MACHINE LEARNING MODELS FOR DESIGNING SMART CITIES AND COMMUNITIES

BY

GISSELLA MARIA BEJARANO NICHO

B*S*, Pontificia Universidad Católica del Perú, 2009
M*S*, Binghamton University, 2017

Submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy in Computer Science
in the Graduate School of
Binghamton University
State University of New York
2021

Accepted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy in Computer Science
in the Graduate School of
Binghamton University
State University of New York
2021

June 01, 2021

Dr. Arti Ramesh, Chair and Faculty Advisor
Department of Computer Science, Binghamton University

Dr. Anand Seetharam, Member
Department of Computer Science, Binghamton University

Dr. Zhongfei (Mark) Zhang, Member
Department of Computer Science, Binghamton University

Dr. Lei Yu, Member
Department of Computer Science, Binghamton University

Dr. Xingye Qiao, Outside Examiner
Department of Mathematical Sciences, Binghamton University

Abstract

Discovering important patterns in data can help cities to plan, monitor, and assign resources more efficiently, converting them in smart cities with more organized communities. Machine learning models can take advantage of this large amount of data to improve and scale these cities' duties. In this work, we explore machine learning approaches to solve different problems in the smart cities domain related to water consumption, energy consumption and emergency events. More specifically, our work sheds light on the design of ensemble learning, sequential models and the combination of probabilistic graphical and deep learning models to this type of problems. Moreover, we carefully compare, adapt and implement methods to address the particular characteristics of the data and the problems of smart cities.

We focus on four specific problems: i) classifying the water pump operation status, quality and quantity, ii) predicting the future water consumption based on historical consumption, iii) time resolution prediction for emergency events and iv) disaggregating energy signals into their component appliances. To address these problems, we develop, compare and combine three families of machine learning models: ensemble learning, probabilistic graphical and deep learning. For the classification problem, we develop ensemble-learning models which outperform a Support Vector Machine approach. We identify the most relevant individual and group

features from two African countries datasets, where the management of this resource is fundamental. For the water consumption prediction problem, we develop a probabilistic graphical model such as an sparse Gaussian Conditional Random Field model and a deep learning model such as an LSTM-based encoder-decoder, to perform structured prediction. Besides, we introduce additional features at each time step and find that in most of the cases they improve the multi-step prediction performance. We continue exploring the suitability of the LSTM-based encoder decoder to the time resolution prediction for emergency events in New York City, a challenging dataset with no discerning temporal patterns. Finally, for the energy disaggregation problem, we build a deep latent generative model that disaggregates the total energy consumption of a house by jointly generating and predicting the continuous values of the disaggregated signals. We evaluate the performance of all our models on real-world data and show their effectiveness through extensive experimentation. Our models and analysis have the potential to positively impact the development of smart cities and communities.

To Román, my family and friends

Acknowledgements

From moral support to financial help, I want to thank several people and institutions for helping me in this research journey. I thank my advisor, Arti Ramesh for accepting me in her lab, guiding and encouraging in every step that I have taken during my studies. I also value a lot the advising and questions of professors Anand Seetharam, Lei Yu, Mark Zhang and Xingye Qiao, members of my thesis comittee.

I will always remain grateful to Fulbright, Binghamton University and Professor Lander, for giving me the opportunity to start my graduate studies as a master student and believing in my capacity to continue my doctoral studies. Besides, to Pontificia Universidad Católica del Perú and Professor César Beltrán, with who, and a bunch of other folks, I found one of the most important research groups in Artificial Intelligence in Peru (IAPUCP).

I feel blessed to have count on my family in Peru and the U.S. and all my friends, specially during the ups and downs of this journey. I want to thank to my siblings for taking care of my parents. I want to thank my friends who always had words of encouragement, a meal to share or a simple hug to know we had each other, specially to Mavi, Rushui and Angel. To not burn out, some breaks were the perfect time to engage in insightful conversations with my labmates and friends from CS and other departments. They have helped me to listen to other perspectives and learn to question and discuss with respect from topics such politics, future work and search space techniques. Finally, and most importantly, I want to thank my optimal love, Román, for supporting me everytime I thought I was not able to continue, for his love and for all the adventures that we had during this long-distance relationship, we made it!

Preface

This thesis is a compilation of the results obtained in linked projects in which the author work in her doctoral studies at Binghamton University. These results were published by the author and her collaborators which acknoledge the lead role of the author. These works were published in the following recognized venues for artificial intelligence, applied systems and social computing:

- Predictive Analytics for Smart Water Management in Developing Regions - The 4th IEEE International Conference on Smart Computing (SMARTCOMP, 2018) [1]
- SWaP: Probabilistic Graphical and Deep Learning Models for Water Consumption Prediction - Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys, 2019) [2]
- DeepER: A Deep Learning based Emergency Resolution Time Prediction System - IEEE International Conferences on Cyber, Physical and Social Computing (CPSCom, 2020) [3]
- Deep Latent Generative Models for Energy Disaggregation - Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI, 2019)[4]

Contents

List of Tables

List of Figures

List of Abbreviations

ARIMA	Auto-Regressive Integrated Moving Average
CNN	Convolutional Neural Networks
CRF	Conditional Random Fields
DL	Deep Learning
DPGM	Deep Probabilistic Graphical Models
PG	Probabilistic Graphical
HMM	Hidden Markov Models
LSTM	Long Short Term Memory
MSE	Mean Squared Error
NILM	Non-instrusive Load Monitoring
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
SEQ2SEQ	Sequente-to-sequence
SVM	Super Vector Machine
VAE	Variational Autoencoder
VRNN	Variational Recurrent Neural Netork

1 Introduction

Cities around the world are producing large amounts of data every day. Machine Learning models are suitable to tranform this data in useful outputs that can leverage decision-making for public services. Moreover, they can contribute to the understanding of population behaviour to better shape public policies. By understanding the patterns of consumption and mobility, public servers can design better solutions and optmize resources in their cities and countries. Problems addressed with machine learning in this thesis belong to a variety of domains such as demand and management of water or energy and forecasting of emergency events.

1.1 Contributions

In this section, we present the problems and the main contributions of this thesis. We start by addressing the water availability and management problem, specially in least developed countries. Many factors including geographic, political, management, and environmental ones affect the availability of water in these regions. For this reason, we develop an ensemble-learning based predictive-analytics framework for smart water management to predict water pump operation status (e.g., functional, non functional), water quality and quantity. First, we perform feature engineering to select relevant features, use them to develop the XGBoost and Random Forest ensemble learning models, and then perform extensive feature analysis to identify the most predictive features, for each prediction problem mentioned above. We evaluate our framework on two publicly available smart water management datasets pertaining to Tanzania and Nigeria and show that our proposed models outperform a Supper Vector Machine approach in most of metrics such as precision, recall and F1 score. We also demonstrate that our models are able to achieve a superior prediction performance for predicting water pump operation status for different water extraction methods. We conduct a detailed feature analysis to investigate the importance of the various feature groups (e.g., geographic, management) on the performance of the models for predicting water pump operation status, water quality and quantity. We then perform a fine-grained feature analysis to identify how individual features, not just feature groups, impact performance. We identify that among individual features, location

(x, y, z coordinates) has the maximum impact on performance. Our analysis is helpful in understanding the types of data that should be collected in future for accurately predicting the different water problems.

Complementary to water related problems in rural areas, consumption prediction in residential and commercial can help identify possible leaks, minimize water wastage, and pave the way developing initiatives for a sustainable future. For this reason, we implement **SWaP**, a **S**mart **Wa**ter **P**redicion system that forecasts hourly water consumption based on historical data. To perform this prediction task, we design discriminative probabilistic graphical (PG) and deep learning models (DL), in particular, sparse Gaussian Conditional Random Fields (GCRFs) and Long Short Term Memory (LSTM) based deep Recurrent Neural Network (RNN) models, to successfully encode dependencies in the water consumption data. We evaluate our system on water consumption data collected from multiple buildings in a university campus and demonstrate that both the GCRF and LSTM based deep models are able to accurately predict future hourly water consumption in advance using just the last 24 hours of data at test time. **SWaP** achieves superior prediction performance for all buildings in comparison to the linear regression and ARIMA baselines in terms of Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), with the GCRF and LSTM models providing 50% and 44% improvements on average, respectively. We also demonstrate that augmenting our models with temporal features such as time of the day and day of the week can improve the overall average prediction performance. Additionally, based on our evaluation, we observe that the GCRF model outperforms the LSTM based deep learning model, while simultaneously being faster to train and execute at test time. The computationally efficient and interpretable nature of GCRF models in **SWaP** make them an ideal choice for practical deployment.

Next, we adapt a similar LSTM-based encoder-decoder used for water consumption prediction to resolution time prediction problem for emergency events in New York City, USA. Due to the lack of obvious temporal patterns in the time series, this dataset provides a unique opportunity. Accurately predicting resolution time for emergency incidents is crucial for public safety and smooth functioning of cities as it helps in planning resources that will be available for immediate assistance. For this reason, we implement DeepER, a deep learning based emergency resolution time prediction system that predicts future resolution times based on past data. We effectively preprocess the data to deal with uneven distribution of resolution times, outliers, and missing values. We compare the performance of the model with ARIMA and Linear Regression using two metrics— Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). DeepER achieves an average performance improvement of 3% and 16% with respect to RMSE and 10%

and 27% with respect to MAE over ARIMA and Linear Regression, respectively.

Analagous to water management, designing machine learning models for smart energy consumption is an important research problem, having a tremendous impact on sustainable societies. A crucial sub-problem in facilitating smart energy consumption is to accurately disaggregate energy signals into their component appliance signals. This process is also known as energy disaggregation/non-intrusive load monitoring (NILM). This exercise provides residents with an accurate view and understanding of their energy consumption and can potentially help in reducing the peak energy consumption and facilitating efficient usage and conservation of energy. Recent advances in variational inference for DL have resulted in more expressive deep generative models such as variational autoencoders and variational recurrent neural networks that possess the ability to encode continuous latent variables. These latent variables provide the models with a powerful layer of abstraction that captures the variations in the input data and helps in generating the output data. These models map the input sequence into continuous latent variables using an inference network (referred to as an encoder), and then use the generative network (referred to as a decoder) to reconstruct the input sequence by sampling from the latent variables. Chung et al. [5] propose variational recurrent neural networks (VRNNs), which extend VAEs to model sequences by introducing high-level latent variables in RNNs. Deep generative models such as VRNNs and VAEs have implemented for many sequence-to-sequence language tasks such as machine translation, paraphrase generation, and textual entailment, but have not been explored for the problem of energy disaggregation. Following the contributions of this thesis, we present a novel deep generative architecture for disaggregation that leverages and adapts VRNNs to jointly disaggregate the total energy consumption into individual component appliance signals. Our proposed approach learns the abstraction of the aggregated energy consumption over latent variables at training time and then generates all the individual appliance signals jointly by sampling from the latent variables at test time. Hence, at test time our model only depends on the aggregated signal and the latent variable abstractions learned during training and does not depend on contextual information and appliance data from previous time steps, making it a meaningful model for energy disaggregation.

