, x_t = i, \Lambda) = \prod_{l=1}^{L} p(y_{lt}|\mu_{il}, \sigma_{il}^2)^{z_l} q(y_{lt}|\epsilon_l, \tau_l^2)^{1-z_l}. \tag{A.1}$$

The joint distribution of y_t and $\mathbf{z}$ given the state is the product of $P(y_t|\mathbf{z}, x_t = i, \Lambda)$ and the marginal distribution of $\mathbf{z}$,

$$P(\mathbf{z} \,|\, \Lambda) = \prod_{l=1}^{L} \rho_l^{z_l} (1 - \rho_l)^{1-z_l}, \tag{A.2}$$

$$\begin{aligned} P(y_t, \mathbf{z}|x_t = i, \Lambda) &= P(y_t|\mathbf{z}, x_t = i, \Lambda) p(\mathbf{z} \,|\, \Lambda) \\ &= \left(\prod_{l=1}^{L} p(y_{lt}|\mu_{il}, \sigma_{il}^2)^{z_l} q(y_{lt}|\epsilon_l, \tau_l^2)^{1-z_l} \right) \prod_{l=1}^{L} \rho_l^{z_l} (1 - \rho_l)^{1-z_l} \\ &= \prod_{l=1}^{L} [\rho_l p(y_{lt}|\mu_{il}, \sigma_{il}^2)]^{z_l} [(1 - \rho_l) q(y_{lt}|\epsilon_l, \tau_l^2)]^{1-z_l}. \end{aligned} \tag{A.3}$$

The marginal distribution of y_t given the state can be found by summing the above equation over all values of $\mathbf{z}$.

$$\begin{aligned} P(y_t|x_t = i, \Lambda) &= \sum_{\mathbf{z}} P(y_t, \mathbf{z}|x_t = i, \Lambda) \\ &= \sum_{\mathbf{z}} \prod_{l=1}^{L} [\rho_l p(y_{lt}|\mu_{il}, \sigma_{il}^2)]^{z_l} [(1-\rho_l) q(y_{lt}|\epsilon_l, \tau_l^2)]^{1-z_l} \\ &= \prod_{l=1}^{L} \sum_{z_l=0}^{1} [\rho_l p(y_{lt}|\mu_{il}, \sigma_{il}^2)]^{z_l} [(1-\rho_l) q(y_{lt}|\epsilon_l, \tau_l^2)]^{1-z_l} \\ &= \prod_{l=1}^{L} \left(\rho_l p(y_{lt}|\mu_{il}, \sigma_{il}^2) + (1-\rho_l) q(y_{lt}|\epsilon_l, \tau_l^2)\right). \end{aligned} \tag{A.4}$$

A.2 FSHMM $\mathcal{Q}$ Function

The $\mathcal{Q}$ function is expanded into a sum of terms where each term can be maximized independently. The terms are the $\mathcal{Q}$ functions for the initial state π, the state transition matrix A, and the parameters for the emission distribution $\theta = \{\mu, \sigma, \epsilon, \tau, \rho\}$,

$$\mathcal{Q}(\Lambda, \Lambda') = \mathcal{Q}(\pi, \pi') + \mathcal{Q}(A, A') + \mathcal{Q}(\theta, \theta'). \tag{A.5}$$

The terms in this sum are given by

$$\begin{aligned} \mathcal{Q}(\pi, \pi') &= \sum_{\mathbf{x},\mathbf{z}} \log \pi_{x_1} P(\mathbf{x}, \mathbf{z}|\mathbf{y}, \Lambda') \\ &= \sum_{i=1}^{I} \log \pi_i P(x_1 = i|\mathbf{y}, \Lambda') \\ &= \sum_{i=1}^{I} \log \pi_i \gamma_1(i), \end{aligned} \tag{A.6}$$

$$\begin{aligned}\mathcal{Q}(A, A') &= \sum_{\mathbf{x},\mathbf{z}} \left(\sum_{t=2}^{T} \log a_{x_{t-1},x_t} \right) P(\mathbf{x}, \mathbf{z}|\mathbf{y}, \Lambda') \\ &= \sum_{i=1}^{I} \sum_{j=1}^{I} \sum_{t=1}^{T-1} \log a_{ij} P(x_t = i, x_{t+1} = j|\mathbf{y}, \Lambda') \\ &= \sum_{i=1}^{I} \sum_{j=1}^{I} \sum_{t=1}^{T-1} \log a_{ij} \xi_t(i, j),\end{aligned} \tag{A.7}$$

$$\begin{aligned}\mathcal{Q}(\theta, \theta') &= \sum_{\mathbf{x},\mathbf{z}} \left(\sum_{t=1}^{T} \log P(y_t, \mathbf{z}|\mathbf{x}) \right) P(\mathbf{x}, \mathbf{z}|\mathbf{y}, \Lambda') \\ &= \sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{l=1}^{L} \sum_{z_l=0}^{1} \Big[P(x_t = i, z_l|\mathbf{y}, \Lambda') \Big(z_l(\log \rho_l + \log p(y_{lt}|\mu_{il}, \sigma_{il}^2)) \\ &\quad + (1 - z_l)(\log(1 - \rho_l) + \log q(y_{lt}|\epsilon_l, \tau_l^2)) \Big) \Big] \\ &= \sum_{t=1}^{T} \Bigg[\sum_{i=1}^{I} \sum_{l=1}^{L} P(x_t = i, z_l = 1|\mathbf{y}, \Lambda') \log p(y_{lt}|\mu_{il}, \sigma_{il}^2) \\ &\quad + \sum_{i=1}^{I} \sum_{l=1}^{L} P(x_t = i, z_l = 0|\mathbf{y}, \Lambda') \log q(y_{lt}|\epsilon_l, \tau_l^2) \\ &\quad + \sum_{l=1}^{L} \Big(\log \rho_l \sum_{i=1}^{I} P(x_t = i, z_l = 1|\mathbf{y}, \Lambda') \\ &\quad + \log(1 - \rho_l) \sum_{i=1}^{I} P(x_t = i, z_l = 0|\mathbf{y}, \Lambda') \Big) \Bigg] \\ &= \sum_{t=1}^{T} \Bigg[\sum_{i=1}^{I} \sum_{l=1}^{L} u_{ilt} \log p(y_{lt}|\mu_{il}, \sigma_{il}^2) + \sum_{i=1}^{I} \sum_{l=1}^{L} v_{ilt} \log q(y_{lt}|\epsilon_l, \tau_l^2) \\ &\quad + \sum_{l=1}^{L} \Big(\log \rho_l \sum_{i=1}^{I} u_{ilt} + \log(1 - \rho_l) \sum_{i=1}^{I} v_{ilt} \Big) \Bigg].\end{aligned} \tag{A.8}$$

A.3 Derivation of ML EM Parameter Updates

Start with the $\mathcal{Q}$ function defined in the previous section. The parameters are separable so each part of the $\mathcal{Q}$ function can be maximized separately. Set the partial derivative with respect to the parameter being optimized equal to 0 to solve.

$$\begin{aligned}\frac{\partial}{\partial \pi_i}\mathcal{Q}(\pi, \pi') &= \frac{\partial}{\partial \pi_i}\left(\sum_{i=1}^{I} \log \pi_i \gamma_1(i)\right) \\ &= \frac{\gamma_1(i)}{\pi_i}.\end{aligned} \tag{A.9}$$

This process does not yield a feasible solution. The joint probability $P(x_1 = i, \mathbf{y}|\Lambda)$ is substituted for $P(x_1 = i|\mathbf{y}, \Lambda)$ and a Lagrange multiplier is added to ensure the contraint $\sum_{i=1}^{I} \pi_i = 1$.

$$\frac{\partial}{\partial \pi_i}\left[\sum_{i=1}^{I} \log \pi_i P(x_1 = i, \mathbf{y}|\Lambda) + \lambda\left(\sum_{i=1}^{I} \pi_i - 1\right)\right] = \frac{P(x_1 = i, \mathbf{y}|\Lambda)}{\pi_i} + \lambda, \tag{A.10}$$

$$\frac{P(x_1 = i, \mathbf{y}|\Lambda)}{\pi_i} + \lambda = 0, \tag{A.11}$$

$$\pi_i = \frac{P(x_1 = i, \mathbf{y}|\Lambda)}{-\lambda}. \tag{A.12}$$

Summing both sides over I yields $-\lambda = P(\mathbf{y}|\Lambda)$ giving

$$\begin{aligned}\pi_i &= \frac{P(x_1 = i, \mathbf{y}|\Lambda)}{P(\mathbf{y}|\Lambda)} \\ &= P(x_1 = i|\mathbf{y}, \Lambda) \\ &= \gamma_1(i).\end{aligned} \tag{A.13}$$

The maximization of $\mathcal{Q}(A, A')$ follows a similar logic and has a constraint that the rows of A must sum to 1.

$$\begin{aligned}&\frac{\partial}{\partial a_{ij}}\left[\sum_{i=1}^{I}\sum_{j=1}^{I}\sum_{t=1}^{T-1} \log a_{ij} P(x_{t-1} = i, x_t = j, \mathbf{y}|\Lambda') + \lambda\left(\sum_{j=1}^{I} a_{ij} - 1\right)\right] = \\ &\quad \frac{\sum_{t=1}^{T-1} P(x_{t-1} = i, x_t = j, \mathbf{y}|\Lambda')}{a_{ij}} + \lambda.\end{aligned} \tag{A.14}$$

Setting the derivative equal to 0 yields

$$a_{ij} = \frac{\sum_{t=1}^{T-1} P(x_{t-1} = i, x_t = j, \mathbf{y}|\Lambda')}{-\lambda}. \tag{A.15}$$

Summing both sides over I for all j gives $-\lambda = \sum_{t=1}^{T-1} P(x_{t-1} = i, \mathbf{y}|\Lambda')$.

$$\begin{aligned} a_{ij} &= \frac{\sum_{t=1}^{T-1} P(x_{t-1} = i, x_t = j, \mathbf{y}|\Lambda')}{\sum_{t=1}^{T-1} P(x_{t-1} = i, \mathbf{y}|\Lambda')} \\ &= \frac{\sum_{t=1}^{T-1} P(x_{t-1} = i, x_t = j|\mathbf{y}, \Lambda')}{\sum_{t=1}^{T-1} P(x_{t-1} = i|\mathbf{y}, \Lambda')} \\ &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j))}{\sum_{t=1}^{T-1} \gamma_t(i)}. \end{aligned} \tag{A.16}$$

The derivations for $\mu, \sigma, \epsilon, \tau$, and ρ all use $\mathcal{Q}(\theta, \theta')$. To find the updates, set the partial derivative with respect to the desired parameter equal to 0 and solve for the parameter. The parameters in $\mathcal{Q}(\theta, \theta')$ are separable so only the terms containing the desired parameter updates are required.

$$\begin{aligned} \frac{\partial}{\partial \mu_{il}} \mathcal{Q}(\theta, \theta') &= \frac{\partial}{\partial \mu_{il}} \left[\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{l=1}^{L} u_{ilt} \log p(y_{lt}|\mu_{il}, \sigma_{il}^2) \right] \\ &= \frac{\partial}{\partial \mu_{il}} \left[\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{l=1}^{L} u_{ilt} \left(-\frac{1}{2} \log(2\pi) - \log(\sigma_{il}) - \frac{(y_{lt} - \mu_{il})^2}{2\sigma_{il}^2} \right) \right] \\ &= \sum_{t=1}^{T} u_{ilt} \left(\frac{y_{lt} - \mu_{il}}{\sigma_{il}^2} \right), \end{aligned} \tag{A.17}$$

$$\sum_{t=1}^{T} u_{ilt} \left(\frac{y_{lt} - \mu_{il}}{\sigma_{il}^2} \right) = 0, \tag{A.18}$$

$$\sum_{t=1}^{T} u_{ilt} \mu_{il} = \sum_{t=1}^{T} u_{ilt} y_{lt}, \tag{A.19}$$

$$\mu_{il} = \frac{\sum_{t=1}^{T} u_{ilt} y_{lt}}{\sum_{t=1}^{T} u_{ilt}}. \tag{A.20}$$

$$\begin{aligned}
\frac{\partial}{\partial \sigma_{il}} \mathcal{Q}(\theta, \theta') &= \frac{\partial}{\partial \sigma_{il}} \left[\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{l=1}^{L} u_{ilt} \log p(y_{lt} | \mu_{il}, \sigma_{il}^2) \right] \\
&= \frac{\partial}{\partial \mu_{il}} \left[\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{l=1}^{L} u_{ilt} \left(-\frac{1}{2} \log(2\pi) - \log(\sigma_{il}) - \frac{(y_{lt} - \mu_{il})^2}{2\sigma^2} \right) \right] \\
&= \sum_{t=1}^{T} u_{ilt} \left(-\frac{1}{\sigma_{il}} + \frac{(y_{lt} - \mu_{il})^2}{\sigma_{il}^3} \right),
\end{aligned} \tag{A.21}$$

$$\sum_{t=1}^{T} u_{ilt} \left(-\frac{1}{\sigma_{il}} + \frac{(y_{lt} - \mu_{il})^2}{\sigma_{il}^3} \right) = 0, \tag{A.22}$$

$$\sum_{t=1}^{T} u_{ilt} \left(-1 + \frac{(y_{lt} - \mu_{il})^2}{\sigma_{il}^2} \right) = 0, \tag{A.23}$$