1.2 Future Challenges

Machine learning models see and train over historic data or training set to predict results in new data. The error and precision of a model are reported based on the results over the testing set. Usually, the data collected for problems in

smart cities belong to at least couple of months to be able to recognize interesting patterns. In our models, data is collected with a granularity of at least minutes which allows us to focus in the planning and decision making process. However, for the nature of some domains in smart cities, models might need to process and infer some results at the granularity of seconds or microseconds in real time. To attend such approaches is what could really make a difference in productionizing and scaling machine learning models for smart cities. We acknoledge that to fulfill such goal, machine learning models would have to be combined with the more proper networking and behaviour knowledge. That is what we could call the real future challenge of the application of machine learning models for smart cities from a computer science perspective.

1.3 Thesis Organization

This thesis is structured as follows. Chapter 2 aims to explain the neccesary background and general concepts to understand more specific models in the rest of the work. Chapter 3 presents the work for water management pump operation status, quality and quantity classificaiton. Chapter 4 continues with **SWaP**, a predictive framework for water consumption. Chapter 5 explains the details of our work on time resolution prediction for emergency events. Chapter 6 shows the details on our variant of the Variational Recurrent Neural Network for the energy disaggregation problem. Finally, chapter 7 summarizes the conclusions and the future research paths provided by this thesis work.

2 Background

In this chapter, we aim to give more theoretical background to the machine learning models used to address our smart city problems. The first section explains more details of two of the more general types of ensemble models implemented in 3.4.3, for water pump classification: boosting and boostrap. The second section provides details on sequential models from the probabilistic graphical approach as in Conditional Random Fields (CRF), and from the deep learning approach with the seq2seq and encoder-decoder architecture. In 4.5, we compare sparse Gaussian Conditional Random Fields (GCRF) and a seq2seq model to address the water consumption prediction problem. This type of model is a variant of CRF for continuos values. In 5.7, we focus on a seq2seq approach as based of our framework to adress the Emergency Resolution Time prediction. Finally, in the third section of this chapter, we aim to introduce more background on Deep Probabilistic Graphical Models (DPGM) and Variational Inference. These two concepts are key to understand the adaptaion of the Variational Recurrent Neural Networks (VRNN) implemented in 6.4.

2.1 Ensemble Models

Ensemble models leverage the accuracy of single supervised learning methods by considering the results of several classifiers. These classifiers or learners can be base on the same approach or they could be different approaches. Among these approachs, we can find ensemble types such as Bayes optimal classifier, Bayesian model averaging, Bayesian model combination, Bucket of models, Stacking, Boosting and Boostrap aggregating (bagging). In 3.3.2, we compare two ensemble models, XGBoost and Random Forest, which implements Boosting and Boostrap aggregating respectively.

2.1.1 Boosting

As the name suggest, this approach boost the performance of weak learners by converting it into a strong learner. This is achieved when identifying instances that are misclassified, re-weight to favorite them and passing them to subsequent learners. One of the first examples of this approach is AdaBoosting [6], whose authors

claim do not tend to over-fit, followed by algorithms such as AnyBoosting and XGBoost, among others. It's worth to mention that this type of ensemble model has its root in the PAC (Probably Approximately Correct) learning framework.

2.1.2 Boostrap aggregating (bagging)

In contrast to Boosting, learners are training on m resample sets, with replacement, and their results are then averaged or aggregated. Leo Breiman coined the term bagging from Boostrap aggregating in his work [7] and was also the creator of Random Forests algorithm [8].

2.2 Sequential Models

From probabilistic graphical models to deep learning, sequential modeling states several challenges when analysing data types such as time series, signal processing, language generation, etc. We give a brief overview of a probabilistic graphical approach such as CRF and a deep learning approach such as Recurrent Neural Networks (RNN) as the base componente of encoder-decoder for seq2seq architectures.

2.2.1 Conditional Random Fields

Conditional Random Fields (CRF) were proposed by [9] in 2001. They are a type of undirected and discriminative probabilistic graphical model used for structured prediction, which work with more than one prediction dependent on each other. It was created as an alternative to the dependency of only the previous state as in the generative models, hidden Markov model (HMM) and maximum-entropy Markov model (MEMM), to better fit sequential modeling and address the label bias problem. Besides, CRF deals with the disadvantage of biased prediction when there are few discrete states. As defined in [9], (X, Y) is a conditional random field when, conditioned in X, the random variable Y_v obeys the Markov property, where, $Y = (Y_v)v\epsilon V$ in the graph $G = (V, E)$ and X can follow any other natural graph structure. As many other probabilistic models, the learning is performed by maximizing the likelihood. Figure 2.1 shows the comparison between structures of HMM, MEMMs and CRFs.

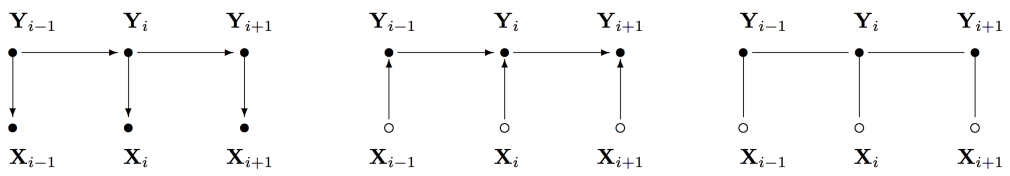


Figure 2.1: HMMs (left), MEMMs (center), CRFs (right) by [9]

Several works have used or adapted CRF to other type of problems such as in semantic image segmentation in computer vision [10] and combined with deep learning for entity recognition and word recognition in natural language processing [11], [12], respectively.

2.2.2 Recurrent Neural Networks

The temporal dependence between the data instances in a sequence prediction problem makes recurrent neural networks (RNNs) an appropriate fit for forecasting problem in smart cities. For example, to address water consumption prediction and emergency resolution time prediction, respectively, our frameworks in 4.3 and 5.3 make use of seq2seq RNN-based models. In this section we describe the main architecture of such frameworks, as well as the RNN component. An RNN consists of a hidden state h and an output Y that operates on input X. At each time step t, the hidden state of the RNN is given by, $h_t = f(h_{t-1}, x_t)$ where, f is any non-linear activation function and $1 \leq t \leq n$. In this work, f follows a neural network architecture comprising of a network of nodes organized into sequential hidden layers with each node in a given layer being full connected to every other node in the next successive layer. Each hidden state serves as memory and its output is calculated using the output of the previous hidden state and the input x_t as shown in Equation 5.1. Since RNNs are known to suffer from the vanishing or exploding gradient problem [13] when sigmoid functions are used, our architecture uses LSTM cells that use memory cells to store relevant information needed to learn long range temporal dependencies in the data. We refer the reader to Goodfellow et al. for more details on RNN [14].

2.2.3 Seq2seq Models

Seq2seq Models were created to perform machine translation based on recurrent neural networks (RNN). We describe here the general process followed by these type of models. First a representation of a word is feeded to every step of an encoder, that can be a recurrent neural network, and all the words are encoded in a vector c. Then, the vector c is feeded to a decoder, that can also be a recurrent neural network, which will output a sequence of words which can be consider a sentence. Two variants were proposed around 2014 by [15], [16] where the encoder and the decoder correspond to the same or different recurrent neural networks, respectively. In other words, having different recurrent neural networks, allows to have different type of RNN such as Long Short Term Memory (LSTM) or Gated recurrent unit (GRU) cells, as well as different parameters to be learned. Seq2seq models have lately been used in multistep forecasting. Figure 2.2 shows the type

of model where an LSTM works as an encoder and an RNN works as a decoder. Although an encoder-decoder architecture can be implemented in several forms (autoencoders, CNNs and other deep learning settings), we explain here in the description of seq2seq encoder-decoder context. As we can observe from Figure 2.2, the encoder takes an input sequence x_1, x_2, ..., x_n that corresponds to the resolution time of events for the last n time steps. The encoder then generates a hidden encoded vector C. After the entire input has been processed, the summary C is provided as input to the decoder. The decoder then generates $\hat{y}_1$, $\hat{y}_2$, ..., $\hat{y}_k$, the predicted resolution times for the future k events. The loss function used is the sigmoid activation function and it is applied to the output of the decoder. This ensures that the predicted values are in the [0-1] range.

The basic cell structure used in the encoder is LSTM that captures the important dependencies in the data. LSTM cells also possess the ability to 'forget' that enables them to overcome well-known vanishing/exploding gradient. To achieve this an LSTM cell has three main gates—input, output, and forget. The input gate receives the pertinent information in the current step and the output gate determines the hidden state for the next step. The forget gate is responsible for discarding unimportant information so that the model can capture the relevant long-term dependencies. We refer the reader to [15], [16] for additional details.

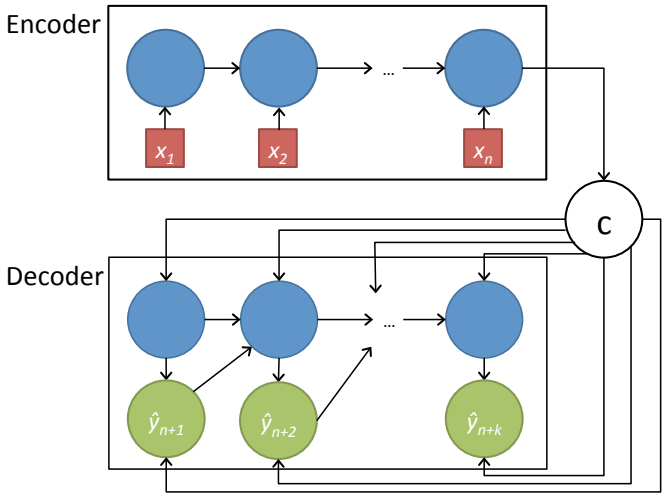


Figure 2.2: Encoder-decoder based RNN (figure taken from [2])

2.3 Deep Probabilistic Graphical Models

Recent works are identifying the potential of the combination of deep learning and probabilistic graphical models, leading to the Deep Probabilistic Graphical Models (DPGM) [17]. In other words, the combination of these types of models bring together the best of two worlds. On one hand, DL has the capacity of

learning a more flexible representation of abstract and complex relations of data. On the other, PG models bring to the table, a more controlable and explainable framework. Moreover, this is a valuable characteristic for AI systems that need to provide more reasons on their results in order to be more interpretable and subject of fair analysis. We consider variational models subject of improvement when combined with DL to estimate the parameteres of its family distributions and learning by optimization of great amounts of data. Our work in 6.4 also adapts a variational approach, which can be considered a kind of DPGM, over a recurrent neural network to address the energy disaggregation problem. For that reason, we provide more background on this kind of models by explaining one of the most simple examples such as variational autoencoders.

2.3.1 Variational Inference (Variational Bayesian Methods)

Variational Inference (VI) helps to learn intractable posterior distribution through an approximated distribution. To learn this proxy distribution, VI optimizes the distance between these distributions through KL divergence and makes use of the marginal loglikelihood of observable data to establish a lower bound. VI is an alternative to classical methods, such as Markov Chain Monte Carlo sampling due to its power and speed. However, it still has open problems and more understanding is missing on how to levarege its potential [18]. In the context of a DPGM, the proxy could be learned with neural networks.

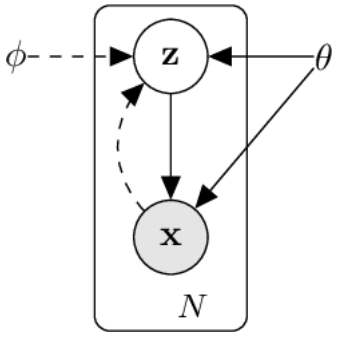


Figure 2.3: Variational Autoencoder by [19]

2.3.2 Variational Autoencoders

Variational autoencoders (VAE) [19] transform the encoded hidden vector of an autoencoder architecture in a latent variable to learn the parameters of a family of distributions. As seen in figure 2.3, there are three sets of probability distributions parameters to learn: a variational approximation (or encoder) of $P(z|x)$, a prior distribution for the latent variable z $P(z)$; and the decoder distribution of $P(x|z)$. As mentioned before, the approximated and other distributions can

be learned thorugh neural networks of different depth. Several variants of VAE has been used in computer vision, natural language and signal processing, to be able to generate images, text and time series data. Some of the recent work in computer vision include models such as PixelCNN, PixelRNN; for language generation, we have seen variants such as Variational Attention; and Variational Recurrent Autoencoders [20] and Variational Recurrent Neural Networks [5] for signal processing.

3 Water Pump Features Classification

Water availability and management is an important problem plaguing many developing and under-developed countries. Many factors including geographic, political, management, and environmental factors can affect the availability of water in these regions. In most developing regions, the primary means of extracting water is water pumps (e.g., manual, hand pump, solar pump). While significant initial capital is needed to install these pumps, a sustained long-term effort and investment is also required to maintain these pumps. For example, nearly 1.42 billion dollars have been donated to address the water access crisis in Tanzania [21]. Though significant strides were undertaken in installing pumps across the country, many of these pumps are in a condition of decay, primarily due to the lack of adequate maintenance.

Therefore, in this section, we adopt a data-driven approach to investigate important questions related to the *water availability:* water pump operation status, water quality and quantity by exploring two publicly available datasets related to the operation of pumps in Tanzania and Nigeria respectively [22], [23]. Specifically, we answer the following questions. *i)* How do individual or groups of features impact the water availability problem? *ii)* Can we accurately predict water pump operation status, water quality and quantity from the data?

Answering these questions is essential for the well being and economic growth of communities in these regions. They provide valuable information that can be used by the authorities to identify and effectively allocate scarce resources (e.g, money, personnel) to places most in need. For example, they provide insight into where to install new pumps depending on the water availability and which pumps are in need of immediate maintenance. Additionally, our analysis helps identify the most important features that influence the operation of a pump and thus can be extremely beneficial and used as a reference by other nations who plan to collect similar data in the future.

Specifically, our contributions are as follows:

1. We start with developing a predictive analysis framework that incorporates the different features to predict different problems related to water availability: i) pump operation status, ii) quality, and iii) quantity.

2. We then perform a two-pronged fine-grained feature analysis to understand the contribution of different features in predicting the different water problems. First, we group similar features into groups and drop/include feature groups individually and measure the corresponding effect on prediction performance. Geographic features and source-related features emerge as the most important feature groups across the different prediction problems. Next, we rank the individual features in order of importance to also understand their contribution toward prediction performance. This endeavor helps in understanding the importance of the different features/feature groups in the various water prediction problems.

3.1 Related Work

In this section, we outline existing work related to smart water management and contrast it with our work. Due to the lack of open datasets, there is limited research in this space. In [24], the authors study a relatively small dataset from Ghana's GAP region. They perform a Bayesian Network analysis and observe strong dependencies between pump functionality and features such as pump type and management. Jimenez *et. al* analyze the relationship between the functionality and the technology of the water points [25].

Another study focusing on Liberia, Sierra Leone, and Uganda [26] analyzes the risk factors associated with non-functionality of hand pumps. They apply a logistic regression model to a dataset of community-managed hand pumps and observe that age of the pump, distance from district/country capital, and absence of user fee collection all contribute significantly toward pump non-functionality. In [27] the authors investigate the performance of demand-driven, community-managed water supply systems in rural areas of developing countries through a large-scale study.

Water quality and quantity have also been investigated in prior work [28], [29]. The author in [30] build a multi-task, multi-view learning framework to predict urban water quality by combining a number of data sources including water hydraulic data, weather, pipe networks, structure of road networks, and point of interests (POIs). In another work, the authors report results from water quality measurement studies carried out in the rural districts of Tanzania [29].

Our work is closest to [23], where the authors explore the same two datasets and investigate the factors influencing the water pump functionality using regression and Bayesian Network analysis. From their analysis, they identify that hand pumps of a particular make have higher functionality in comparison to others. The find strong correlation between management type and functionality. In contrast

to existing work, we design a framework to predict water pump operation status, water quality, and quantity. We then use these models to identify the most important features that influence functionality. We note that similar machine learning techniques have been used to predict other environmental factors (e.g., air, river water quality, landslide) for enabling a smarter world [31]–[33].

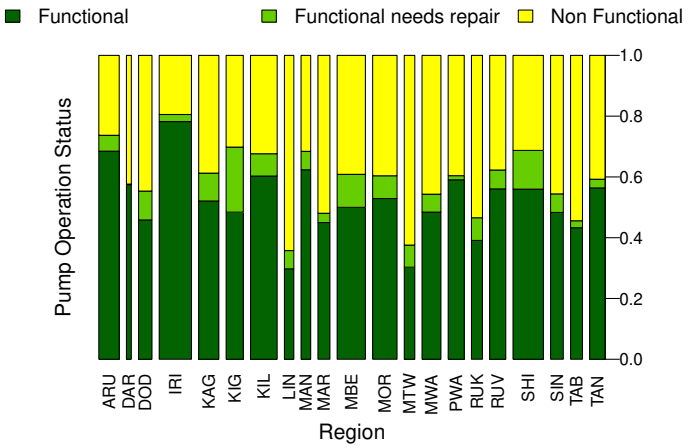
3.2 Dataset

We work with two datasets from Africa: Tanzania and Nigeria. Both of them have been made publicly available by Taarifa and the Tanzanian and Nigerian Ministry of Water, respectively [22], [23]. The Tanzania dataset was collected using hand-held sensors, paper reports, and feedback from people using cellular phones. This dataset has $59,401$ instances and contains information such as the pump operation status, water quality, water quantity, pump location, source type, extraction technique, and population demographics in the region where the pump is installed. The Nigeria dataset has $132,542$ instances and has features similar to, but less in comparison to the Tanzania dataset.

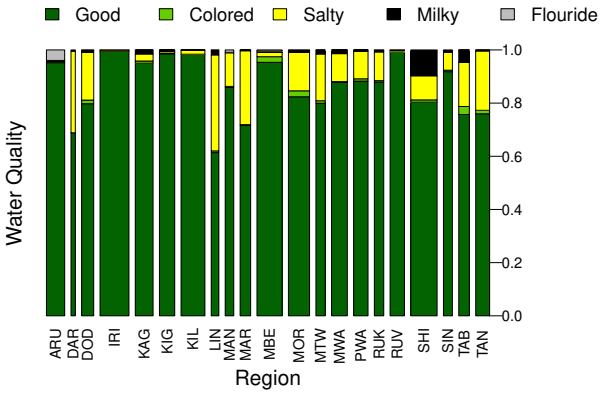
The primary difference between the datasets is that the Nigeria dataset does not contain information regarding the water quality and quantity. Therefore, we present significantly more results for Tanzania in comparison to Nigeria. For the Tanzania dataset, the pump operation status is described using three values namely *functional*, *functional needs repair* and *non-functional*, while for Nigeria the pump operation status is described using only two values *functional* and *non-functional*. The water quality for the Tanzania dataset is described using the values *good*, *milky*, *salty*, *colored* and *fluoride* while the water quantity takes the values *dry*, *enough*, *insufficient* and *seasonal*.

Figures 3.1a, 3.1b, and 3.1c show the normalized distribution of the water pump operation status, the water quality, and quantity for the different regions for Tanzania and Figure 3.2 shows the normalized distribution of the water pump operation status for Nigeria. The width of the bars in the figures denote the number of instances that correspond to a particular region or state. We make multiple important observations from these figures - *i)* the total number of instances vary with region/state, *ii)* there is a significant portion of pumps that are *non-functional*, and *iii)* there is an uneven distribution of the values for water quality and quantity. For example, if we consider water quality, a large fraction of instances have *good* value.

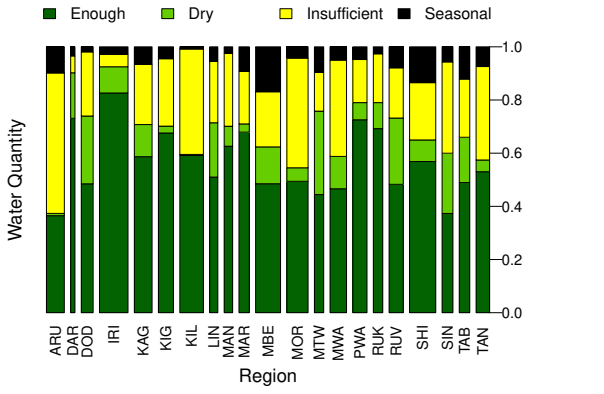
The pump operation status, water quality, and quantity all vary considerably with the different attribute types. To provide the reader a better understand of the data, we next compare the counts of the pipe operation status with respect to



(a) Pump operation status



(b) Water quality



(c) Water quantity

Figure 3.1: Tanzania: Comparison of pump operation status, water quality and water quantity for different regions

method used for extracting water for Tanzania and Nigeria in Figures 3.3 and 3.4 respectively. Note that the methods used for water extraction vary between the two countries.