$$\sum_{t=1}^{T} u_{ilt} \frac{(y_{lt} - \mu_{il})^2}{\sigma_{il}^2} = \sum_{t=1}^{T} u_{ilt}, \tag{A.24}$$

$$\sigma^2 = \frac{\sum_{t=1}^{T} u_{ilt} (y_{lt} - \mu_{il})^2}{\sum_{t=1}^{T} u_{ilt}}. \tag{A.25}$$

$$\begin{aligned}
\frac{\partial}{\partial \epsilon_l} \mathcal{Q}(\theta, \theta') &= \frac{\partial}{\partial \epsilon_l} \left[\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{l=1}^{L} v_{ilt} \log q(y_{lt} | \epsilon_l, \tau_l^2) \right] \\
&= \frac{\partial}{\partial \epsilon_l} \left[\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{l=1}^{L} v_{ilt} \left(-\frac{1}{2} \log(2\pi) - \log(\tau_l) - \frac{(y_{lt} - \epsilon_l)^2}{2\tau_l^2} \right) \right] \\
&= \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} \left(\frac{y_{lt} - \epsilon_l}{\tau_l^2} \right),
\end{aligned} \tag{A.26}$$

$$\sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} \left(\frac{y_{lt} - \epsilon_l}{\tau_l^2} \right) = 0, \tag{A.27}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}\epsilon_l = \sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}y_{lt}, \tag{A.28}$$

$$\epsilon_l = \frac{\sum_{t=1}^{T}\big(\sum_{i=1}^{I} v_{ilt}\big)y_{lt}}{\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}}. \tag{A.29}$$

$$\begin{aligned}\frac{\partial}{\partial\tau_l}\mathcal{Q}(\theta,\theta') &= \frac{\partial}{\partial\tau_l}\left[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{l=1}^{L} v_{ilt}\log q(y_{lt}|\epsilon_l,\tau_l^2)\right]\\ &= \frac{\partial}{\partial\tau_l}\left[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{l=1}^{L} v_{ilt}\left(-\frac{1}{2}\log(2\pi)-\log(\tau_l)-\frac{(y_{lt}-\epsilon_l)^2}{2\tau_l^2}\right)\right]\\ &= \sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}\left(-\frac{1}{\tau_l}+\frac{(y_{lt}-\epsilon_l)^2}{\tau_l^3}\right),\end{aligned} \tag{A.30}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}\left(-\frac{1}{\tau_l}+\frac{(y_{lt}-\epsilon_l)^2}{\tau_l^3}\right) = 0, \tag{A.31}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}\left(-1+\frac{(y_{lt}-\epsilon_l)^2}{\tau_l^2}\right) = 0, \tag{A.32}$$

$$\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}\frac{(y_{lt}-\epsilon_l)^2}{\tau_l^2} = \sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}, \tag{A.33}$$

$$\tau^2 = \frac{\sum_{t=1}^{T}\big(\sum_{i=1}^{I} v_{ilt}\big)(y_{lt}-\epsilon_l)^2}{\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}}. \tag{A.34}$$

$$\begin{aligned}\frac{\partial}{\partial\rho_l}\mathcal{Q}(\theta,\theta') &= \frac{\partial}{\partial\rho_l}\left[\sum_{t=1}^{T}\sum_{l=1}^{L}\Big(\log\rho_l\sum_{i=1}^{I}u_{ilt}+\log(1-\rho_l)\sum_{i=1}^{I}v_{ilt}\Big)\right]\\ &= \frac{1}{\rho_l}\sum_{t=1}^{T}\sum_{i=1}^{I}u_{ilt}-\frac{1}{1-\rho_l}\sum_{t=1}^{T}\sum_{i=1}^{I}v_{ilt},\end{aligned} \tag{A.35}$$

$$\frac{1}{\rho_l}\sum_{t=1}^{T}\sum_{i=1}^{I}u_{ilt}-\frac{1}{1-\rho_l}\sum_{t=1}^{T}\sum_{i=1}^{I}v_{ilt} = 0, \tag{A.36}$$

$$(1-\rho_l)\sum_{t=1}^{T}\sum_{i=1}^{I}u_{ilt}-\rho_l\sum_{t=1}^{T}\sum_{i=1}^{I}v_{ilt}=0, \tag{A.37}$$

$$\rho_l\left(\sum_{t=1}^{T}\sum_{i=1}^{I}u_{ilt}+\sum_{t=1}^{T}\sum_{i=1}^{I}v_{ilt}\right)=\sum_{t=1}^{T}\sum_{i=1}^{I}u_{ilt}, \tag{A.38}$$

$$\rho_l=\frac{\sum_{t=1}^{T}\sum_{i=1}^{I}u_{ilt}}{\sum_{t=1}^{T}\sum_{i=1}^{I}u_{ilt}+\sum_{t=1}^{T}\sum_{i=1}^{I}v_{ilt}}. \tag{A.39}$$

A.4 Derivation of MAP EM Parameter Updates

For MAP estimation, add the log of the prior distribution to the $\mathcal{Q}$ function and perform the same steps as the previous section.

$$\begin{aligned}
&\frac{\partial}{\partial\pi_i}\left[\mathcal{Q}(\pi,\pi')+\log(G(\pi))+\lambda\left(\sum_{i=1}^{I}\pi_i-1\right)\right]\\
&\quad=\frac{\partial}{\partial\pi_i}\left[\sum_{i=1}^{I}\log\pi_i\gamma_1(i)-\log(B(\bar{\mathbf{p}})+\sum_{i=1}^{I}(\bar{p}_i-1)\log(\pi_i)+\lambda\left(\sum_{i=1}^{I}\pi_i-1\right)\right]\\
&\quad=\frac{\gamma_1(i)}{\pi_i}+\frac{\bar{p}_i-1}{\pi_i}+\lambda,
\end{aligned} \tag{A.40}$$

$$\frac{\gamma_1(i)}{\pi_i}+\frac{\bar{p}_i-1}{\pi_i}+\lambda=0, \tag{A.41}$$

$$\gamma_1(i)+\bar{p}_i-1=-\lambda\pi_i, \tag{A.42}$$

$$\pi_i=\frac{\gamma_1(i)+\bar{p}_i-1}{-\lambda}. \tag{A.43}$$

Summing both sides over all I yields

$$\pi_i=\frac{\gamma_1(i)+\bar{p}_i-1}{\sum_{i=1}^{I}\left(\gamma_1(i)+\bar{p}_i-1\right)}. \tag{A.44}$$

The derivations for the rest of the parameters are

$$\begin{aligned}
&\frac{\partial}{\partial a_{ij}}\left[\mathcal{Q}(A,A')+\log(G(A))+\lambda\left(\sum_{j=1}^{I}a_{ij}-1\right)\right]\\
&\quad=\frac{\partial}{\partial a_{ij}}\left[\sum_{i=1}^{I}\sum_{j=1}^{I}\sum_{t=1}^{T-1}\log a_{ij}\xi_t(i,j)+\sum_{i=1}^{I}\Bigg(-\log(B(\bar{\mathbf{a}}_i)\right.\\
&\qquad\left.+\sum_{j=1}^{I}(\bar{a}_{ij}-1)\log(a_{ij})\Bigg)+\lambda\left(\sum_{j=1}^{I}a_{ij}-1\right)\right]\\
&\quad=\frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{a_{ij}}+\frac{\bar{a}_{ij}-1}{a_{ij}}+\lambda,
\end{aligned}\tag{A.45}$$

$$\frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{a_{ij}}+\frac{\bar{a}_{ij}-1}{a_{ij}}+\lambda=0,\tag{A.46}$$

$$\sum_{t=1}^{T-1}\xi_t(i,j)+\bar{a}_{ij}-1=-\lambda a_{ij},\tag{A.47}$$

$$\begin{aligned}
a_{ij}&=\frac{\sum_{t=1}^{T-1}\xi_t(i,j)+\bar{a}_{ij}-1}{-\lambda}\\
&=\frac{\sum_{t=1}^{T-1}\xi_t(i,j)+\bar{a}_{ij}-1}{\sum_{j=1}^{I}\left(\sum_{t=1}^{T-1}\xi_t(i,j)+\bar{a}_{ij}-1\right)}.
\end{aligned}\tag{A.48}$$

$$\begin{aligned}
\frac{\partial}{\partial\mu_{il}}[\mathcal{Q}(\theta,\theta')+\log(G(\mu))]&=\frac{\partial}{\partial\mu_{il}}\left[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{l=1}^{L}u_{ilt}\log p(y_{lt}|\mu_{il},\sigma_{il}^2)+\log(G(\mu))\right]\\
&=\frac{\partial}{\partial\mu_{il}}\left[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{l=1}^{L}u_{ilt}\left(-\frac{1}{2}\log(2\pi)-\log(\sigma_{il})-\frac{(y_{lt}-\mu_{il})^2}{2\sigma_{il}^2}\right)\right.\\
&\quad\left.+\sum_{i=1}^{I}\sum_{l=1}^{L}\left(-\frac{1}{2}\log(2\pi)-\log(s_{il})-\frac{(\mu_{il}-m_{il})^2}{2s_{il}^2}\right)\right]\\
&=\sum_{t=1}^{T}u_{ilt}\left(\frac{y_{lt}-\mu_{il}}{\sigma_{il}^2}\right)-\frac{\mu_{il}-m_{il}}{s_{il}^2},
\end{aligned}\tag{A.49}$$

$$\sum_{t=1}^{T} u_{ilt}\left(\frac{y_{lt}-\mu_{il}}{\sigma_{il}^2}\right)-\frac{\mu_{il}-m_{il}}{s_{il}^2}=0, \tag{A.50}$$

$$s_{il}^2\sum_{t=1}^{T} u_{ilt}(y_{lt}-\mu_{il})-\sigma_{il}^2(\mu_{il}-m_{il})=0, \tag{A.51}$$

$$s_{il}^2\mu_{il}\sum_{t=1}^{T} u_{ilt}+\sigma_{il}^2\mu_{il}=s_{il}^2\sum_{t=1}^{T} y_{lt}u_{ilt}+\sigma_{il}^2 m_{il}, \tag{A.52}$$

$$\mu_{il}=\frac{s_{il}^2\sum_{t=1}^{T} y_{lt}u_{ilt}+\sigma_{il}^2 m_{il}}{s_{il}^2\sum_{t=1}^{T} u_{ilt}+\sigma_{il}^2}. \tag{A.53}$$

$$\begin{aligned}
\frac{\partial}{\partial\sigma_{il}}[\mathcal{Q}(\theta,\theta')+\log(G(\sigma))] &= \frac{\partial}{\partial\sigma_{il}}\left[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{l=1}^{L} u_{ilt}\log p(y_{lt}|\mu_{il},\sigma_{il}^2)+\log(G(\sigma))\right]\\
&= \frac{\partial}{\partial\sigma_{il}}\left[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{l=1}^{L} u_{ilt}\left(-\frac{1}{2}\log(2\pi)-\log(\sigma_{il})-\frac{(y_{lt}-\mu_{il})^2}{2\sigma_{il}^2}\right)\right.\\
&\quad\left.+\sum_{i=1}^{I}\sum_{l=1}^{L}\left(\zeta_{il}\log(\eta_{il})-\log(\Gamma(\zeta_{il}))+(-\zeta_{il}-1)\log(\sigma_{il}^2)-\frac{\eta_{il}}{\sigma_{il}^2}\right)\right]\\
&= \sum_{t=1}^{T} u_{ilt}\left(-\frac{1}{\sigma_{il}}+\frac{(y_{lt}-\mu_{il})^2}{\sigma_{il}^3}\right)-2\frac{\zeta_{il}+1}{\sigma_{il}}+2\frac{\eta_{il}}{\sigma_{il}^3},
\end{aligned} \tag{A.54}$$

$$\sum_{t=1}^{T} u_{ilt}\left(-\frac{1}{\sigma_{il}}+\frac{(y_{lt}-\mu_{il})^2}{\sigma_{il}^3}\right)-2\frac{\zeta_{il}+1}{\sigma_{il}}+2\frac{\eta_{il}}{\sigma_{il}^3}=0, \tag{A.55}$$

$$\sum_{t=1}^{T} u_{ilt}\left(-1+\frac{(y_{lt}-\mu_{il})^2}{\sigma_{il}^2}\right)-2(\zeta_{il}+1)+2\frac{\eta_{il}}{\sigma_{il}^2}=0, \tag{A.56}$$

$$\sigma_{il}^2\sum_{t=1}^{T} u_{ilt}+2\sigma_{il}^2(\zeta_{il}+1)=\sum_{t=1}^{T} u_{ilt}(y_{lt}-\mu_{il})^2+2\eta_{il}, \tag{A.57}$$

$$\sigma_{il}^2=\frac{\sum_{t=1}^{T} u_{ilt}(y_{lt}-\mu_{il})^2+2\eta_{il}}{\sum_{t=1}^{T} u_{ilt}+2(\zeta_{il}+1)}. \tag{A.58}$$