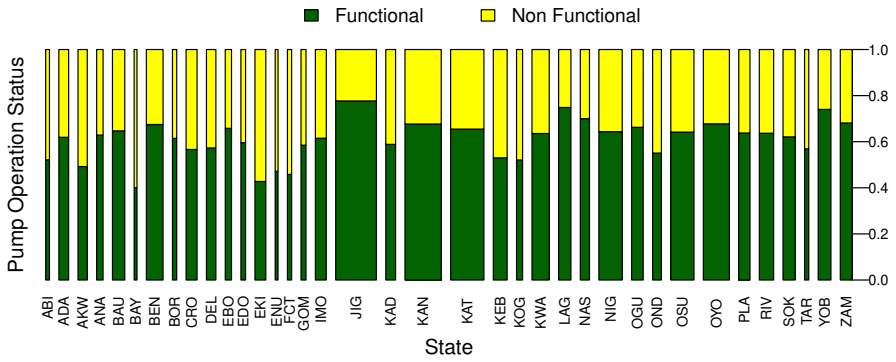


Figure 3.2: Nigeria: Comparison of pump operation status for different states

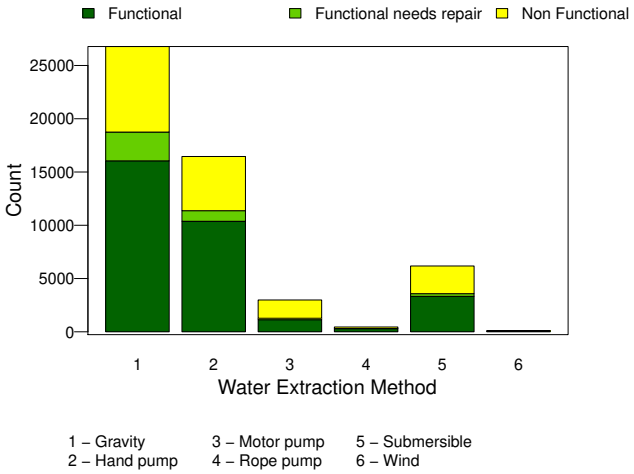


Figure 3.3: Tanzania: Comparison of pump operation status for different water extraction types

3.3 Smart Water Management Prediction Framework

Having provided an overview of the datasets in the previous section, we first describe the problems and then present predictive modeling approaches to address them. Our goal is to design a smart water management framework that can accurately predict the pump operation status, water quality and quantity. Our models can be potentially extended for addressing water-related issues in other developing and under-developed regions.

3.3.1 Feature Engineering

In order to design effective models for the problems studied in this section, we remove irrelevant features from the entire feature set. For example, we remove features such as the name of the water point or village name as they are either

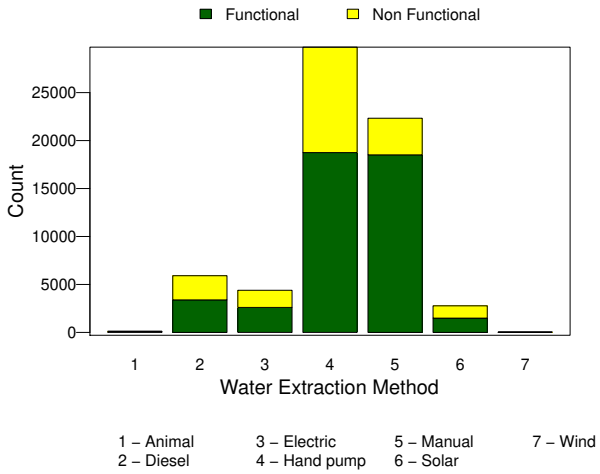


Figure 3.4: Nigeria: Comparison of pump operation status for different water extraction types

unique or shared between few instances in the dataset. We use Pearson correlation coefficient (PCC) and Spearman's rank correlation coefficient (SCC) to determine the correlation between features and class variables. The correlation values are used to eliminate features from the dataset that may not be useful. Additionally, we also convert few features to relevant units. For example, we convert the latitude and longitude values in the datasets to x, y, and z coordinates. Similarly, we use the year in which the pump was manufactured to determine the age of the pump. Missing values in the dataset are replaced by a measure of central tendency (i.e., mean, median) or not applicable (NA) depending on the appropriateness.

3.3.2 Predictive Models

We leverage two ensemble learning models, namely Random Forest and a recently developed ensemble model, XGBoost, to address the smart water management problem. Ensemble learning methods leverage multiple learning algorithms to obtain better prediction performance than what could be obtained from the respective individual learning algorithms in the ensemble.

Random Forest

It constructs multiple decision trees based on bootstrapping and random attribute selection during the training phase. The algorithm uses them to predict the the class during the test phase, and then outputs the result by carefully combining the results from the different trees [8]. Random Forest avoids overfitting by randomly selecting a set of attributes instead of taking all the available attributes

into consideration for constructing the trees.

XGBoost

In contrast to Random Forest, XGBoost uses dependent but smaller decision trees. It uses a gradient boosting algorithm to improve the results of the previous trees to predict the next tree. The final output is decided on the basis of a voting algorithm that is applied on the results obtained from all the trees [34].

3.4 Experimental Results

We conduct experiments to answer the following questions:

- How good are our models in predicting different attributes of water management: pump operation status, quality, and quantity in Tanzania, and pump operation status in Nigeria?
- What features/feature groups are most predictive of pump operation status, quality, and quantity?

In all our experiments, we use *5-fold* crossvalidation, where we divide the data into 5 partitions, iteratively train on four partitions and report the prediction performance on the fifth partition. We report standard performance metrics of precision, recall, and F1 score for all the models. *Precision* is defined as a ratio of the true positives to the sum of the true positives and false negatives and *recall* is the ratio of the true positives to the true instances in the dataset (i.e., the sum of true positives and the false negatives). The *F1 score* is calculated as the harmonic mean of the precision and recall. We compare our models with several baseline approaches such as Support Vector Machines (SVM), Logistic Regression, Multilayer Perceptrons, and Naive Bayes. We report results for SVM, the baseline model that performs the best on our dataset. Statistically significant differences evaluated at a rejection threshold of $p = 0.05$ are typed in **bold** in all the tables below. We measure statistical significance between XGBoost and Random Forest, wherever relevant. For scores where we cannot establish statistical significance between XGBoost and Random Forest, we report statistical significance with SVM. We observe that our proposed models achieve superior performance across all prediction tasks and across all the performance metrics.

3.4.1 Pump Operation Status Prediction

In this subsection, we report performance results for pump operation status for Tanzania and Nigeria. Tables 3.1 gives the performance results for the pump

operation status for Tanzania. We observe that Random Forest and XGBoost perform better than SVM across all performance metrics. Our models achieve a 78% performance improvement in F1 score over SVM for *non-functional*, 75% for *functional needs repair*, and 13% for *functional*, respectively. Looking closely at the results for individual class values, we observe from Table 3.1 that the performance of the proposed models is better for the *functional* and *non-functional* classes in comparison to the *functional needs repair* class for the Tanzania dataset. The main reason behind the lower performance for the *functional needs repair* class is the lack of enough instances pertaining to this class in our dataset (as shown in Figure 3.1a).

Similarly, we observe that XGBoost and Random Forest perform better than SVM on the Nigeria dataset. From Table 3.2, we observe that the F1 score performance is higher for the *non-functional* class in the Nigeria dataset in comparison to the *functional* class for all the three models. We observe that our proposed models achieve a performance improvement of 36% in the *functional* class when compared to SVM.

Model	**Class**	**F1 Score**	**Precision**	**Recall**
SVM	Functional	0.75	0.62	**0.94**
	Functional needs repair	0.24	0.62	0.14
	Non Functional	0.46	0.79	0.32
XGBOOST	Functional	**0.85**	**0.81**	0.91
	Functional needs repair	0.42	**0.63**	0.31
	Non Functional	**0.82**	**0.85**	0.78
Random Forest	Functional	0.85	0.81	0.88
	Functional needs repair	**0.43**	0.54	**0.36**
	Non Functional	0.81	0.84	**0.79**

Table 3.1: Tanzania: F1 scores, Precision and recall, and for predicting *pump operation status*

Model	Class	F1 Score	Precision	Recall
SVM	Functional	0.38	0.49	0.31
	Non Functional	0.74	0.68	**0.82**
XGBOOST	Functional	0.52	**0.57**	**0.47**
	Non Functional	**0.76**	**0.73**	0.80
Random Forest	Functional	**0.53**	0.50	0.32
	Non Functional	0.74	0.68	0.78

Table 3.2: Nigeria: F1 scores, Precision and recall for predicting *pump operation status*

Extraction Type	**Accuracy**	
	XGBOOST	Random Forest
Gravity	86.65%	79.21%
Hand pump	86.67%	72.88%
Motor pump	93.79%	90.86%
Rope pump	95.37%	85.64%
Submersible	93.31%	84.07%
Wind	84.31%	72.54%

Table 3.3: Tanzania: Accuracy of predictions across different extraction types

Extraction Type	Accuracy	
	XGBOOST	Random Forest
Animal	65.57%	63.92%
Diesel	65.06%	68.85%
Electric	64.47%	63.96%
Hand pump	64.70%	64.96%
Manual	65.03%	65.64%
Solar	63.70%	63.92%
Wind	86.66%	66.67%

Table 3.4: Nigeria: Accuracy of predictions across different extraction types

In Tables 3.3 and 3.4, we compare the accuracy of our XGBoost and Random Forest models in predicting pump operation status for the different extraction methods. *Accuracy* is defined as percentage of instances predicted correctly by our models in the total number of instances. We observe that both our proposed models can accurately predict the pump operation status for the different water extraction methods.

Model	**Class**	**F1 Score**	**Precision**	**Recall**
SVM	Good	0.94	0.90	0.99
	Bad	0.33	0.83	0.20
XGBOOST	Good	0.95	0.92	**0.99**
	Bad	0.49	**0.84**	0.35
Random Forest	Good	**0.96**	**0.95**	0.97
	Bad	**0.69**	0.78	**0.62**

Table 3.5: Tanzania:Precision, recall, and F1 scores for predicting *quality*

3.4.2 Quality and Quantity Prediction

Recall that water quality and quantity measurements are only available for the Tanzania dataset. From Figure 3.1b, we observe that majority of data instances correspond to *good* water quality, while for the remaining data instances the water quality is spread across *salty, fluoride, colored,* and *milky.* As the number of

Model	Class	F1 Score	Precision	Recall
SVM	Dry	0.49	0.87	0.34
	Enough	0.77	0.64	**0.97**
	Insufficient	0.35	0.76	0.23
	Seasonal	0.46	**0.83**	0.31
XGBOOST	Dry	0.84	**0.90**	0.79
	Enough	**0.89**	0.85	0.94
	Insufficient	**0.77**	**0.82**	0.72
	Seasonal	**0.73**	0.82	**0.66**
Random Forest	Dry	**0.85**	0.86	**0.82**
	Enough	0.89	**0.87**	0.92
	Insufficient	0.77	0.80	**0.74**
	Seasonal	0.72	0.79	0.66

Table 3.6: Tanzania: Precision, recall, and F1 scores for predicting *quantity*

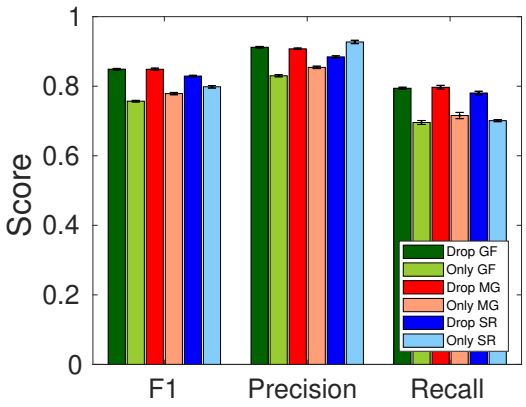
instances in *salty, fluoride, colored,* and *milky* classes is limited, we group them into *bad* water quality class. In comparison, from Figure 3.1c, we observe that there are sufficient data points in all classes for predicting quantity. Hence, in our quantity prediction models, we consider all the four different quantity classes.

Table 3.5 shows the performance results for water quality. We observe that predicting *bad* quality is a more challenging prediction problem as there are lesser number of instances for *bad* quality as opposed to number of instances for *good* quality. Here, our ensemble models achieve a significant improvement in the F1 score when compared to the SVM model, giving a performance improvement of 109% for the *bad* class.

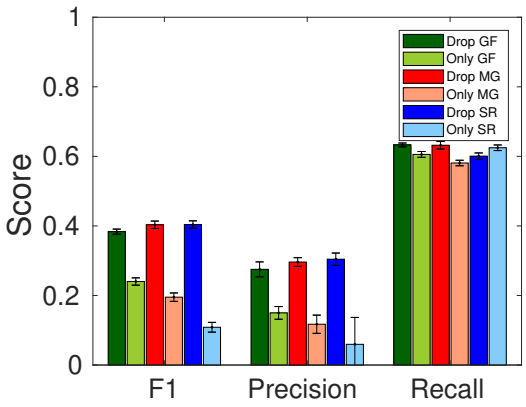
Table 3.6 shows the performance results for predicting water quantity. Here again, we observe that our ensemble models achieve a superior prediction performance in F1 score when compared to SVM, significantly improving the prediction performance for *insufficient* and *dry* classes by 120% and 71%, respectively.

3.4.3 Fine-grained Feature Analysis

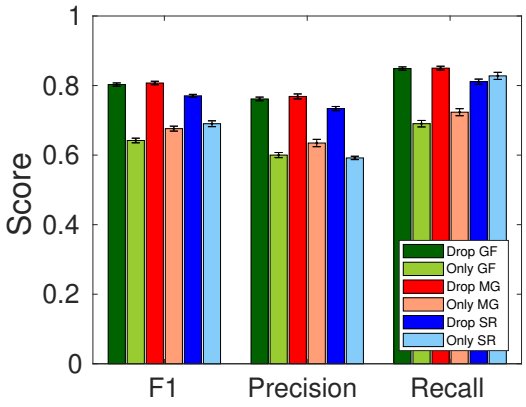
In this section, we perform a fine-grained feature analysis by: i) leaving groups of features out and including specific feature groups and examining the corresponding effect on performance, and ii) ranking features based on their contribution in the prediction problem. Our feature analysis is especially useful when extending the prediction models to new datasets/regions where only a subset of the features are available.



(a) Performance scores for *functional*



(b) Performance scores for *functional needs repair*

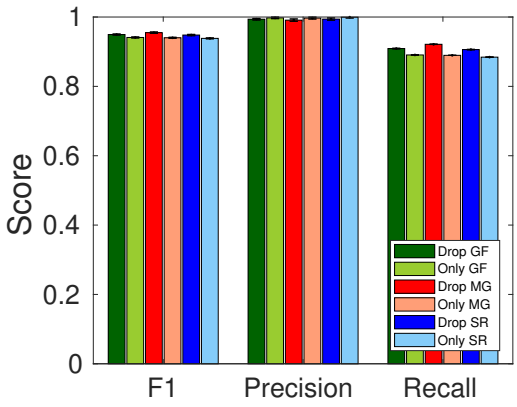


(c) Performance scores for *non-functional*

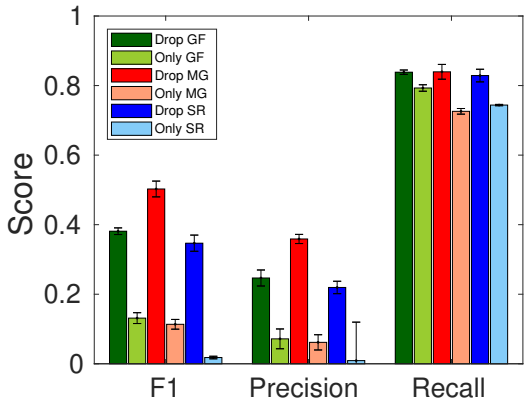
Figure 3.5: Tanzania: Performance scores for *functional*, *functional needs repair* and *non-functional* when a group of features are dropped or only when a group of features is included.

Feature Group Analysis

Next, we examine the effectiveness of different feature groups in the above-mentioned prediction problems. We first conduct a fine-grained feature analysis to investigate the impact of the contribution of features groups on prediction per-

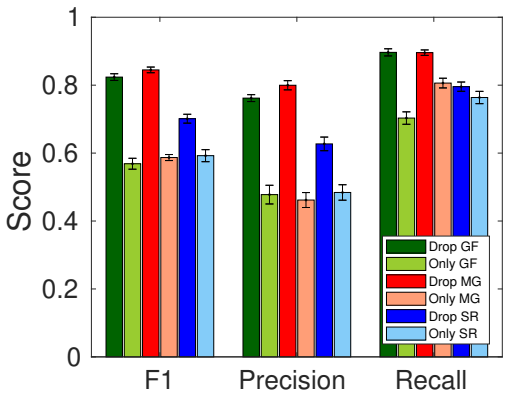


(a) Performance scores for *good*

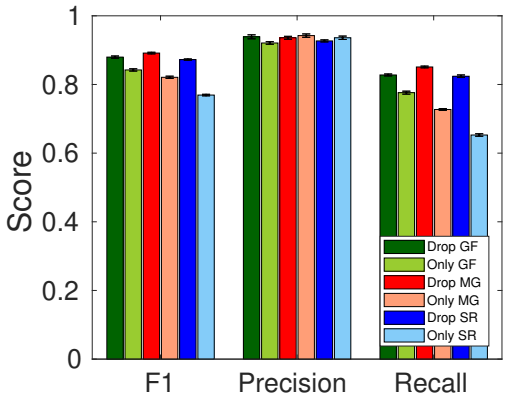


(b) Performance scores for *bad*

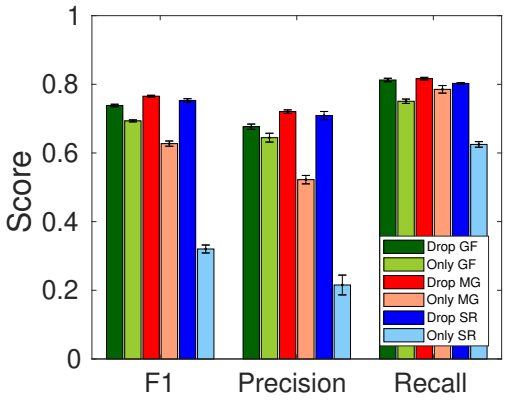
Figure 3.6: Tanzania: Performance scores for *good* and *bad* water quality when a group of features are dropped or only when a group of features is included.



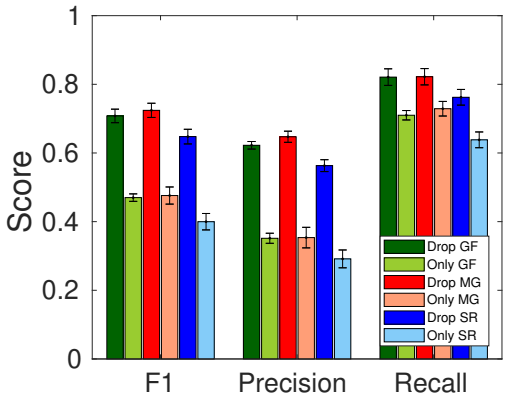
(a) Performance scores for *dry*



(b) Performance scores for *enough*

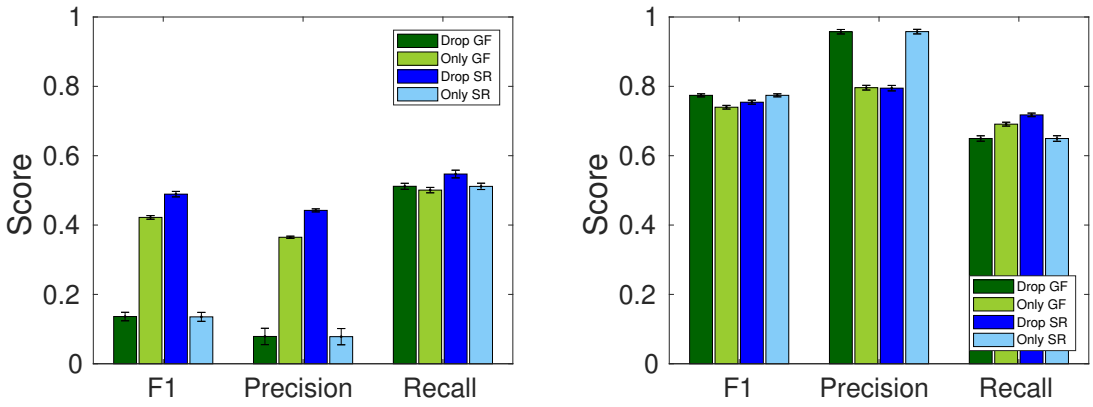


(c) Performance scores for *insufficient*



(d) Performance scores for *seasonal*

Figure 3.7: Tanzania: Performance scores for *dry, enough, insufficient* and *seasonal* water quantity when a group of features are dropped or only when a group of features is included.