$$\begin{aligned}\frac{\partial}{\partial \epsilon_l}[\mathcal{Q}(\theta,\theta')+\log(G(\epsilon))] &= \frac{\partial}{\partial \epsilon_l}\left[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{l=1}^{L} v_{ilt}\log q(y_{lt}|\epsilon_l,\tau_l^2)+\log(G(\mu))\right]\\ &= \frac{\partial}{\partial \epsilon_l}\left[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{l=1}^{L} v_{ilt}\left(-\frac{1}{2}\log(2\pi)-\log(\tau_l)-\frac{(y_{lt}-\epsilon_l)^2}{2\tau_l^2}\right)\right.\\ &\quad \left.+\sum_{i=1}^{I}\sum_{l=1}^{L}\left(-\frac{1}{2}\log(2\pi)-\log(c_l)-\frac{(\epsilon_l-b_l)^2}{2c_l^2}\right)\right]\\ &= \sum_{i=1}^{I}\sum_{t=1}^{T} v_{ilt}\left(\frac{y_{lt}-\epsilon_l}{\tau_l^2}\right)-\frac{\epsilon_l-b_l}{c_l^2},\end{aligned} \tag{A.59}$$

$$\sum_{i=1}^{I}\sum_{t=1}^{T} v_{ilt}\left(\frac{y_{lt}-\epsilon_l}{\tau_l^2}\right)-\frac{\epsilon_l-b_l}{c_l^2}=0, \tag{A.60}$$

$$c_l^2\sum_{i=1}^{I}\sum_{t=1}^{T} v_{ilt}(y_{lt}-\epsilon_l)-\tau_l^2(\epsilon_l-b_l)=0, \tag{A.61}$$

$$c_l^2\epsilon_l\sum_{i=1}^{I}\sum_{t=1}^{T} v_{ilt}+\tau_l^2\epsilon_l = c_l^2\sum_{i=1}^{I}\sum_{t=1}^{T} y_{lt}v_{ilt}+\tau_l^2 b_l, \tag{A.62}$$

$$\epsilon_l = \frac{c_l^2\sum_{t=1}^{T} y_{lt}\left(\sum_{i=1}^{I} v_{ilt}\right)+\tau_l^2 b_l}{c_l^2\sum_{t=1}^{T}\left(\sum_{i=1}^{I} v_{ilt}\right)+\tau_l^2}. \tag{A.63}$$

$$\begin{aligned}\frac{\partial}{\partial \tau_l}[\mathcal{Q}(\theta,\theta')+\log(G(\tau))] &= \frac{\partial}{\partial \tau_l}\left[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{l=1}^{L} v_{ilt}\log q(y_{lt}|\epsilon_l,\tau_l^2)+\log(G(\tau))\right]\\ &= \frac{\partial}{\partial \tau_l}\left[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{l=1}^{L} v_{ilt}\left(-\frac{1}{2}\log(2\pi)-\log(\tau_l)-\frac{(y_{lt}-\epsilon_l)^2}{2\tau_l^2}\right)\right.\\ &\quad \left.+\sum_{i=1}^{I}\sum_{l=1}^{L}\left(\nu_l\log(\psi_l)-\log(\Gamma(\nu_l))+(-\nu_l-1)\log(\tau_l^2)-\frac{\psi_l}{\tau_l^2}\right)\right]\\ &= \sum_{i=1}^{I}\sum_{t=1}^{T} u_{ilt}\left(-\frac{1}{\tau_l}+\frac{(y_{lt}-\epsilon_l)^2}{\tau_l^3}\right)-2\frac{\nu_l+1}{\tau_l}+2\frac{\psi_l}{\tau_l^3},\end{aligned} \tag{A.64}$$

$$\sum_{i=1}^{I}\sum_{t=1}^{T} v_{ilt}\left(-\frac{1}{\tau_l} + \frac{(y_{lt}-\epsilon_l)^2}{\tau_l^3}\right) - 2\frac{\nu_l+1}{\tau_l} + 2\frac{\psi_l}{\tau_l^3} = 0, \tag{A.65}$$

$$\sum_{i=1}^{I}\sum_{t=1}^{T} v_{ilt}\left(-1 + \frac{(y_{lt}-\epsilon_l)^2}{\tau_l^2}\right) - 2(\nu_l+1) + 2\frac{\psi_l}{\tau_l^2} = 0, \tag{A.66}$$

$$\tau_l^2 \sum_{i=1}^{I}\sum_{t=1}^{T} v_{ilt} + 2\tau_l^2(\nu_l+1) = \sum_{t=1}^{T} v_{ilt}(y_{lt}-\epsilon_l)^2 + 2\psi_l, \tag{A.67}$$

$$\tau_l^2 = \frac{\sum_{t=1}^{T}\left(\sum_{i=1}^{I} v_{ilt}\right)(y_{lt}-\epsilon_l)^2 + 2\psi_l}{\sum_{t=1}^{T}\left(\sum_{i=1}^{I} v_{ilt}\right) + 2(\nu_l+1)}. \tag{A.68}$$

For the truncated exponential prior

$$\begin{aligned}
&\frac{\partial}{\partial\rho_l}[\mathcal{Q}(\theta,\theta') + \log(G(\rho))] \\
&\quad= \frac{\partial}{\partial\rho_l}\left[\sum_{t=1}^{T}\sum_{l=1}^{L}\Big(\log\rho_l\sum_{i=1}^{I}u_{ilt} + \log(1-\rho_l)\sum_{i=1}^{I}v_{ilt}\Big) + \log(G(\rho))\right] \\
&\quad= \frac{\partial}{\partial\rho_l}\left[\sum_{t=1}^{T}\sum_{l=1}^{L}\Big(\log\rho_l\sum_{i=1}^{I}u_{ilt} + \log(1-\rho_l)\sum_{i=1}^{I}v_{ilt}\Big) + \right. \\
&\qquad\left. + \sum_{l=1}^{L}\Big(-\log(Z_l) - k_l\rho_l\Big)\right] \\
&\quad= \frac{1}{\rho_l}\sum_{t=1}^{T}\sum_{i=1}^{I}u_{ilt} - \frac{1}{1-\rho_l}\sum_{t=1}^{T}\sum_{i=1}^{I}v_{ilt} - k_l,
\end{aligned} \tag{A.69}$$

$$\frac{1}{\rho_l}\sum_{t=1}^{T}\sum_{i=1}^{I}u_{ilt} - \frac{1}{1-\rho_l}\sum_{t=1}^{T}\sum_{i=1}^{I}v_{ilt} - k_l = 0, \tag{A.70}$$

$$k_l\rho_l^2 - \left(\sum_{t=1}^{T}\sum_{i=1}^{I}u_{ilt} + \sum_{t=1}^{T}\sum_{i=1}^{I}v_{ilt} + k_l\right)\rho_l + \sum_{t=1}^{T}\sum_{i=1}^{I}u_{ilt} = 0, \tag{A.71}$$

$$k_l\rho_l^2 - (T+k_l)\,\rho_l + \sum_{t=1}^{T}\sum_{i=1}^{I}u_{ilt} = 0. \tag{A.72}$$

The quadratic formula is used to find the root of Equation A.72

$$\hat{\rho}_l = \frac{T + k_l - \sqrt{(T + k_l)^2 - 4k_l(\sum_{t=1}^{T}\sum_{i=1}^{I} u_{ilt})}}{2k_l}. \tag{A.73}$$

The solution using + in the quadratic formula yields estimates for $\rho > 1$ so only the solution using − is used for the parameter estimate.

For the beta prior

$$\begin{aligned}
&\frac{\partial}{\partial \rho_l}[\mathcal{Q}(\theta, \theta') + \log(G(\rho))] \\
&\quad = \frac{\partial}{\partial \rho_l}\left[\sum_{t=1}^{T}\sum_{l=1}^{L}\Big(\log \rho_l \sum_{i=1}^{I} u_{ilt} + \log(1-\rho_l)\sum_{i=1}^{I} v_{ilt}\Big) + \log(G(\rho))\right] \\
&\quad = \frac{\partial}{\partial \rho_l}\left[\sum_{t=1}^{T}\sum_{l=1}^{L}\Big(\log \rho_l \sum_{i=1}^{I} u_{ilt} + \log(1-\rho_l)\sum_{i=1}^{I} v_{ilt}\Big) + \right. \\
&\qquad \left. + \sum_{l=1}^{L}\Big(-\log(\mathrm{B}(k_l, \kappa_l)) + (k_l - 1)\log \rho_l + (\kappa_l - 1)\log(1-\rho_l)\Big)\right] \\
&\quad = \frac{1}{\rho_l}\sum_{t=1}^{T}\sum_{i=1}^{I} u_{ilt} - \frac{1}{1-\rho_l}\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt} + \frac{k_l - 1}{\rho_l} - \frac{\kappa_l - 1}{1-\rho_l},
\end{aligned} \tag{A.74}$$

$$\frac{1}{\rho_l}\sum_{t=1}^{T}\sum_{i=1}^{I} u_{ilt} - \frac{1}{1-\rho_l}\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt} + \frac{k_l - 1}{\rho_l} - \frac{\kappa_l - 1}{1-\rho_l} = 0, \tag{A.75}$$

$$(1-\rho_l)\sum_{t=1}^{T}\sum_{i=1}^{I} u_{ilt} - \rho_l \sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt} + (1-\rho_l)(k_l - 1) - \rho_l(\kappa_l - 1) = 0, \tag{A.76}$$

$$\rho_l = \frac{\sum_{t=0}^{T}\sum_{i=1}^{I} u_{ilt} + k_l - 1}{T + k_l + \kappa_l - 2}. \tag{A.77}$$

A.5 Derivation of GMM Parameter Updates

The derivations for π_i and a_{ij} for ML and MAP are the same as in the single Gaussian case. The derivations for μ_{ilm}, σ_{ilm}, ϵ_l, τ_l, and ρ_l are trivial given the previous derivations in the single Gaussian case and the updated $\mathcal{Q}$ function given below, so they are omitted from this section.

Begin by redefining the $\mathcal{Q}$ function to include Φ.

$$
\begin{aligned}
\mathcal{Q}(\theta,\theta') &= \sum_{\mathbf{x},\mathbf{z},\Phi}\left(\sum_{t=1}^{T}\log P(y_t,\mathbf{z},\Phi|\mathbf{x})\right)P(\mathbf{x},\mathbf{z},\Phi|\mathbf{y},\Lambda') \\
&= \sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{l=1}^{L}\sum_{z_l=0}^{1}\sum_{m=1}^{M}\sum_{\phi_{mt}=0}^{1}\Big\{P(x_t=i,z_l,\phi_{mt}|\mathbf{y},\Lambda') \\
&\quad * \Big[\phi_{mt}\Big(\log\omega_{im}+z_l\big(\log\rho_l+\log p(y_{lt}|\mu_{ilm},\sigma^2_{ilm})\big) \\
&\quad + (1-z_l)\big(\log(1-\rho_l)+\log q(y_{lt}|\epsilon_l,\tau_l^2)\big)\Big)\Big]\Big\} \\
&= \sum_{t=1}^{T}\Bigg[\sum_{i=1}^{I}\sum_{m=1}^{M}\log\omega_{im}\sum_{l=1}^{L}P(x_t=i,z_l=1,\phi_{mt}=1|\mathbf{y},\Lambda') \\
&\quad + \sum_{i=1}^{I}\sum_{l=1}^{L}\sum_{m=1}^{M}P(x_t=i,z_l=1,\phi_{mt}=1|\mathbf{y},\Lambda')\log p(y_{lt}|\mu_{ilm},\sigma^2_{ilm}) \\
&\quad + \sum_{i=1}^{I}\sum_{l=1}^{L}\sum_{m=1}^{M}P(x_t=i,z_l=0,\phi_{mt}=1|\mathbf{y},\Lambda')\log q(y_{lt}|\epsilon_l,\tau_l^2) \\
&\quad + \sum_{l=1}^{L}\Bigg(\log\rho_l\sum_{i=1}^{I}\sum_{m=1}^{M}P(x_t=i,z_l=1,\phi_{mt}=1|\mathbf{y},\Lambda') \\
&\quad + \log(1-\rho_l)\sum_{i=1}^{I}\sum_{m=1}^{M}P(x_t=i,z_l=0,\phi_{mt}=1|\mathbf{y},\Lambda')\Bigg)\Bigg] \\
&= \sum_{t=1}^{T}\Bigg[\sum_{i=1}^{I}\sum_{m=1}^{M}\log\omega_{im}\sum_{l=1}^{L}u_{ilmt} \\
&\quad + \sum_{i=1}^{I}\sum_{l=1}^{L}\sum_{m=1}^{M}u_{ilmt}\log p(y_{lt}|\mu_{ilm},\sigma^2_{ilm}) + \sum_{i=1}^{I}\sum_{l=1}^{L}\sum_{m=1}^{M}v_{ilmt}\log q(y_{lt}|\epsilon_l,\tau_l^2) \\
&\quad + \sum_{l=1}^{L}\Bigg(\log\rho_l\sum_{i=1}^{I}\sum_{m=1}^{M}u_{ilmt}+\log(1-\rho_l)\sum_{i=1}^{I}\sum_{m=1}^{M}v_{ilmt}\Bigg)\Bigg].
\end{aligned}
\tag{A.78}
$$

For ML estimation, the update equation for ω_{im} is found using the first part of the $\mathcal{Q}$ function

$$\mathcal{Q}(\omega, \omega') = \sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{m=1}^{M} \log \omega_{im} \sum_{l=1}^{L} u_{ilmt}. \tag{A.79}$$