(a) Performance scores for *functional*

(b) Performance scores for *non-functional*

Figure 3.8: Nigeria: Performance scores for *functional* and *non-functional* when a group of features are dropped or only when a group of features is included.

formance and then extend this to identify the most important individual features affecting performance in each of the prediction problems. To conduct this analysis, we group similar features into features groups. For Tanzania, we identify three feature groups: *i)* geographic features (GF), *ii)* management features (MG), *iii)* source related features (SR). The GF group includes features such as location (x, y, z coordinates) and the GPS height. The MG group includes features such as the installer of the pump and the entity responsible for maintaining the pump. The SR group contains features related to the water source including quality, quantity and water extraction method. We do a similar grouping of features in the Nigeria dataset. The Nigeria dataset only contains geographic (GF) and source related (SR) groups.

We investigate the predictive power of the various feature groups by adopting a two-pronged approach - *i)* dropping a particular feature group, *ii)* including only a particular feature group and examining the effect on performance. Dropping a particular feature group helps us understand how excluding it can adversely impact performance. In comparison, only including a particular feature group helps us appreciate the predictive power of the feature group when applied alone. Figures 3.5a, 3.5b, and 3.5c capture the performance impact of leaving feature groups out and including only a single feature group for pump operation status for the three classes: *functional*, *functional needs repair*, and *non-functional*, respectively. We observe that dropping the SR feature group has a greater performance impact for the *functional* and *functional needs repair* classes in comparison to dropping the other two feature groups, while the GF feature group has the highest impact for the *non-functional* class. Similarly, including only the SR feature group achieves the highest prediction performance for *functional* and *functional needs repair* classes, while including the GF feature gives the highest prediction performance for the

non-functional class. On average, we observe that the SR and GF individual feature groups have the highest impact on the *functional* and *functional needs repair* classes and the *non-functional* classes respectively.

Figures 3.6a and 3.6b capture the effect on prediction performance when specific feature groups are dropped/included in water quality prediction for *good* and *bad* class values. Again, we observe that dropping the SR group features have the highest impact on average on the water quality. We also observe that including only one of the feature groups provides poor performance for the *bad* class in comparison to the *good* class as predicting the *bad* class is a more challenging prediction problem. We observe that the prediction performance drops the most when SR feature group is excluded. Another interesting observation is that including just the SR group features does not give a high prediction performance, even though excluding them affects the performance the most. We conclude that GF features (which emerge as the strongest group of features individually) together with SR features are two most helpful feature groups for this prediction problem.

We observe a similar phenomena for quantity prediction in Figures 3.7a, 3.7b, 3.7c, and 3.7d. Overall, we observe that dropping SR features has highest affect on performance across most quantity class values. But, including just geographic features gives the highest performance when compared to including SR or MG feature groups. From this we conclude that GF and SR features are the two most helpful feature groups for this prediction problem and including them both is essential to achieve a good prediction performance. Across all the feature analysis experiments for Tanzania, we observe that the contribution of MG group in the prediction performance is more prevalent when the GF and SR feature groups are not available (i.e., including only MG feature group).

Figure 3.8 shows the leave one group out and include only a single group analysis for the pump operation status for the Nigeria dataset. We observe from the figure that the GF feature group plays a crucial role in predicting *functional* class for the Nigeria dataset. In comparison, it is hard to pick a winner for the *non-functional* group.

Feature Importance Ranking

Having studied the impact of the different feature groups on performance, we next investigate the contribution of individual features across pump operation status, quality, and quantity prediction in Tanzania and pump operation status prediction in Nigeria in Table 3.7. We observe that x coordinate (denoted by position_x), y coordinate (denoted by position_y), z coordinate (denoted by position_z), gps_height are the most predictive features across all the prediction

problems. This helps us in understanding the most important features and thus can be used as a reference for collection of similar data in future.

Rank	**Tanzania**			**Nigeria**
	Pump Status Class	Quantity Class	Quality Class	Pump Status Class
1	position_x	position_z	position_z	gps_height
2	position_y	position_x	position_x	position_x
3	position_z	position_y	position_y	position_y
4	gps_height	days_recorded	days_recorded	position_z
5	days_recorded	gps_height	gps_height	lga
6	population	funder	funder	city
7	funder	installer	installer	waterpoint_type
8	installler	population	population	extraction_type
9	age	lga	lga	-
10	lga	age	age	-

Table 3.7: Feature Importance Ranking for Tanzania and Nigeria across prediction problems.

4 Multi-step water consumption prediction

With climate change exacerbating extreme weather conditions including droughts and famines [35], understanding and predicting human water consumption is critical for ensuring a sustainable future. For example, the state of California, USA experienced one of its longest droughts from December 2011 to March 2019 [36]. Similarly, in recent times, the city of Capetown, South Africa was faced with a severe water crisis, where it was about to run out of drinking water for its citizens [37]. Therefore, predicting future water consumption in residential and commercial buildings has become an extremely important problem, particularly to efficiently monitor water consumption, identify possible leaks, minimize wastage, and match demand and supply. However, despite this need to design intelligent solutions to facilitate smart water usage, there is limited prior research from the computing community in this research area [38], [39].

Therefore, we design **SWaP**, a **S**mart **Wa**ter **P**redict-ion system, which predicts future hourly water consumption based on historical data. The water consumption prediction problem can be viewed as a classic time series prediction problem, thus making it amenable to statistical methods such as ARIMA as well as recently developed machine learning methods. To enable **SWaP** make effective predictions, we explore two classes of discriminative machine learning models, PG models and DL models, that have been shown to be effective for multiple time-series prediction problems [4], [40]. We design a structured regression graphical model, Gaussian Conditional Random fields (GCRFs), to successfully encode dependencies between historical and future water consumption [9]. Specifically, we leverage and adapt a recently developed sparse and computationally efficient variant of GCRFs [41]. We also design a Long Short-Term Memory (LSTM) based recurrent neural network (RNN) model that captures the underlying patterns in water consumption data.

The proposed GCRF model is parsimonious in nature and captures the underlying dependencies between the input (i.e., the past water consumption data) and output variables (i.e., the future water consumption predictions) as well as those between the output variables. As we construct a sparse GCRF model, the model only learns the necessary dependencies among the input and output variables that are helpful in the prediction. In comparison, the proposed DL model consists of

an encoder and a decoder, each of which separately is an RNN. The encoder takes past water consumption data and computes a state vector that encodes the underlying dependancies in the data. The decoder then utilizes this state vector to generate water consumption predictions.

To evaluate the performance of **SWaP**, we collect hourly water consumption data for 14 buildings from a university campus for the Fall 2018 semester (approximately 4.5 months). We classify these buildings into 4 categories—academic building, dining hall, gym and residence hall. The buildings in the dataset comprise of 6 academic buildings, 1 dining hall, 1 gym and 6 residence halls. We compare the performance of **SWaP** with linear regression and ARIMA baselines with respect to the Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) and demonstrate that **SWaP** significantly outperforms the baselines. The GCRF and LSTM based deep learning models in **SWaP** provide an average improvement of 50% and 44%, respectively. Additionally, we demonstrate that augmenting our models with temporal features such as time of the day and day of the week can improve the overall average prediction performance.

We note that both the GCRF and deep models only require the past 24 hours of water consumption data to predict future water consumption at test time, thus making **SWaP** an attractive system that can be readily deployed in practice. Additionally, our experiments also show that the GCRF model provides overall better performance than the LSTM-based deep model. Therefore, based on our experiments, we recommend using a GCRF-based **SWaP** for the hourly water consumption prediction problem. The superior performance of GCRF models along with its low computational requirement during the training and execution phases makes **SWaP** a highly desirable and practically feasible prediction framework. Moreover, the sparse GCRF model only captures the necessary dependencies between the input and output variables, thus making the GCRF-based **SWaP** inherently interpretable.

4.1 Related Work

In this section, we first outline research related to addressing water management problems, and then review literature related to forecasting applications in the ubiquitous computing domain.

To mitigate the negative impacts of climate change, a number of recent research initiatives have focused their attention on water management related problems [38], [39], [42]–[44]. Short-term forecasting of water consumption based on water meter readings is conducted in [45], while neural network based models for daily water demand forecasting on a touristic island is proposed in [43]. Assem

et al. use DeepCNN to predict urban water flow and water level based on input features such as maximum temperature, minimum temperature and run-off [39]. Bejarano et al. design a random forest and SVM based framework to investigate the availability of water pumps in developing and under-developed regions [46]. Similarly, logistic regression and Bayesian analysis have been applied to understand the factors associated with the non-functionality of hand pumps [24], [26]. Prior work has also investigated the interaction and use of water with other resources such as energy and food, popularly known as the water-energy-food nexus [47]–[50]. In comparison to existing research, we propose GCRF and LSTM based deep learning models for water consumption prediction and validate the efficacy of the models using real-world data collected from multiple buildings in a university campus.

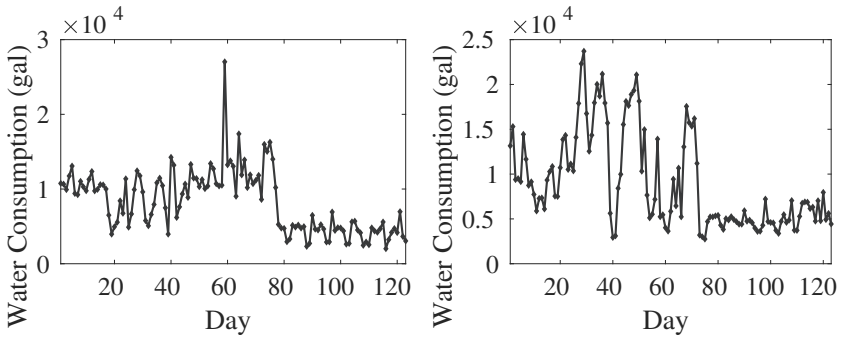
Recently, a variety of different models including statistical models such as ARIMA [51], [52], evolutionary algorithms [53] and data-driven approaches [4], [54], [55] have been applied to variety of forecasting and smart computing tasks. Arjunan et al. design a framework called OpenBAN for electricity demand forecasting leveraging algorithms such as decision tree, neural networks, SVM, naive bayes and k-NN [54]. Deep learning models for crime prediction from multi-modal data and spotting garbage from images has been proposed in [56] and [57], respectively. Mobility and traffic flow modeling at the city level has been explored in [58]–[60]. Similarly, model-based and machine learning techniques have also been proposed for solar power and irradiance forecasting [61], [62].

4.2 Problem Statement and Data

In this section, we discuss the water consumption prediction problem and provide an overview of the data collected to validate the performance of our model.

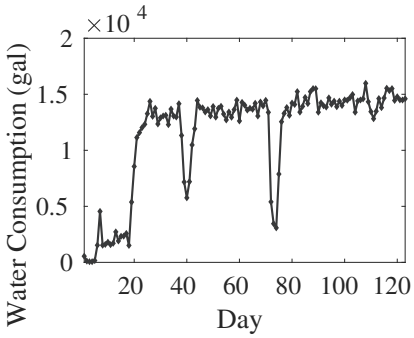
4.2.1 Problem Statement

Our goal is to design a system to predict hourly water consumption based on real-world data collected from multiple buildings in a university campus. Water consumption forecasting can be modeled as a classic univariate time series forecasting problem, where at any time T, the goal is to predict water consumption k steps into the future (i.e., $\hat{y}_{T+1}$, $\hat{y}_{T+2}$ ..... $\hat{y}_{T+k}$) based on data available for the past n time steps (i.e., x_T, x_{T-1}..... x_{T-n}). Note that $\hat{y}_{T+i}$ denotes the predicted value of the actual water consumption y_{T+i} at time $T+i$. As the problem studied here can be cast as a time series forecasting problem, both statistical techniques such as ARIMA and recently developed data-driven and machine learning approaches can be leveraged and adapted to address this problem. In this section, we de-

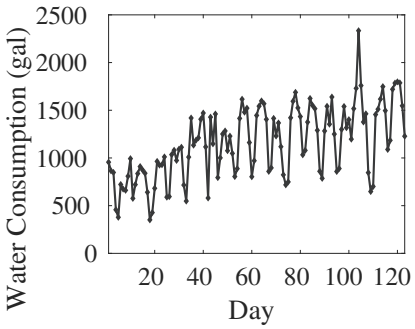


(a) Gym GE

(b) Dining Hall C4



(c) Residence Hall RA



(d) Academic Building S2

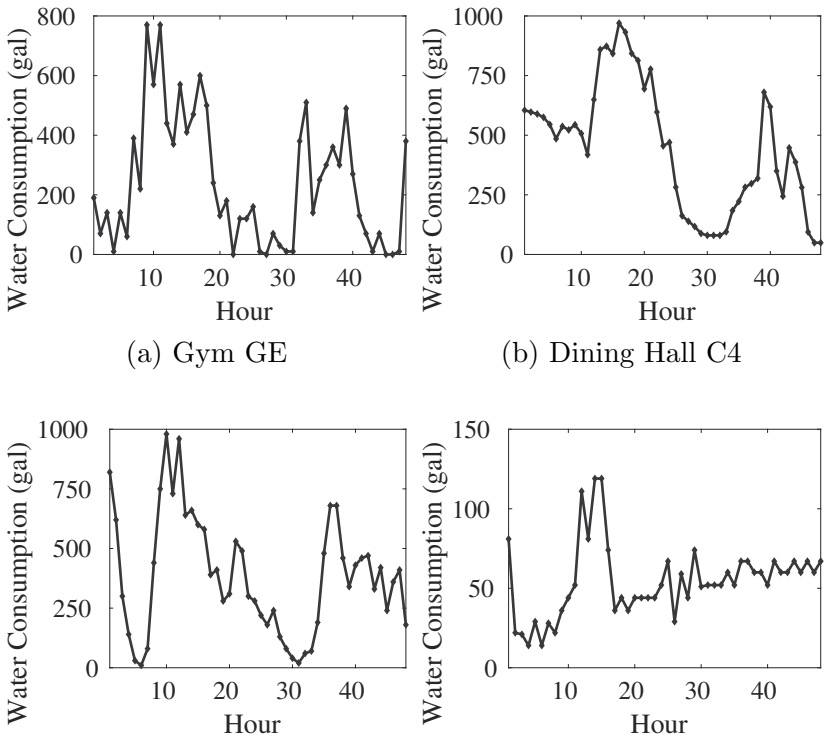
Figure 4.1: Trends in datasets (daily consumption)

velop sequence-to-sequence probabilistic graphical and deep learning models for the water consumption prediction problem and demonstrate empirically that they perform better than ARIMA models. We discuss our rationale for choosing the above-mentioned models and the details of our system in Section 4.3.

4.2.2 Data

We collect hourly water consumption data for 14 buildings in a university campus. These buildings fall into 4 categories— academic building, dining hall, gym, and residence hall. The buildings in the dataset comprise of 6 academic buildings, 1 dining hall, 1 gym and 6 residence halls. We collect data for approximately 4.5 months when the university is in session, beginning from August 1, 2018 to December 8, 2018 (i.e., Fall 2018 semester). Therefore, we have approximately 3000 data points for each building. Table 4.1 shows the median hourly and daily water consumption for all buildings.

We discuss the general trends in water usage for buildings in each category. Figure 4.1 shows the daily water usage for one representative building in each category for the entire time period. We observe that during the last two months, the total water consumption decreases for the dining hall and gym (Figures 4.1a and 4.1b). While the exact reason is unknown, based on the timing, we hypothesize



(a) Gym GE (b) Dining Hall C4

(c) Residence Hall RA (d) Academic Building S2

Figure 4.2: Trends in datasets (hourly consumption)

Table 4.1: Median Water Consumption

Building	**Category**	**Median (Hourly)**	**Median (Daily)**
EB	Academic Building	33	1765
FA	Academic Building	49	2478
LH	Academic Building	72	4029
S2	Academic Building	43	1148
S3	Academic Building	357	12413
SN	Academic Building	243	6455
C4	Dining Hall	304	7029
GE	Gym	280	8420
BN	Residence Hall	380	10180
BR	Residence Hall	220	5920
DE	Residence Hall	500	12910
DG	Residence Hall	410	11230
JS	Residence Hall	520	13510
RA	Residence Hall	490	13490

that this could be related to air-conditioning, cooling/heating. We also observe that residence halls have lower water consumption for the first 20 days (Figures 4.1c). This is because residence halls open from 20th August after the student orientations. In comparison, as academic buildings are in use throughout the year, we observe that the water consumption remains in the same range throughout the year (Figures 4.1d).

Figure 4.2 shows the hourly water usage for 48 hours (September 6 and 7) where hour 1 and hour 25 correspond to the time between 12 am and 1 am for two consecutive days. We observe that gym and dining hall have highest water usage from 9 am to 9 pm (which approximately corresponds to the time duration for which these facilities are open). Water consumption for residence halls drops at night for around 5 hours when most students are asleep. In comparison, academic buildings have water consumption in the same range throughout the day. We hypothesize long/late working hours of graduate students and cooling needs for equipment to be the main reason for this behavior. We note that most utilities including water and electricity are shut down during Thanksgiving week for all campus buildings. As water consumption values mostly correspond to zeroes during this week, we remove the Thanksgiving week values to prevent possible misrepresentation in the model due to this data. Additionally, the dataset has around 0.3% missing values. We use linear regression to fill in these missing values.

4.3 SWaP: Smart Water Prediction

In this section, we provide an overview of **SWaP**, a **S**mart **Wa**ter **P**rediction system that takes as input historical water consumption data and outputs future water consumption predictions. Figure 4.3 shows the different components of our system. **SWaP** comprises of a data pre-processing component, which pre-processes the water consumption data and a prediction component consisting of the proposed models that takes the pre-processed data to generate the desired predictions. We design two models, a discriminative PG model and a DL model for the prediction component in **SWaP**. Specifically, we design i) sparse Gaussian Conditional Random Fields (GCRFs) and ii) Long Short Term Memory (LSTM) based deep Recurrent Neural Network (RNN) models to successfully encode dependencies in the water consumption data. At time T, both models accept an input sequence $X = [x_T, x_{T-1}, ..., x_{T-n}]$, which corresponds to amount of water consumed in the last n time steps and generate predictions $Y = [\hat{y}_{T+1}, \hat{y}_{T+2}, ..., \hat{y}_{T+k}]$ for the next k time steps. We note that the input and output sequences can be of different lengths.

4.3.1 Why Sequence-to-Sequence Models?

Traditional model-based and statistical approaches (e.g., ARIMA models, filtering techniques) provide valuable insights into data, and are highly desirable when limited computational resources and data are available to make decisions. The increase in computational power, the availability of large amounts of data, and growth in the field of machine learning presents the opportunity to design data-driven techniques capable of providing superior prediction performance in real-world settings. This provides us the opportunity to explore sequence-to-sequence models that are well suited for time-series data problems requiring mapping input sequences to output sequences. Sequence-to-sequence models possess the ability to predict an entire sequence of data points based on past data, thus being able to predict further into the future. To this end, we identify sequence-to-sequence probabilistic graphical (i.e., sparse GCRFs) and deep learning models that have been extensively used for a number of forecasting and prediction tasks [4], [39]. Both GCRF and deep models elegantly learn and capture non-linear dependencies as the encoded signal passes through the network, thus having a positive impact on prediction.

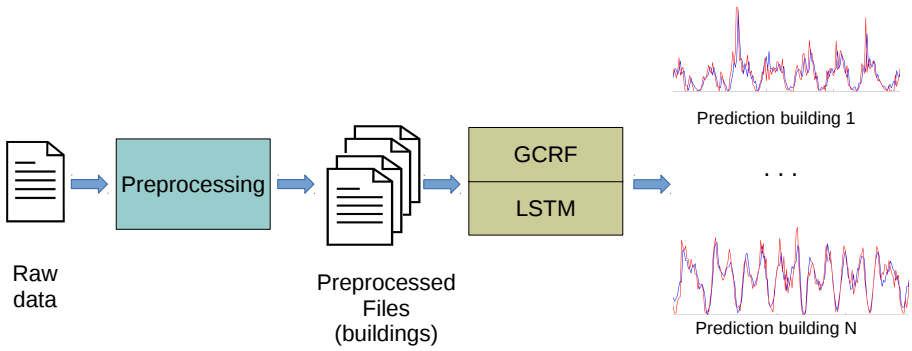


Figure 4.3: System Architecture

4.3.2 Sparse GCRF Model

In any machine learning model, a careful tradeoff between model complexity and prediction performance needs to be made to avoid overfitting and achieve good prediction performance. Hence, it is important to capture the dependencies that are important to the prediction, specially the ones in a structured output. As mentioned in 2.2.1, the discriminative approach of Conditional Random Fields are a perfect fit for this kind of prediction. Moreover, the regression version of CRF, GCRF, allows us to model the multi-step water consumption in the future, given previous water consumption. In our problem, we leverage a recent version

of CRF extended to structured regression, sparse Gaussian Conditional Random Fields (SGCRF) [41], for predicting future consumption.

The GCRF distribution is given by

$$P(Y|X;\Lambda,\Theta) = (1/Z(X)) * exp(-Y'\Lambda Y - 2X'\Theta Y) \tag{4.1}$$

where, $X = [x_1, x_2, ..., x_n]$ represents historical hourly consumption, n is the number of hours in the past, $Y = [\hat{y}_{n+1}, \hat{y}_{n+2}, ..., \hat{y}_{n+k}]$ represents predicted hourly consumption, and k indicates the number of hours in the future. We need to estimate Θ and Λ that are parameters/regression coefficients of the GCRF model. As also explained in [40], [41], Θ is an nxm matrix, containing the edges between X and Y and Λ is the mxm inverse covariance matrix, containing the edges amongst the y's. The CRF is a Gaussian distribution with mean $-\Lambda^{-1}\Theta'X$ and variance Λ^{-1}, $\mathcal{N}(-\Lambda^{-1}\Theta'X, \Lambda^{-1})$. $Z(X)$ in Equation 4.1 is the partition function, which ensures that the posterior is integrated to 1.