In order for $\sum_{m=1}^{M} \omega_{im} = 1$, a Lagrange multiplier is added to $\mathcal{Q}(\omega, \omega')$ when the partial derivative is taken and set to 0

$$\frac{\partial}{\partial \omega_{im}} \Big[\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{m=1}^{M} \log \omega_{im} \sum_{l=1}^{L} u_{ilmt} + \lambda \Big(\sum_{m=1}^{M} \omega_{im} - 1 \Big) = 0, \tag{A.80}$$

$$\frac{\sum_{t=1}^{T} \sum_{l=1}^{L} u_{ilmt}}{\omega_{im}} + \lambda = 0, \tag{A.81}$$

$$\omega_{im} = \frac{\sum_{t=1}^{T} \sum_{l=1}^{L} u_{ilmt}}{-\lambda}. \tag{A.82}$$

Summing both sides over M yields $-\lambda = \sum_{t=1}^{T} \sum_{l=1}^{L} \sum_{m=1}^{M} u_{ilmt}$ which gives

$$\omega_{im} = \frac{\sum_{t=1}^{T} \sum_{l=1}^{L} u_{ilmt}}{\sum_{t=1}^{T} \sum_{l=1}^{L} \sum_{m=1}^{M} u_{ilmt}}. \tag{A.83}$$

For MAP estimation, the prior is added to $\mathcal{Q}(\omega, \omega')$ with the Lagrange multiplier

$$\begin{aligned}
&\frac{\partial}{\partial \omega_{im}} \Big[\mathcal{Q}(\omega, \omega') + \log(G(\omega)) + \lambda \Big(\sum_{m=1}^{M} \omega_{im} - 1 \Big) \Big] \\
&\quad = \frac{\partial}{\partial \omega_{im}} \Big[\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{m=1}^{M} \log \omega_{im} \sum_{l=1}^{L} u_{ilmt} + \\
&\quad \sum_{i=1}^{I} \Big(- \log(B(\mathbf{w}_i)) + \sum_{m=1}^{M} (w_{im} - 1) \log(\omega_{im}) \Big) + \lambda \Big(\sum_{m=1}^{M} \omega_{im} - 1 \Big) \Big] \\
&\quad = \frac{\sum_{t=1}^{T} \sum_{l=1}^{L} u_{ilmt}}{\omega_{im}} + \frac{w_{im}}{\omega_{im}} + \lambda,
\end{aligned} \tag{A.84}$$

where $B(\mathbf{w}_i)$ is the Beta function. Solving for ω_{im} and summing both sides over M yields

$$\omega_{im} = \frac{\sum_{t=1}^{T}\sum_{l=1}^{L} u_{ilmt} + w_{im} - 1}{\sum_{m=1}^{M}\left(\sum_{t=1}^{T}\sum_{l=1}^{L} u_{ilmt} + w_{im} - 1\right)}. \tag{A.85}$$

A.6 Derivation of Exponential EM Parameter Updates

For the exponential FSHMM, all parameter update equations are the same as the single Gaussian case except those associated with the state dependent distribution $p(\cdot|\cdot)$. The only change to the $\mathcal{Q}$ function is making $p(\cdot|\cdot)$ an exponential distribution so the derivation of $\mathcal{Q}(\mu_{il}, \mu'_{il})$ is omitted.

For ML estimation, the derivation for the μ_{il} update equation is

$$\frac{\partial}{\partial \mu_{il}}\Big[\sum_{t=1}^{T} u_{ilt}(\log \mu_{il} - \mu_{il} y_{lt})\Big] = 0, \tag{A.86}$$

$$\frac{\sum_{t=1}^{T} u_{ilt}}{\mu_{il}} - \sum_{t=1}^{T} u_{ilt} y_{lt} = 0, \tag{A.87}$$

$$\mu_{il} = \frac{\sum_{t=1}^{T} u_{ilt}}{\sum_{t=1}^{T} u_{ilt} y_{lt}}. \tag{A.88}$$

For MAP estimation, the log of the prior is added

$$\begin{aligned}\frac{\partial}{\partial \mu_{il}}\Big[\sum_{t=1}^{T} u_{ilt}(\log \mu_{il} - \mu_{il} y_{lt}) + m_{il}\log(s_{il}) - \\ \log(\Gamma(m_{il})) + (m_{il}-1)\log(\mu_{il}) - \mu_{il} s_{il}\Big] = 0,\end{aligned} \tag{A.89}$$

$$\frac{\sum_{t=1}^{T} u_{ilt}}{\mu_{il}} - \sum_{t=1}^{T} u_{ilt} y_{lt} + \frac{m_{il}-1}{\mu_{il}} - s_{il} = 0, \tag{A.90}$$

$$\mu_{il} = \frac{\sum_{t=1}^{T} u_{ilt} + m_{il} - 1}{\sum_{t=1}^{T} u_{ilt} y_{lt} + s_{il}}. \tag{A.91}$$

A.7 Derivation of Gamma EM Parameter Updates

The parameter updates for π_i, a_{ij} and ρ_l are the same as in the single Gaussian case so their derivations are omitted from this section. When using a Gamma distribution, only the rate parameter has a closed form solution. The shape parameter is found using Newton's method and this is outlined in Section 4.3.3.

For ML estimation, the closed form solutions for σ_{il} and τ_l are found through the following derivations

$$\frac{\partial}{\partial \sigma_{il}} \Big[\sum_{t=1}^{T} u_{ilt} \big(\mu_{il} \log(\sigma_{il}) - \log(\Gamma(\mu_{il})) + (\mu_{il} - 1) \log(y_{lt}) - \sigma_{il} y_{lt} \big) \Big] = 0, \tag{A.92}$$

$$\frac{\mu_{il} \sum_{t=1}^{T} u_{ilt}}{\sigma_{il}} - \sum_{t=1}^{T} u_{ilt} y_{lt} = 0, \tag{A.93}$$

$$\sigma_{il} = \frac{\mu_{il} \sum_{t=1}^{T} u_{ilt}}{\sum_{t=1}^{T} u_{ilt} y_{lt}}. \tag{A.94}$$

$$\frac{\partial}{\partial \tau_{l}} \Big[\sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt} \big(\epsilon_{l} \log(\tau_{l}) - \log(\Gamma(\epsilon_{l})) + (\epsilon_{l} - 1) \log(y_{lt}) - \tau_{l} y_{lt} \big) \Big] = 0, \tag{A.95}$$

$$\frac{\epsilon_{l} \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}}{\tau_{l}} - \sum_{t=1}^{T} \Big(\sum_{i=1}^{I} v_{ilt} \Big) y_{lt} = 0, \tag{A.96}$$

$$\tau_{l} = \frac{\epsilon_{l} \sum_{t=1}^{T} \sum_{i=1}^{I} v_{ilt}}{\sum_{t=1}^{T} \big(\sum_{i=1}^{I} v_{ilt} \big) y_{lt}}. \tag{A.97}$$

For MAP estimation, the log of the prior is added

$$\begin{aligned}\frac{\partial}{\partial\sigma_{il}}\Big[\sum_{t=1}^{T} u_{ilt}\big(\mu_{il}\log(\sigma_{il}) - \log(\Gamma(\mu_{il})) + (\mu_{il}-1)\log(y_{lt}) - \sigma_{il}y_{lt}\big) \\ + m_{il}\log(s_{il}) - \log(\Gamma(m_{il})) + (m_{il}-1)\log(\mu_{il}) - s_{il}\mu_{il} \\ + \zeta_{il}\log(\eta_{il}) - \log(\Gamma(\zeta_{il})) + (\zeta_{il}-1)\log(\sigma_{il}) - \eta_{il}\sigma_{il}\Big] = 0,\end{aligned} \tag{A.98}$$

$$\frac{\mu_{il}\sum_{t=1}^{T} u_{ilt}}{\sigma_{il}} - \sum_{t=1}^{T} u_{ilt}y_{lt} + \frac{\zeta_{il}-1}{\sigma_{il}} - \eta_{il} = 0, \tag{A.99}$$

$$\sigma_{il} = \frac{\mu_{il}\sum_{t=1}^{T} u_{ilt} + \zeta_{il} - 1}{\sum_{t=1}^{T} u_{ilt}y_{lt} + \eta_{il}}. \tag{A.100}$$

$$\begin{aligned}\frac{\partial}{\partial\tau_{l}}\Big[\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}\big(\epsilon_{l}\log(\tau_{l}) - \log(\Gamma(\epsilon_{l})) + (\epsilon_{l}-1)\log(y_{lt}) - \tau_{l}y_{lt}\big) \\ + b_{l}\log(c_{l}) - \log(\Gamma(b_{l})) + (b_{l}-1)\log(\epsilon_{l}) - c_{l}\epsilon_{l} \\ + \nu_{l}\log(\psi_{l}) - \log(\Gamma(\nu_{l})) + (\nu_{l}-1)\log(\tau_{l}) - \psi_{l}\tau_{l}\Big] = 0,\end{aligned} \tag{A.101}$$

$$\frac{\epsilon_{l}\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}}{\tau_{l}} - \sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}y_{lt} + \frac{\nu_{l}-1}{\tau_{l}} - \psi_{l} = 0, \tag{A.102}$$

$$\tau_{l} = \frac{\epsilon_{l}\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt} + \nu_{l} - 1}{\sum_{t=1}^{T}\big(\sum_{i=1}^{I} v_{ilt}\big)y_{lt} + \psi_{l}}. \tag{A.103}$$

A.8 Derivation of Poisson EM Update Parameters

For the Poisson FSHMM, all parameter update equations for π, a_{ij} and ρ are the same as the single Gaussian case. The only changes to the $\mathcal{Q}$ function is making $p(\cdot|\cdot)$ and $q(\cdot|\cdot)$ Poisson distributions.

For ML estimation, the derivations for the μ_{il} and ϵ_{l} update equations are

$$\frac{\partial}{\partial \mu_{il}}\Big[\sum_{t=1}^{T} u_{ilt}(Y_{lt}\log\mu_{il} - \mu_{il} - \log(Y_{lt}!))\Big] = 0, \tag{A.104}$$

$$\frac{\sum_{t=1}^{T} u_{ilt}Y_{lt}}{\mu_{il}} - \sum_{t=1}^{T} u_{ilt} = 0, \tag{A.105}$$

$$\mu_{il} = \frac{\sum_{t=1}^{T} u_{ilt}Y_{lt}}{\sum_{t=1}^{T} u_{ilt}}. \tag{A.106}$$

$$\frac{\partial}{\partial \epsilon_{l}}\Big[\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}(Y_{lt}\log\epsilon_{l} - \epsilon_{l} - \log(Y_{lt}!))\Big] = 0, \tag{A.107}$$

$$\frac{\sum_{t=1}^{T}\big(\sum_{i=1}^{I} v_{ilt}\big)Y_{lt}}{\epsilon_{l}} - \sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt} = 0, \tag{A.108}$$

$$\epsilon_{l} = \frac{\sum_{t=1}^{T}\big(\sum_{i=1}^{I} v_{ilt}\big)Y_{lt}}{\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}}. \tag{A.109}$$

For MAP estimation, the log of the prior is added

$$\begin{aligned}\frac{\partial}{\partial \mu_{il}}\Bigg[&\Bigg(\sum_{t=1}^{T} u_{ilt}(Y_{lt}\log\mu_{il} - \mu_{il} - \log(Y_{lt}!))\Bigg)\\ &+ m_{il}\log s_{il} - \log(\Gamma(m_{il})) + (m_{il}-1)\log\mu_{il} - s_{il}\mu_{il}\Bigg] = 0,\end{aligned} \tag{A.110}$$

$$\frac{\sum_{t=1}^{T} u_{ilt}Y_{lt}}{\mu_{il}} - \sum_{t=1}^{T} u_{ilt} + \frac{m_{il}-1}{\mu_{il}} - s_{il} = 0, \tag{A.111}$$

$$\mu_{il} = \frac{\sum_{t=1}^{T} u_{ilt}Y_{lt} + m_{il} - 1}{\sum_{t=1}^{T} u_{ilt} + s_{il}}. \tag{A.112}$$

$$\frac{\partial}{\partial \epsilon_l}\Bigg[\Bigg(\sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt}(Y_{lt}\log\epsilon_l - \epsilon_l - \log(Y_{lt}!))\Bigg) + b_l\log c_l - \log(\Gamma(b_l)) + (b_l-1)\log\epsilon_l - c_l\epsilon_l\Bigg] = 0, \tag{A.113}$$

$$\frac{\sum_{t=1}^{T}\big(\sum_{i=1}^{I} v_{ilt}\big)Y_{lt}}{\epsilon_l} - \sum_{t=1}^{T}\sum_{i=1}^{I} v_{ilt} + \frac{b_l-1}{\epsilon_l} - c_l = 0, \tag{A.114}$$

$$\epsilon_l = \frac{\sum_{t=1}^{T}\big(\sum_{i=1}^{I} v_{ilt}\big)Y_{lt} + b_l - 1}{\sum_{t=1}^{T}\big(\sum_{i=1}^{I} v_{ilt}\big) + c_l}. \tag{A.115}$$

A.9 Derivation of Discrete Non-Parametric EM Update Parameters

For the discrete non-parametric FSHMM, all parameter update equations for π, a_{ij} and ρ are the same as the single Gaussian case. The only changes to the $\mathcal{Q}$ function is making $p(\cdot|\cdot)$ and $q(\cdot|\cdot)$ discrete non-parametric distributions. To ensure that $\sum_{\mathcal{Y}} P_{il}(\mathcal{Y}) = 1$, a Lagrange multiplier is added to the $\mathcal{Q}$ function.

For ML estimation, the derivations for the $P_{il}(\mathcal{Y})$ and $P_l(\mathcal{Y})$ update equations are

$$\frac{\partial}{\partial P_{il}(\mathcal{Y})}\left[\sum_{t=1}^{T} u_{ilt}\mathbb{I}(\mathcal{Y}=Y_{lt})\log P_{il}(\mathcal{Y}) + \lambda\left(\sum_{\mathcal{Y}} P_{il}(\mathcal{Y}) - 1\right)\right] = 0, \tag{A.116}$$

$$\frac{\sum_{t=1}^{T} u_{ilt}\mathbb{I}(\mathcal{Y}=Y_{lt})}{P_{il}(\mathcal{Y})} + \lambda = 0, \tag{A.117}$$

$$P_{il}(\mathcal{Y}) = \frac{\sum_{t=1}^{T} u_{ilt}\mathbb{I}(\mathcal{Y}=Y_{lt})}{-\lambda}. \tag{A.118}$$

Summing both sides over $\mathcal{Y}$, leads to $\lambda = -\sum_{\mathcal{Y}}\left(\sum_{t=1}^{T} u_{ilt}\mathbb{I}(\mathcal{Y}=Y_{lt})\right) = -T$.

$$P_{il}(\mathcal{Y}) = \frac{\sum_{t=1}^{T} u_{ilt}\mathbb{I}(\mathcal{Y} = Y_{lt})}{T}. \tag{A.119}$$

$$\frac{\partial}{\partial P_l(\mathcal{Y})}\left[\sum_{t=1}^{T} big(\sum_{i=1}^{I} v_{ilt})\mathbb{I}(\mathcal{Y} = Y_{lt}) \log P_l(\mathcal{Y}) + \lambda\left(\sum_{\mathcal{Y}} P_l(\mathcal{Y}) - 1\right)\right] = 0, \tag{A.120}$$

$$\frac{\sum_{t=1}^{T}\left(\sum_{i=1}^{I} v_{ilt}\right)\mathbb{I}(\mathcal{Y} = Y_{lt})}{P_l(\mathcal{Y})} + \lambda = 0, \tag{A.121}$$

$$P_l(\mathcal{Y}) = \frac{\sum_{t=1}^{T}\left(\sum_{i=1}^{I} v_{ilt}\right)\mathbb{I}(\mathcal{Y} = Y_{lt})}{-\lambda}. \tag{A.122}$$

Summing both sides over $\mathcal{Y}$, leads to $\lambda = -\sum_{\mathcal{Y}}\left(\sum_{t=1}^{T}\left(\sum_{i=1}^{I} v_{ilt}\right)\mathbb{I}(\mathcal{Y} = Y_{lt})\right) = -T$.

$$P_l(\mathcal{Y}) = \frac{\sum_{t=1}^{T}\left(\sum_{i=1}^{I} v_{ilt}\right)\mathbb{I}(\mathcal{Y} = Y_{lt})}{T}. \tag{A.123}$$

For MAP estimation, the log of the prior is added

$$\begin{aligned}\frac{\partial}{\partial P_{il}(\mathcal{Y})}\Bigg[&\sum_{t=1}^{T} u_{ilt}\mathbb{I}(\mathcal{Y} = Y_{lt}) \log P_{il}(\mathcal{Y}) \\ &- \log(\mathrm{B}(\mathbf{m}_{il})) + \sum_{\mathcal{Y}}(m_{il}(\mathcal{Y}) - 1)\log P_{il}(\mathcal{Y}) \\ &+ \lambda\left(\sum_{\mathcal{Y}} P_{il}(\mathcal{Y}) - 1\right)\Bigg] = 0,\end{aligned} \tag{A.124}$$

$$\frac{\sum_{t=1}^{T} u_{ilt}\mathbb{I}(\mathcal{Y} = Y_{lt})}{P_{il}(\mathcal{Y})} + \frac{m_{il}(\mathcal{Y}) - 1}{P_{il}(\mathcal{Y})} + \lambda = 0, \tag{A.125}$$

$$P_{il}(\mathcal{Y}) = \frac{\sum_{t=1}^{T} u_{ilt}\mathbb{I}(\mathcal{Y} = Y_{lt}) + m_{il}(\mathcal{Y}) - 1}{-\lambda}. \tag{A.126}$$

Summing both sides over $\mathcal{Y}$, leads to
$\lambda = -\sum_{\mathcal{Y}} \left(\sum_{t=1}^{T} u_{ilt} \mathbb{I}(\mathcal{Y} = Y_{lt}) + m_{il}(\mathcal{Y}) - 1 \right)$ which gives

$$P_{il}(\mathcal{Y}) = \frac{\sum_{t=1}^{T} u_{ilt} \mathbb{I}(\mathcal{Y} = Y_{lt}) + m_{il}(\mathcal{Y}) - 1}{\sum_{\mathcal{Y}} \left(\sum_{t=1}^{T} u_{ilt} \mathbb{I}(\mathcal{Y} = Y_{lt}) + m_{il}(\mathcal{Y}) - 1 \right)}. \tag{A.127}$$

$$\begin{aligned} \frac{\partial}{\partial P_l(\mathcal{Y})} \Bigg[& \sum_{t=1}^{T} \Big(\sum_{i=1}^{I} v_{ilt} \Big) \mathbb{I}(\mathcal{Y} = Y_{lt}) \log P_l(\mathcal{Y}) \\ & - \log(\mathrm{B}(\mathbf{b}_l)) + \sum_{\mathcal{Y}} (b_l(\mathcal{Y}) - 1) \log P_l(\mathcal{Y}) \\ & + \lambda \left(\sum_{\mathcal{Y}} P_l(\mathcal{Y}) - 1 \right) \Bigg] = 0, \end{aligned} \tag{A.128}$$

$$\frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) \mathbb{I}(\mathcal{Y} = Y_{lt})}{P_l(\mathcal{Y})} + \frac{b_l(\mathcal{Y}) - 1}{P_l(\mathcal{Y})} + \lambda = 0, \tag{A.129}$$

$$P_l(\mathcal{Y}) = \frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) \mathbb{I}(\mathcal{Y} = Y_{lt}) + b_l(\mathcal{Y}) - 1}{-\lambda}. \tag{A.130}$$

Summing both sides over $\mathcal{Y}$, leads to
$\lambda = -\sum_{\mathcal{Y}} \left(\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) \mathbb{I}(\mathcal{Y} = Y_{lt}) + b_l(\mathcal{Y}) - 1 \right)$ which gives

$$P_l(\mathcal{Y}) = \frac{\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) \mathbb{I}(\mathcal{Y} = Y_{lt}) + b_l(\mathcal{Y}) - 1}{\sum_{\mathcal{Y}} \left(\sum_{t=1}^{T} \left(\sum_{i=1}^{I} v_{ilt} \right) \mathbb{I}(\mathcal{Y} = Y_{lt}) + b_l(\mathcal{Y}) - 1 \right)}. \tag{A.131}$$

A.10 Derivation of FSEDHMM Parameter Updates

As with the FSHMM, the $\mathcal{Q}$ function can be split into several parts separating the parameter associated with the Markov chain from those associated with the state duration and the emission distribution.

$$\mathcal{Q}(\Lambda, \Lambda') = \mathcal{Q}(\pi, \pi') + \mathcal{Q}(A, A') + \mathcal{Q}(\theta, \theta'), \tag{A.132}$$

where $\theta = \{\mu, \sigma, \epsilon, \tau, \rho, \lambda\}$. Expanding Equation 5.3 using the appropriate substitutions for the FSEDHMM and separating the parameters yields the following $\mathcal{Q}$ functions:

$$\begin{aligned}\mathcal{Q}(\pi, \pi') &= \sum_{\mathbf{x},\mathbf{z},\mathcal{D}} \log(\pi_1) P(\mathbf{x}, \mathbf{z}, \mathcal{D}|\mathbf{y}, \Lambda') \\ &= \sum_{i=1}^{I} \log(\pi_i) P(x_1 = i|\mathbf{y}, \Lambda') \\ &= \sum_{i=1}^{I} \log(\pi_i) \gamma_1(i),\end{aligned} \tag{A.133}$$

$$\begin{aligned}\mathcal{Q}(A, A') &= \sum_{\mathbf{x},\mathbf{z},\mathcal{D}} \left(\sum_{n=2}^{\mathcal{N}} \log(a_{x_{n-1},x_n}) \right) P(\mathbf{x}, \mathbf{z}, \mathcal{D}|\mathbf{y}, \Lambda') \\ &= \sum_{i=1}^{I} \sum_{j=1}^{I} \sum_{t=1}^{T-1} \log(a_{i,j}) P(x_{t]} = i, x_{[t+1} = j|\mathbf{y}, \Lambda') \\ &= \sum_{i=1}^{I} \sum_{j=1}^{I} \sum_{t=1}^{T-1} \log(a_{i,j}) \xi_t(i, j),\end{aligned} \tag{A.134}$$

$$\begin{aligned}\mathcal{Q}(\theta, \theta') &= \sum_{\mathbf{x},\mathbf{z},\mathcal{D}} \sum_{n=1}^{\mathcal{N}} \left(\log(p_{x_n}(d_n)) + \sum_{\tau=\hat{d}_n+1}^{\hat{d}_n+d_n} \log(P(y_\tau, \mathbf{z}|x_n = i, \Lambda') \right) P(\mathbf{x}, \mathbf{z}, \mathcal{D}|\mathbf{y}, \Lambda') \\ &= \sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{d=d_{min}}^{D} \log(p_{x_t}(d)) P(x_{[t-d+1:t]} = i|\mathbf{y}, \Lambda') \\ &\quad + \sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{l=1}^{L} \sum_{z_l=0}^{1} \Big[P(x_t = i, z_l|\mathbf{y}, \Lambda') \Big(z_l(\log \rho_l + \log p(y_{lt}|\mu_{il}, \sigma_{il}^2)) \\ &\quad + (1 - z_l)(\log(1 - \rho_l) + \log q(y_{lt}|\epsilon_l, \tau_l^2)) \Big) \Big].\end{aligned} \tag{A.135}$$

From this point, $\mathcal{Q}(\theta, \theta')$ can be separated into two summations. The first only has parameters pertaining to the duration distribution. The derivation for this part can be finished by substituting $\eta_t(i, d)$ for $P(x_{[t-d+1:t]} = i|\mathbf{y}, \Lambda')$. The second

is identical to the $\mathcal{Q}$ function for the FSHMM which has already been derived in the Appendices.

The update for all parameters except λ are the same as previously derived for the FSHMM. The ML update equation is can be found by setting the derivative of the first part of $\mathcal{Q}(\theta, \theta')$ equal to 0.

$$\frac{\partial}{\partial \lambda_i}\left[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{d=d_{min}}^{D}\log(p_{x_t}(d))\eta_t(i,d)\right] = 0, \tag{A.136}$$

$$\frac{\partial}{\partial \lambda_i}\left[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{d=d_{min}}^{D}\left(\lambda_i \log(d) - \lambda_i - \log(d!)\right)\eta_t(i,d)\right] = 0, \tag{A.137}$$

$$\sum_{t=1}^{T}\sum_{d=d_{min}}^{D}\left(\frac{\lambda_i}{d} - 1\right)\eta_t(i,d) = 0, \tag{A.138}$$

$$\lambda_i = \frac{\sum_{t=1}^{T}\sum_{d=d_{min}}^{D}\left(\eta_t(i,d)d\right)}{\sum_{t=1}^{T}\sum_{d=d_{min}}^{D}\eta_t(i,d)}. \tag{A.139}$$

The MAP update equation is found by adding the prior then setting the derivative equal to 0.

$$\begin{aligned}\frac{\partial}{\partial \lambda_i}\Bigg[\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{d=d_{min}}^{D}&\left(\lambda_i \log(d) - \lambda_i - \log(d!)\right)\eta_t(i,d) \\ &+ o_i\log(\varpi_i) - \log(\Gamma(o_i)) + (o_i - 1)\log(\lambda_i) - \varpi_i\lambda_i\Bigg] = 0,\end{aligned} \tag{A.140}$$

$$\sum_{t=1}^{T}\sum_{d=d_{min}}^{D}\left(\frac{\lambda_i}{d} - 1\right)\eta_t(i,d) + \frac{o_i - 1}{\lambda_i} - \varpi_i = 0, \tag{A.141}$$

$$\lambda_i = \frac{\sum_{t=1}^{T}\sum_{d=d_{min}}^{D}\left(\eta_t(i,d)d\right) + o_i - 1}{\sum_{t=1}^{T}\sum_{d=d_{min}}^{D}\eta_t(i,d) + \varpi_i}. \tag{A.142}$$

Appendix B

Sequential Search Results

This Appendix contains all of the sequential search method results from Chapter 3. The first column labeled "Test" contains the tool used for testing. The second column labeled "Eval" contains the tools used for evaluation. The third column labeled "Train" contains the tools used for training the model and "U" designates all unsupervised tools (Tools 2, 3, and 5). The column labeled either "Added" or "Removed" contains a list of the features or sensors added or removed from the model in the order they were added or removed from the model. The final two columns contain the results of testing the selected features on the test tool. The evaluation function used during the evaluation step is used as the performance metric. The performance metric followed by a "1" is the model using only the training data when building the model for testing. The performance metric followed by a "2" is the model using the training and evaluation data for testing.

B.1 Sequential Forward Search

Test	Eval	Train	Added	RMSE1	RMSE2
1	4 6	U	VZ_rmse VZ_sle FX_sle VZ_me	23.2850	27.3782
4	1 6	U	FY_sle FX_rmse FZ_sle	29.0999	30.7657
6	1 4	U	VY_sle	46.7726	46.9585

Table B.1: Experiment 1 SFS

Test	Eval	Train	Added	RMSE1	RMSE2
1	4	U 6 A	VX_sle VZ_sle VY_rms VX_me VY_me	19.0689	17.9445
1	4	U 6 BA	VX_sle VZ_sle VY_me FY_rms VY_sle FY_me	18.3028	18.0947
1	4	U 6 BBA	VX_sle VZ_sle VY_me FY_me VY_sle FY_rms	18.3028	18.0947
1	4	U 6 BBB	VX_sle VZ_sle VY_me FY_me VY_sle FZ_rms	18.3028	16.8768
1	6	U 4	FZ_sle VZ_sle VX_me	24.5273	26.8862
4	1	U 6	VY_sle	17.0147	17.1893
4	6	U 1	FY_sle FZ_sle FY_me	26.2982	30.2979
6	1	U 4	VY_sle VY_rms	42.5751	47.3041
6	4	U 1 A	VY_sle VY_me VX_sle	34.2946	37.9831

Table B.2: Experiment 2 SFS

Test	Eval	Train	Added	RMSE1	RMSE2
1	4 6	U	VZ	21.9622	26.7892
4	1 6	U	FZ	24.2417	23.4235
6	1 4	U	FY FX	24.5433	24.7485

Table B.3: Experiment 3 SFS

Test	Eval	Train	Added	RMSE1	RMSE2
1	4	U 6	VZ VY	29.3652	18.2035
1	6	U 4	VZ	25.4810	26.7892
4	1	U 6	FY FX	22.6151	21.2156
4	6	U 1	VZ	18.8469	19.7496
6	1	U 4	FY FX	25.1434	24.7485
6	4	U 1	VX VY	36.1040	39.3351

Table B.4: Experiment 4 SFS

Test	Eval	Train	Added	RMSE1	RMSE2
1	4 6	U	VX_sle FY_sle VZ_sle VZ_me FZ_sle VZ_rms	0.6399	0.5225
4	1 6	U	FY_sle FX_rms VX_sle VZ_sle VX_me	0.7281	0.7216
6	1 4	U	VY_sle	1.1778	1.1805

Table B.5: Experiment 5 SFS

Test	Eval	Train	Added	RMSE1	RMSE2
1	4	U 6 AA	VX_sle VZ_sle VY_rms FY_rms	0.5225	0.3695
1	4	U 6 AB	VX_sle VZ_sle VY_rms VX_me	0.5225	0.5133
1	4	U 6 BA	VX_sle VZ_sle VY_me FY_rms	0.5164	0.3695
1	4	U 6 BB	VX_sle VZ_sle VY_me FY_me	0.5195	0.4880
1	6	U 4	FZ_sle VZ_sle VX_me	0.6249	0.6667
4	1	U 6	VY_sle	0.5040	0.5008
4	6	U 1	VX_me	0.9759	0.5434
6	1	U 4	VY_sle	1.1738	1.1805
6	4	U 1 A	VY_sle VY_me VZ_rms	0.8747	0.9984
6	4	U 1 B	VY_sle VY_me VX_sle	0.8819	0.9578

Table B.6: Experiment 6 SFS

Test	Eval	Train	Added	RMSE1	RMSE2
1	4 6	U	VZ	0.5801	0.6619
4	1 6	U	FZ FX VY	0.9050	0.9103
6	1 4	U	VX VY	0.8927	0.9791

Table B.7: Experiment 7 SFS

Test	Eval	Train	Added	RMSE1	RMSE2
1	4	U 6	VX VY	0.4472	0.3780
1	6	U 4	VZ	0.6325	0.6619
4	1	U 6	FY FX	0.7389	0.6878
4	6	U 1	VZ	0.6449	0.6325
6	1	U 4	FY FX	0.6947	0.6878
6	4	U 1	VX VY	0.9241	0.9791

Table B.8: Experiment 8 SFS

Test	Eval	Train	Added	Acc1	Acc2
1	4 6	U	VZ_rms	0.5651	0.5206
4	1 6	U	FY_sle FX_rms VX_sle VZ_sle VX_me	0.4984	0.4794
6	1 4	U	VX_sle	0.4603	0.3810

Table B.9: Experiment 9 SFS

Test	Eval	Train	Added	Acc1	Acc2
1	4	U 6 AA	VX_sle VZ_sle VY_rms FY_rms	0.7270	0.8635
1	4	U 6 AB	VX_sle VZ_sle VY_rms VX_me	0.7270	0.7365
1	4	U 6 BA	VX_sle VZ_sle VY_me FY_rms	0.7333	0.8635
1	4	U 6 BB	VX_sle VZ_sle VY_me FY_me	0.7302	0.7619
1	6	U 4	FZ_sle VZ_sle VX_me	0.6095	0.5556
4	1	U 6	VY_sle	0.7460	0.7492
4	6	U 1	VX_me VZ_sle FZ_sle	0.4413	0.5968
6	1	U 4	VY_sle	0.2794	0.2730
6	4	U 1	VY_rms VZ_rms VY_sle VX_me	0.3302	0.3302

Table B.10: Experiment 10 SFS

Test	Eval	Train	Added	Acc1	Acc2
1	4 6	U	VZ	0.6635	0.5619
4	1 6	U	FZ FY FX VY	0.4730	0.4190
6	1 4	U	VY VX	0.3270	0.2222

Table B.11: Experiment 11 SFS

B.2 Sequential Backward Search

Test	Eval	Train	Added	Acc1	Acc2
1	4	U 6	VX VY	0.8000	0.8571
1	6	U 4	VZ	0.6000	0.5619
4	1	U 6	FY FX	0.4540	0.5270
4	6	U 1	VZ	0.5841	0.6000
6	1	U 4	FY FX	0.5175	0.5270
6	4	U 1	VX VY	0.3079	0.2222

Table B.12: Experiment 12 SFS

Test	Eval	Train	Removed	RMSE1	RMSE2
1	4 6	U	VZ_me FX_me VZ_rms VX_rms FZ_sle	19.8326	19.6901
4	1 6	U	VY_me	23.5113	23.1872
6	1 4	U	VY_me	20.0487	34.8861

Table B.13: Experiment 1 SBS

Test	Eval	Train	Removed	RMSE1	RMSE2
1	4	U 6	VZ_rms	18.0304	17.2636
1	6	U 4	VX_rms VZ_me VX_sle	17.2790	17.7750
4	1	U 6 AA	VY_rms FX_rms	22.9934	23.0519
4	1	U 6 AB	VY_rms FZ_me	23.1521	23.1598
4	1	U 6 BA	VZ_rms FX_rms	23.6343	22.9987
4	1	U 6 BB	VZ_rms FZ_rms	23.8629	23.0786
4	6	U 1	VX_rms VZ_me VZ_rms VY_rms FX_sle	23.4939	24.1017
6	1	U 4 A	FX_rms	37.9241	37.9207
6	4	U 1 A	VZ_me VY_rms FX_rms FZ_me FX_me FZ_rms VX_me	30.7464	34.4576
6	4	U 1 B	VZ_me VY_me VX_me FZ_sle FZ_me FY_sle FX_rms	29.3038	40.4509

Table B.14: Experiment 2 SBS

Test	Eval	Train	Removed	RMSE1	RMSE2
1	4 6	U	VY FX	25.9527	24.8857
4	1 6	U	VY VZ VX FZ	23.7596	21.2156
6	1 4	U	FZ VY FX FY VX	23.6698	26.5374

Table B.15: Experiment 3 SBS

Test	Eval	Train	Removed	RMSE1	RMSE2
1	4	U 6	FX	27.1968	27.4125
1	6	U 4	VZ	17.2790	24.7388
4	1	U 6	VX	23.1817	21.6103
4	6	U 1	FZ	25.0843	26.4993
6	1	U 4	VZ	14.7172	32.2658
6	4	U 1		34.7496	34.7584

Table B.16: Experiment 4 SBS

Test	Eval	Train	Removed	RMSE1	RMSE2
1	4 6	U	VZ_me FX_me VY_me VX_me	0.5285	0.4880
4	1 6	U	VY_me VZ_me FZ_me FY_me	0.6547	0.7015
6	1 4	U	VY_me	0.5801	0.9085

Table B.17: Experiment 5 SBS

Test	Eval	Train	Removed	RMSE1	RMSE2
1	4	U 6	VZ_rms FX_rms FZ_me VX_me FZ_rms VZ_me	0.3984	0.4063
1	6	U 4	VX_rms VZ_me VX_sle	0.4063	0.4140
4	1	U 6 AA	VY_rms FX_rms	0.7015	0.7015
4	1	U 6 AB	VY_rms FZ_me	0.7127	0.7105
4	1	U 6 BA	VZ_rms FX_rms	0.7105	0.6969
4	1	U 6 BB	VZ_rms FZ_rms	0.7259	0.7037
4	6	U 1	VX_rms VZ_me VZ_rms VY_rms	0.7193	0.7216
6	1	U 4 A	FX_rms VY_sle	0.9809	0.8997
6	1	U 4 B	VX_sle VZ_sle FX_rms VX_me	1.0220	0.9743
6	4	U 1 AA	VZ_me VZ_sle FZ_rms FX_rms FZ_me VX_me FY_rms	0.7908	0.9808
6	4	U 1 AB	VZ_me VZ_sle FZ_rms FX_rms FZ_me VX_me FY_me	0.9677	1.0435
6	4	U 1 BA	VZ_me VZ_sle FZ_me FX_rms FZ_rms VX_me FY_rms	0.7908	0.9808
6	4	U 1 BB	VZ_me VZ_sle FZ_me FX_rms FZ_rms VX_me FY_me	0.9677	1.0435

Table B.18: Experiment 6 SBS

Test	Eval	Train	Removed	RMSE1	RMSE2
1	4 6	U	FZ VZ	0.6992	0.4577
4	1 6	U	VY VZ VX FZ	0.7517	0.6878
6	1 4	U	FZ VY FY	0.6901	0.8106

Table B.19: Experiment 7 SBS

Test	Eval	Train	Removed	RMSE1	RMSE2
1	4	U 6	FX FY	0.5285	0.6498
1	6	U 4	VZ	0.4063	0.5375
4	1	U 6	VX	0.7237	0.6992
4	6	U 1	FZ	0.7601	0.7705
6	1	U 4	VZ VY VX FZ	0.6947	0.6878
6	4	U 1	VZ	0.5463	0.8545

TABLE B.20: Experiment 8 SBS

Test	Eval	Train	Removed	Acc1	Acc2
1	4 6	U	FX_rms FX_sle	0.6603	0.7333
4	1 6	U	VY_me	0.5270	0.4921
6	1 4	U	VY_me	0.6635	0.6317

TABLE B.21: Experiment 9 SBS

Test	Eval	Train	Removed	Acc1	Acc2
1	4	U 6	VZ_rms FX_rms FZ_me VX_me FZ_rms VZ_me	0.8413	0.8349
1	6	U 4 AA	VX_rms FX_rms FX_me VX_sle VZ_rms VZ_me FZ_sle	0.2286	0.4032
1	6	U 4 AB	VX_rms FX_rms FX_me VZ_me VZ_rms	0.3746	0.4222
1	6	U 4 B	VX_rms FY_rms FX_sle	0.6762	0.7143
1	6	U 4 C	VX_rms FZ_rms FX_sle	0.6730	0.6317
1	6	U 4 DA	VX_rms FX_me FX_rms VX_sle VZ_rms VZ_me FZ_sle	0.2286	0.4032
1	6	U 4 DB	VX_rms FX_me FX_rms VZ_me VZ_rms	0.3746	0.4222
1	6	U 4 E	VX_rms FY_me FX_sle	0.6762	0.5111
1	6	U 4 F	VX_rms FZ_me FX_sle	0.6698	0.5143
4	1	U 6 AA	VY_rms FX_rms	0.5079	0.5079
4	1	U 6 AB	VY_rms FZ_me	0.4921	0.4952
4	1	U 6 BA	VZ_rms FX_rms	0.4952	0.5143
4	1	U 6 BB	VZ_rms FZ_rms	0.4730	0.5048
4	6	U 1	VX_rms VZ_me	0.5111	0.4762
6	1	U 4 A	FX_rms VY_sle	0.6095	0.6571
6	1	U 4 B	VX_sle VZ_sle FX_rms VX_me	0.5937	0.6222
6	4	U 1	VZ_me FY_rms FZ_rms FZ_me VX_me	0.3302	0.3302

TABLE B.22: Experiment 10 SBS

Test	Eval	Train	Removed	Acc1	Acc2
1	4 6	U	FX VZ	0.3492	0.3587
4	1 6	U	VY VZ VX FZ	0.4349	0.5270
6	1 4	U	FZ VY	0.4857	0.6254

Table B.23: Experiment 11 SBS

Test	Eval	Train	Removed	Acc1	Acc2
1	4	U 6	FX FY	0.7206	0.5778
1	6	U 4	VZ	0.8349	0.7111
4	1	U 6	VX	0.4762	0.5111
4	6	U 1	VZ FX	0.4635	0.4794
6	1	U 4	VZ	0.7905	0.6413
6	4	U 1	FM	0.6444	0.6571

Table B.24: Experiment 12 SBS

Appendix C

Probability Distributions

Gaussian Distribution $\mathcal{N}(x|\mu, \sigma^2)$

$$p(x|\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}. \tag{C.1}$$

Gamma Distribution Gamma$(x|\alpha, \beta)$

$$p(x|\alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}. \tag{C.2}$$

Inverse Gamma Distribution IG$(x|\alpha, \beta)$

$$p(x|\alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{-\alpha-1} e^{-\frac{\beta}{x}}. \tag{C.3}$$

Dirichlet Distribution Dir$(x_1, ..., x_{K-1}|\alpha_1, ..., \alpha_K)$

$$p(x_1, ..., x_{K-1}|\alpha_1, ..., \alpha_K) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{K} x_i^{\alpha_i - 1}, \tag{C.4}$$

where

$$B(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{K} \alpha_i)}, \tag{C.5}$$

and $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_K)$.

Exponential Distribution $\text{Exp}(x|\lambda)$

$$p(x|\lambda) = \lambda e^{-\lambda x}. \tag{C.6}$$

Poisson Distribution $\text{Pois}(x|\lambda)$

$$p(x|\lambda) = \frac{\lambda^x}{x!} e^{-\lambda}. \tag{C.7}$$

Bibliography

[1] Stephen Adams and Randy Cogill. Simultaneous feature selection and model training for hidden Markov models using MAP estimation. *Under Review*, 2015.

[2] Jalal Almhana, Zikuan Liu, Vartan Choulakian, and Robert McGorman. A recursive algorithm for gamma mixture models. In *Communications, 2006. ICC'06. IEEE International Conference on*, volume 1, pages 197–202. IEEE, 2006.

[3] Hussein Almuallim and Thomas G Dietterich. Learning with many irrelevant features. In *Proceedings of the ninth National conference on Artificial intelligence-Volume 2*, pages 547–552. AAAI Press, 1991.

[4] Nicos Angelopoulos and James Cussens. Exploiting informative priors for bayesian classification and regression trees. In *Proceedings of the 19th international joint conference on Artificial intelligence*, pages 641–646. Morgan Kaufmann Publishers Inc., 2005.

[5] Les Atlas, Mari Ostendorf, and Gary D Bernard. Hidden Markov models for monitoring machining tool-wear. In *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*, volume 6, pages 3887–3890. IEEE, 2000.

[6] Faisal I Bashir, Ashfaq A Khokhar, and Dan Schonfeld. Object trajectory-based activity classification and recognition using hidden Markov models. *Image Processing, IEEE Transactions on*, 16(7):1912–1919, 2007.

[7] Ardhendu Behera, Anthony Cohn, and David Hogg. Workflow activity monitoring using dynamics of pair-wise qualitative spatial relations. *Advances in Multimedia Modeling*, pages 196–209, 2012.

[8] Jeff Bilmes. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. Technical Report TR-97-021, International Computer Science Institute, 1998.

[9] Avrim L Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial intelligence*, 97(1):245–271, 1997.

[10] Enrico Bocchieri. Vector quantization for the efficient computation of continuous density likelihoods. In *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on*, volume 2, pages 692–695. IEEE, 1993.

[11] Christos Boutsidis, Petros Drineas, and Michael W Mahoney. Unsupervised feature selection for the k-means clustering problem. In *Advances in Neural Information Processing Systems*, pages 153–161, 2009.

[12] Richard Boys and Daniel Henderson. A comparison of reversible jump MCMC algorithms for DNA sequence segmentation using hidden Markov models. *Computing Science and Statistics*, 33:35–49, 2001.

[13] Matthew Brand, Nuria Oliver, and Alex Pentland. Coupled hidden Markov models for complex action recognition. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 994–999. IEEE, 1997.

[14] Peter Carbonetto, Nando De Freitas, Paul Gustafson, and Natalie Thompson. Bayesian feature weighting for unsupervised learning, with application to object recognition. In *Artificial Intelligence and Statistics (AI & Statistics' 03)*. Society for Artificial Intelligence and Statistics, 2003.

[15] Rich Caruana and Dayne Freitag. Greedy attribute selection. In *In Proceedings of the Eleventh International Conference on Machine Learning*, pages 28–36, 1994.

[16] Rich Caruana and Dayne Freitag. How useful is relevance? *FOCUS*, 14(8):2, 1994.

[17] Ozgür Cetin and Mari Ostendorf. Multi-rate hidden Markov models and their application to machining tool-wear classification. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, volume 5, pages V–837. IEEE, 2004.

[18] Ozgür Cetin, Mari Ostendorf, and Gary D Bernard. Multirate coupled hidden Markov models and their application to machining tool-wear classification. *Signal Processing, IEEE Transactions on*, 55(6):2885–2896, 2007.

[19] Shaorong Chang, Nilanjan Dasgupta, and Lawrence Carin. A Bayesian approach to unsupervised feature selection and density estimation using expectation propagation. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 1043–1050. IEEE, 2005.

[20] Jose M Chaquet, Enrique J Carmona, and Antonio Fernández-Caballero. A survey of video datasets for human action and activity recognition. *Computer Vision and Image Understanding*, 117(6):633–659, 2013.

[21] Sotirios Chatzis and Dimitrios Kosmopoulos. A variational Bayesian methodology for hidden Markov models utilizing Student's-t mixtures. *Pattern Recognition*, 44(2):295–306, 2011.

[22] Matthew C Coleman and David E Block. Bayesian parameter estimation with informative priors for nonlinear systems. *AIChE journal*, 52(2):651–667, 2006.

[23] Guido Consonni and Jean-Micheal Marin. Mean-field variational approximate Bayesian inference for latent variable models. *Computational Statistics and Data Analysis*, 52(2):790–798, 2007.

[24] Constantinos Constantinopoulos, Michalis K Titsias, and Aristidis Likas. Bayesian feature and model selection for Gaussian mixture models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(6):1013–1018, 2006.

[25] Thomas M Cover and Jan M Van Campenhout. On the possible orderings in the measurement selection problem. *Systems, Man and Cybernetics, IEEE Transactions on*, 7(9):657–661, 1977.

[26] Walter Daelemans, Véronique Hoste, Fien De Meulder, and Bart Naudts. Combined optimization of feature selection and algorithm parameters in machine learning of language. In *Machine Learning: Ecml 2003*, pages 84–95. Springer, 2003.

[27] Manoranjan Dash, Hua Liu, and Jun Yao. Dimensionality reduction of unsupervised data. In *Tools with Artificial Intelligence, 1997. Proceedings., Ninth IEEE International Conference on*, pages 532–539. IEEE, 1997.

[28] Manoranjan Dash, Huan Liu, and Hiroshi Motoda. Consistency based feature selection. In *Knowledge Discovery and Data Mining. Current Issues and New Applications*, pages 98–109. Springer, 2000.

[29] Dimla E Dimla. Sensor signals for tool-wear monitoring in metal cutting operations—a review of methods. *International Journal of Machine Tools and Manufacture*, 40(8):1073–1098, 2000.

[30] Thi V Duong, Hung Hai Bui, Dinh Q Phung, and Svetha Venkatesh. Activity recognition and abnormality detection with the switching hidden semi-Markov model. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 838–845. IEEE, 2005.

[31] Jennifer G Dy and Carla E Brodley. Feature subset selection and order identification for unsupervised learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 247–254. Morgan Kaufmann Publishers Inc., 2000.

[32] Jennifer G Dy and Carla E Brodley. Feature selection for unsupervised learning. *The Journal of Machine Learning Research*, 5:845–889, 2004.

[33] Jennifer G. Dy, Carla E. Brodley, Avi Kak, Lynn S. Broderick, and Alex M. Aisen. Unsupervised feature selection applied to content-based retrieval of lung images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(3):373–378, 2003.

[34] Huseyin M Ertunc, Kenneth A Loparo, and Hasan Ocak. Tool wear condition monitoring in drilling operations using hidden Markov models (HMMs). *International Journal of Machine Tools and Manufacture*, 41(9):1363–1384, 2001.

[35] Huseyin Metin Ertunc and Cuneyt Oysu. Drill wear monitoring using cutting force signals. *Mechatronics*, 14(5):533–548, 2004.

[36] Mário AT Figueiredo, Anil K Jain, and Martin H Law. A feature selection wrapper for mixtures. In *Pattern Recognition and Image Analysis*, pages 229–237. Springer, 2003.

[37] Randall K. Fish, Mari Ostendorf, Gary D. Bernard, and David A. Castanon. Multilevel classification of milling tool wear with confidence estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(1):75–85, 2003.

[38] George Forman. An extensive empirical study of feature selection metrics for text classification. *The Journal of Machine Learning Research*, 3:1289–1305, 2003.

[39] Aphrodite Galata, Anthony Cohn, Derek Magee, and David Hogg. Modeling interaction using learnt qualitative spatio-temporal relations and variable length Markov models. In *In Proceedings of the European Conference on Artificial Intelligence*, pages 741–745, 2002.

[40] Like Gao and X Sean Wang. Feature selection for building cost-effective data stream classifiers. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 621–624. IEEE Computer Society, 2005.

[41] Jean-Luc Gauvian and Chin-Hui Lee. Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov models. *IEEE Transactions on Speech and Audio Processing*, 2(2), April 1994.

[42] Donghai Guan, Weiwei Yuan, Young-Koo Lee, Andrey Gavrilov, and Sungyoung Lee. Activity recognition based on semi-supervised learning. In *Embedded and Real-Time Computing Systems and Applications, 2007. RTCSA 2007. 13th IEEE International Conference on*, pages 469–475. IEEE, 2007.

[43] Seth D Guikema. Formulating informative, data-based priors for failure probability estimation in reliability analysis. *Reliability Engineering & System Safety*, 92(4):490–502, 2007.

[44] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[45] Jungong Han, Ling Shao, Dong Xu, and Jamie Shotton. Enhanced computer vision with microsoft Kinect sensor: A review. *Cybernetics, IEEE Transactions on*, 43(5):1318–1334, 2013.

[46] Alexander Ihler, Jon Hutchins, and Padhraic Smyth. Adaptive event detection with time-varying Poisson processes. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 207–216. ACM, 2006.

[47] K Iswandy and A Koenig. Feature selection with acquisition cost for optimizing sensor system design. *Advances in Radio Science*, 4(7):135–141, 2006.

[48] Ahmad Jalal, Sungyoung Lee, Jeong Tai Kim, and Tae-Seong Kim. Human activity recognition via the features of labeled depth body parts. In *Impact Analysis of Solutions for Chronic Disease Prevention and Management*, pages 246–249. Springer, 2012.

[49] Hakjin Jang, Soobeom Lee, and Seong W Kim. Bayesian analysis for zero-inflated regression models with the power prior: Applications to road safety countermeasures. *Accident Analysis & Prevention*, 42(2):540–547, 2010.

[50] E.T. Jaynes. Highly informative priors. *Bayesian Statistics*, 2:329–360, 1985.

[51] Shihao Ji and Lawrence Carin. Cost-sensitive feature acquisition and classification. *Pattern Recognition*, 40(5):1474–1485, 2007.

[52] George H John, Ron Kohavi, Karl Pfleger, et al. Irrelevant features and the subset selection problem. In *Machine Learning: Proceedings of the Eleventh International Conference*, pages 121–129, 1994.

[53] Jing Kang, Ni Kang, Chang-jian Feng, and Hong-ying Hu. Research on tool failure prediction and wear monitoring based HMM pattern recognition theory. In *Wavelet Analysis and Pattern Recognition, 2007. ICWAPR'07. International Conference on*, volume 3, pages 1167–1172. IEEE, 2007.

[54] YeongSeog Kim, W Nick Street, and Filippo Menczer. Feature selection in unsupervised learning via evolutionary search. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 365–369. ACM, 2000.

[55] Kenji Kira and Larry A Rendell. The feature selection problem: Traditional methods and a new algorithm. In *AAAI*, volume 2, pages 129–134, 1992.

[56] Ron Kohavi and George H John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1):273–324, 1997.

[57] Igor Kononenko. Estimating attributes: analysis and extensions of RELIEF. In *Machine Learning: ECML-94*, pages 171–182. Springer, 1994.

[58] Andreas Krause, Daniel P Siewiorek, Asim Smailagic, and Jonny Farringdon. Unsupervised, dynamic identification of physiological and activity context in wearable computing. In *2012 16th International Symposium on Wearable Computers*, pages 88–88. IEEE Computer Society, 2003.

[59] Junghyun Kwon and Frank C Park. Natural movement generation using hidden Markov models and principal components. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 38(5):1184–1194, 2008.

[60] Oscar D Lara and Miguel A Labrador. A survey on human activity recognition using wearable sensors. *Communications Surveys & Tutorials, IEEE*, 15(3):1192–1209, 2013.

[61] Martin H Law, Anil K Jain, and Mário Figueiredo. Feature selection in mixture-based clustering. In *Advances in Neural Information Processing Systems*, pages 625–632, 2002.

[62] Martin HC Law, Mario AT Figueiredo, and Anil K Jain. Simultaneous feature selection and clustering using mixture models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1154–1166, 2004.

[63] Xiao Li and Jeff Bilmes. Feature pruning in likelihood evaluation of HMM-based speech recognition. In *Automatic Speech Recognition and Understanding, 2003. ASRU'03. 2003 IEEE Workshop on*, pages 303–308. IEEE, 2003.

[64] Xiao Li and Jeff Bilmes. Feature pruning for low-power ASR systems in clean and noisy environments. *Signal Processing Letters, IEEE*, 12(7):489–492, 2005.

[65] Yuanhong Li, Ming Dong, and Jing Hua. Simultaneous localized feature selection and model detection for Gaussian mixtures. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(5):953–960, 2009.

[66] M. Lichman. UCI machine learning repository, 2013.

[67] Charles X Ling, Qiang Yang, Jianning Wang, and Shichao Zhang. Decision trees with minimal costs. In *Proceedings of the Twenty-First International Conference on Machine Learning*, page 69. ACM, 2004.

[68] Xiaoming Liu and Tsuhan Chen. Video-based face recognition using adaptive hidden Markov models. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–340. IEEE, 2003.

[69] Dominique Lord and Luis F Miranda-Moreno. Effects of low sample mean values and small sample size on the estimation of the fixed dispersion parameter of Poisson-gamma models for modeling motor vehicle crashes: a Bayesian perspective. *Safety Science*, 46(5):751–770, 2008.

[70] Fengjun Lv and Ramakant Nevatia. Recognition and segmentation of 3-d human action using HMM and multi-class adaboost. In *Computer Vision–ECCV 2006*, pages 359–372. Springer, 2006.

[71] Maryam Mahdaviani and Tanzeem Choudhury. Fast and scalable training of semi-supervised CRFs with application to activity recognition. In *Advances in Neural Information Processing Systems*, pages 977–984, 2008.

[72] Cathy Maugis, Gilles Celeux, and Marie-Laure Martin-Magniette. Variable selection for clustering with Gaussian mixture models. *Biometrics*, 65(3):701–709, 2009.

[73] C.A. McGrory and D.M. Titterington. Variational Bayesian analysis for hidden Markov models. *Australian and New Zealand Journal of Statistics*, 51(2):227–244, 2009.

[74] Fan Min, Huaping He, Yuhua Qian, and William Zhu. Test-cost-sensitive attribute reduction. *Information Sciences*, 181(22):4928–4942, 2011.

[75] Fan Min, Qinghua Hu, and William Zhu. Feature selection with test cost constraint. *International Journal of Approximate Reasoning*, 55(1):167–179, 2014.

[76] Fan Min and Qihe Liu. A hierarchical model for test-cost-sensitive decision systems. *Information Sciences*, 179(14):2442–2452, 2009.

[77] Pabitra Mitra, CA Murthy, and Sankar K. Pal. Unsupervised feature selection using feature similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):301–312, 2002.

[78] Jose Antonio Montero and Luis Enrique Sucar. Feature selection for visual gesture recognition using hidden Markov models. In *Proc. 5th Int. Conf. Computer Science, 2004. ENC 2004.*, pages 196–203. IEEE, 2004.

[79] Sach Mukherjee and Terence P Speed. Network inference using informative priors. *Proceedings of the National Academy of Sciences*, 105(38):14313–14318, 2008.

[80] Kevin P Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[81] Patrenahalli M Narendra and Keinosuke Fukunaga. A branch and bound algorithm for feature subset selection. *Computers, IEEE Transactions on*, 100(9):917–922, 1977.

[82] Nam Thanh Nguyen, Dinh Q Phung, Svetha Venkatesh, and Hung Bui. Learning and detecting activities from movement trajectories using the hierarchical hidden Markov model. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 955–960. IEEE, 2005.

[83] Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision*, 79(3):299–318, 2008.

[84] Jan Nouza. Feature selection methods for hidden Markov model-based speech recognition. *Proc. 13th Int. Conf. Pattern Recognition*, 2:186–190, 1996.

[85] Sebastian Nowozin and Jamie Shotton. Action points: A representation for low-latency online human action recognition. *Microsoft Research Cambridge, Tech. Rep. MSR-TR-2012-68*, 2012.

[86] Marlon Núñez. The use of background knowledge in decision tree induction. *Machine Learning*, 6(3):231–250, 1991.

[87] Nuria Oliver, Ashutosh Garg, and Eric Horvitz. Layered representations for learning and inferring office activity from multiple sensory channels. *Computer Vision and Image Understanding*, 96(2):163–180, 2004.

[88] Omar Oreifej and Zicheng Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 716–723. IEEE, 2013.

[89] Lane MD Owsley, Les E Atlas, and Gary D Bernard. Self-organizing feature maps and hidden Markov models for machine-tool monitoring. *Signal Processing, IEEE Transactions on*, 45(11):2787–2798, 1997.

[90] Pavel Paclík, Robert PW Duin, Geert MP van Kempen, and Reinhard Kohlus. On feature selection with measurement cost and grouped features. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 461–469. Springer, 2002.

[91] Wei Pan and Xiaotong Shen. Penalized model-based clustering with application to variable selection. *The Journal of Machine Learning Research*, 8:1145–1164, 2007.

[92] Ronald Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976–990, 2010.

[93] Pavel Pudil, FJ Ferri, J Novovicova, and J Kittler. Floating search methods for feature selection with nonmonotonic criterion functions. In *In Proceedings of the Twelveth International Conference on Pattern Recognition, IAPR*. Citeseer, 1994.

[94] Pavel Pudil, Jana Novovičová, and Josef Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15(11):1119–1125, 1994.

[95] Lawrence Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–286, February 1989.

[96] Lawrence R Rabiner and Biing-Hwang Juang. *Fundamentals of Speech Recognition*, volume 14. PTR Prentice Hall Englewood Cliffs, 1993.

[97] Adrian E Raftery and Nema Dean. Variable selection for model-based clustering. *Journal of the American Statistical Association*, 101(473):168–178, 2006.

[98] Rajat Raina, Andrew Y Ng, and Daphne Koller. Constructing informative priors using transfer learning. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 713–720. ACM, 2006.

[99] Adam G Rehorn, Jin Jiang, and Peter E Orban. State-of-the-art methods and results in tool condition monitoring: a review. *The International Journal of Advanced Manufacturing Technology*, 26(7-8):693–710, 2005.

[100] Marko Robnik-Šikonja and Igor Kononenko. Theoretical and empirical analysis of ReliefF and RReliefF. *Machine Learning*, 53(1-2):23–69, 2003.

[101] D.J. Rude, S. Adams, and Peter A. Beling. A Benchmark Dataset for Depth Sensor-Based Activity Recognition in a Manufacturing Process. pages 702–708, Ottawa, Canada, May 2015. IFAC.

[102] C Scheffer, H Engelbrecht, and PS Heyns. A comparative evaluation of neural networks and hidden Markov models for monitoring turning tool wear. *Neural Computing & Applications*, 14(4):325–336, 2005.

[103] Victor S Sheng, Charles X Ling, Ailing Ni, and Shichao Zhang. Cost-sensitive test strategies. In *Proceedings of the National Conderence on Artificial Intelligence*, volume 21, page 482. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.

[104] PC Smits and A Annoni. Cost-based feature selection for GIS-embedded data fusion. In *Geoscience and Remote Sensing Symposium, 2000. Proceedings. IGARSS 2000. IEEE 2000 International*, volume 6, pages 2614–2616. IEEE, 2000.

[105] Petr Somol, Pavel Pudil, and Josef Kittler. Fast branch & bound algorithms for optimal feature selection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(7):900–912, 2004.

[106] Maja Stikic, Kristof Van Laerhoven, and Bernt Schiele. Exploring semi-supervised and active learning for activity recognition. In *Wearable Computers, 2008. ISWC 2008. 12th IEEE International Symposium on*, pages 81–88. IEEE, 2008.

[107] Jaeyong Sung, Colin Ponce, Bart Selman, and Ashutosh Saxena. Human activity detection from RGBD images. *Plan, Activity, and Intent Recognition*, 64, 2011.

[108] Jaeyong Sung, Colin Ponce, Bart Selman, and Ashutosh Saxena. Unstructured human activity detection from rgbd images. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 842–849. IEEE, 2012.

[109] Duncan C Thomas, John S Witte, and Sander Greenland. Dissecting effects of complex mixtures: who's afraid of informative priors? *Epidemiology*, 18(2):186–190, 2007.

[110] Pavan Turaga, Rama Chellappa, Venkatramana S Subrahmanian, and Octavian Udrea. Machine recognition of human activities: A survey. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(11):1473–1488, 2008.

[111] Fabio Valente and Christian Wellekens. Variational Bayesian feature selection for Gaussian mixture models. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, volume 1, pages I–513. IEEE, 2004.

[112] Antonio G Vallejo Jr, Juan A Nolazco-Flores, Rubén Morales-Menéndez, L Enrique Sucar, and Ciro A Rodríguez. Tool-wear monitoring based on continuous hidden Markov models. In *Progress in Pattern Recognition, Image Analysis and Applications*, pages 880–890. Springer, 2005.

[113] Marina Vannucci and Francesco C Stingo. Bayesian models for variable selection that incorporate biological information. *Bayesian Statistics*, 9, 2010.

[114] Wolf Vanpaemel. Constructing informative model priors using hierarchical methods. *Journal of Mathematical Psychology*, 55(1):106–117, 2011.

[115] S Varma and John S Baras. Tool wear estimation from acoustic emissions: a model incorporating wear-rate. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 1, pages 492–495. IEEE, 2002.

[116] Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1290–1297. IEEE, 2012.

[117] Litao Wang, Mostafa G Mehrabi, and Elijah Kannatey-Asibu. Hidden Markov model-based tool wear monitoring in turning. *Journal of Manufacturing Science and Engineering*, 124(3):651–658, 2002.

[118] Sijian Wang and Ji Zhu. Variable selection for model-based high-dimensional clustering and its application to microarray data. *Biometrics*, 64(2):440–448, 2008.

[119] Xiaogang Wang, Xiaoxu Ma, and W Eric L Grimson. Unsupervised activity perception in crowded and complicated scenes using hierarchical Bayesian models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(3):539–555, 2009.

[120] Ying Wang, Kaiqi Huang, and Tieniu Tan. Human activity recognition based on R transform. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[121] Simon Washington and Jutaek Oh. Bayesian methodology incorporating expert judgment for ranking countermeasure effectiveness under uncertainty: Example applied to at grade railroad crossings in Korea. *Accident Analysis & Prevention*, 38(2):234–247, 2006.

[122] David Windridge and Richard Bowden. Hidden Markov chain estimation and parameterisation via ICA-based feature-selection. *Pattern Analysis and Applications*, 8(1-2):115–124, 2005.

[123] Robert L Winkler, James E Smith, and Dennis G Fryback. The role of informative priors in zero-numerator problems: being conservative versus being candid. *The American Statistician*, 56(1):1–4, 2002.

[124] Danny Wyatt, Matthai Philipose, and Tanzeem Choudhury. Unsupervised activity recognition using automatically mined common sense. In *AAAI*, volume 5, pages 21–27, 2005.

[125] Lu Xia, Chia-Chih Chen, and JK Aggarwal. View invariant human action recognition using histograms of 3d joints. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 20–27. IEEE, 2012.

[126] Lexing Xie, Shih-Fu Chang, Ajay Divakaran, and Huifang Sun. Structure analysis of soccer video with hidden Markov models. *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, 4, 2002.

[127] Qiang Yang, Charles Ling, Xiaoyong Chai, and Rong Pan. Test-cost sensitive classification on data with missing values. *Knowledge and Data Engineering, IEEE Transactions on*, 18(5):626–638, 2006.

[128] Xiaodong Yang and YingLi Tian. Eigenjoints-based action recognition using naive-bayes-nearest-neighbor. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 14–19. IEEE, 2012.

[129] Yiyu Yao, Yan Zhao, Jue Wang, and Suqing Han. A model of machine learning based on user preference of attributes. In *Rough Sets and Current Trends in Computing*, pages 587–596. Springer, 2006.

[130] Pei Yin, Irfan Essa, Thad Starner, and James M Rehg. Discriminative feature selection for hidden Markov models using segmental boosting. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 2001–2004. IEEE, 2008.

[131] Lei Yu and Huan Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, volume 3, pages 856–863, 2003.

[132] Rongjie Yu and Mohamed Abdel-Aty. Investigating different approaches to develop informative priors in hierarchical Bayesian safety performance functions. *Accident Analysis & Prevention*, 56:51–58, 2013.

[133] Shun-Zheng Yu. Hidden semi-Markov models. *Artificial Intelligence*, 174(2):215–243, 2010.

[134] Hao Zhang and Lynne E Parker. 4-dimensional local spatio-temporal features for human activity recognition. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 2044–2049. IEEE, 2011.

[135] Hong Zhao, Fan Min, and William Zhu. Cost-sensitive feature selection of numeric data with measurement errors. *Journal of Applied Mathematics*, 2013, 2013.

[136] Jian Zhou and Xiao-Ping Zhang. An ICA mixture hidden Markov model for video content analysis. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(11):1576–1586, 2008.

[137] Hao Zhu, Zhongshi He, and Henry Leung. Simultaneous feature and model selection for continuous hidden Markov models. *IEEE Signal Processing Letters*, 19(5), May 2012.

[138] Kunpeng Zhu, Yoke San Wong, and Geok Soon Hong. Multi-category micro-milling tool wear monitoring with continuous hidden Markov models. *Mechanical Systems and Signal Processing*, 23(2):547–560, 2009.