At training time, we estimate the parameters Θ and Λ by maximizing the probability of the data given the parameters using maximum likelihood,

$$\max_{(\Lambda,\Theta)} P(Y|X;\Lambda,\Theta)$$

This is equivalent to minimizing the log-likelihood,

$$\min_{(\Lambda,\Theta)} -log(P(Y|X;\Lambda,\Theta))$$

Regularization is a way to avoid overfitting by penalizing high-valued regression coefficients and helps in making models generalize better at test time. L_1 and L_2 are two popularly used regularization norms that add a penalty term corresponding to the absolute value of the magnitude of the coefficients and square of the magnitude of the coefficients, respectively. The total number of parameters in this problem for n historical time steps given by X and predicting k future time steps for Y is $nk + \frac{k(k+1)}{2}$, where nk edges are given by Θ, and $\frac{k(k+1)}{2}$ by Λ. Even for $k = 12$ (as is the case in our setting), it is possible that the model can overfit due to the large number of parameters.

To retain only meaningful dependencies, this sparse variant of GCRFs incorporates L_1 regularization. L_1 regularization reduces the parameter values of dependencies that do not contribute to the prediction to zero, thus creating sparsity in the graphical model structure. As part of L_1 regularization, a penalty term equal to the absolute value of the magnitude of the coefficients is added to the GCRF objective to penalize high-valued regression coefficients and avoid overfitting due to large number of parameters. L_1 is more preferred than L_2 here as it drives

less contributing parameter values to zero, thus completely removing their effect on the prediction. Thus, L_1 learns a model that is appropriately complex for the prediction problem.

We use the optimization method developed by Wytock et al. [41] to solve the GCRF with the L_1 regularization term. They develop a second-order active set method that iteratively produces a second-order approximation to the objective function without the L_1 regularization term, and then solve the L_1 regularized objective function using alternating Newton coordinate descent. For additional details, we refer the reader to [41]. Figure 4.4 gives the structure of the GCRF

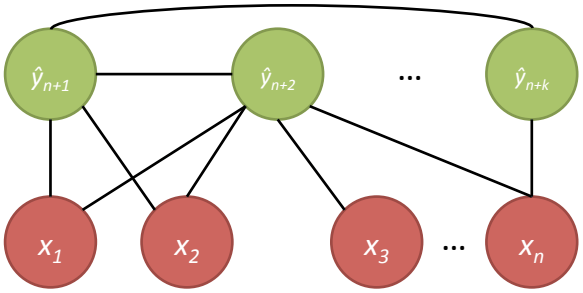


Figure 4.4: GCRF water consumption prediction model showing connections between historical consumption, x_1,...,x_n and y_{n+1}, ...,y_{n+k}, and among y_{n+1}, ...,y_{n+k}. Note that our model is sparse, learning only edges between variables that matter. In the graphical model, we illustrate this by leaving out some edges.

model. We can see that there are edges showing the dependencies between the inputs X and outputs Y. Also note that some edges in the graphical model have been left out to illustrate sparsity in the learned model.

Implementation Details

The GCRF training and test setup is given in Figure 4.5. We implement our models using *SGCRFPy*, a Python toolkit for sparse GCRFs[1]. We split the datasets into two parts—the first part consisting of 75% of the data is used for training and the remaining 25% is used for testing. At training time, our GCRF models use past n hours as input and next k hours as output. As water consumption patterns typically are likely to follow a 24-hour cycle, we use $n = 24$ and $k = 12$ in our experiments. The parameter Λ is initialized to the identity matrix and Θ is initialized to all zeros. We use regularization constant $\lambda = 0.1$ and train the model for $10,000$ iterations to converge on a set of dependencies learned from the training data. For each building, we train a separate GCRF model.

[1]Sparse GCRF implementation: https://github.com/dswah/sgcrfpy.

The parameter values Λ and Θ learned at training time are plugged into each test sequence of length n to generate a prediction for the next k hours. To compute the prediction performance scores, the predicted values $\hat{Y}$ are compared with the ground truth water consumption values Y. For a particular configuration of parameter values, the training time for GCRF is less than 5 minutes on a standalone lab machine. The RAM requirement for training is also low. The testing phase only takes a couple of minutes.

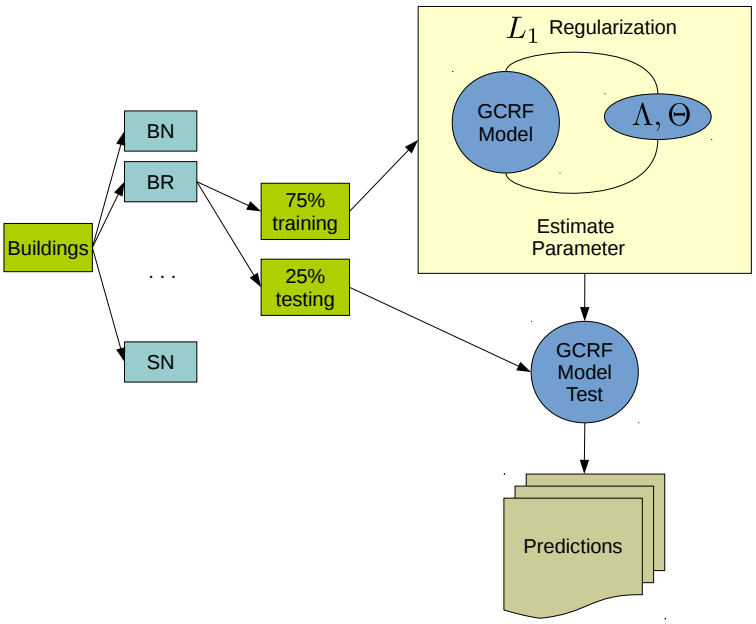


Figure 4.5: GCRF Training and Test Setup

4.3.3 RNN Encoder-Decoder Model

We develop an LSTM-based RNN encoder-decoder sequence-to-sequence model as as explained in 2.2. The architecture of both the encoder and decoder is an RNN. The basic cell in both the encoder and the decoder is an LSTM. The encoder accepts an input sequence and generates a hidden encoded vector c encapsulating the information for the input sequence. This encoded vector is given as an input to the decoder, which then generates the predictions. The input X is transformed into the output $\hat{Y}$ using the hidden layers and the weight matrices. The weight matrices essentially capture the information needed to generate the output predictions based on the input data. The LSTM cell used in our model consists of a number of interconnected gated units. The three gates in an LSTM cell are namely, the input gate, the output gate, and the forget gate that lets it handle long-term dependencies. To prevent prediction of negative water consumption values, a ReLU activation function is used after each decoder output.

Implementation Details

We use TensorFlow[2] for implementing the deep learning models. As mentioned in Section 4.3.2, we similarly split the datasets into two parts—the first part consisting of 75% of the data is used for training and the remaining 25% is used for testing. Similar to the GCRF models, we use water consumption of past 24 hours (i.e., $n = 24$) and predict 12 hours into the future (i.e., $k = 12$). These settings ensure that the results from both these models are directly comparable. As deep models are computationally expensive, we train our models on a shared high performance computing cluster available at our university. Using this cluster, we are able to execute 10 to 15 experiments in parallel. Each experiment is allocated 4 cores and 10 GB of RAM. For the datasets considered in this work, for a particular configuration of parameters, training the deep models (i.e., a single experiment) can take in the order of 1 - 12 hours, which is typical of deep learning models.

We experiment with different number of stacked layers, different numbers of hidden units in each layer as well as the lengths of the input and output sequences. We observe that depending on the dataset, different parameter configurations provide the best performance. However, we empirically observe that overall 1 stacked layer with 200 hidden units generalizes better across all the buildings. We use a learning rate of 0.01 and train the model for 1000 epochs. At training time, the encoder and decoder are trained jointly using the backpropagation algorithm. We use unguided training as the training scheme, where the decoder uses previous predicted output value as an input to the next step of the decoder. Unguided training enables the model to better explore the state space, which usually results in superior prediction performance at test time. Additionally, to minimize overfitting, we incorporate L2 regularization in the models.

In comparison to training, the testing phase of a model takes only a couple of minutes for each experiment. The learned weight values for the different connections in the neural network are used to generate a prediction for the test data instances.

4.4 Performance Evaluation

We compare the performance of GCRF and deep learning models with two baselines—linear regression and Auto-Regressive Integrated Moving Average (ARIMA) models. The code for our models, the pre-processed data, and the experiments is available in [63].

Linear Regression: It is a simple statistical model that fits the best straight line based on the input data.

[2]https://www.tensorflow.org/

ARIMA(p, d, q): It is a statistical model that has three components — AR (autoregressive term), I (differencing term) and MA (moving average term), which are specified by p, d and q respectively. p represents the past values used for predicting the future values, d represents the degree of differencing (i.e., the number of times the differencing operation is performed to make a series stationary), and q represents the number of error terms used to predict the future values. At any time T, the equation of ARIMA used for prediction is given by,

$$(1 - \sum_{i=1}^{p} \phi_i L^i)(1 - L)^d x_T = (1 - \sum_{i=1}^{q} \theta_i L^i) e_T \tag{4.2}$$

where x_T corresponds to the water consumption values, ϕ_i and θ_i are the auto-regressive and moving average parameters, e_T are the error terms and L is the lag term. The error terms e_T are assumed to be independently and identically distributed according to normal distribution. In our experiments, we use the Auto-ARIMA toolkit[3] in python that searches through a combination of the parameters p, d, and q, and picks the optimal combination for the data in consideration. As both linear regression and ARIMA are statistical baselines, they do not require any explicit training. They use water consumption for the past 24 hours to predict 12 hours into the future.

We use root mean squared error (RMSE) and mean absolute error (MAE) as the main evaluation metrics, which are given by Equations 4.3 and 4.4 respectively.

$$RMSE_j = \sqrt{\frac{\sum_{i=1}^{h} (\hat{y}_{ij} - y_{ij})^2}{h}} \tag{4.3}$$

$$MAE_j = \frac{\sum_{i=1}^{h} |\hat{y}_{ij} - y_{ij}|}{h} \tag{4.4}$$

where y_{ij} is the i^{th} test sample for j^{th} hour, $\hat{y}_{ij}$ is the predicted value of y_{ij}, and h is the total number of test samples.

4.4.1 RMSE

In this subsection, we discuss RMSE results for all models. Figure 4.6 shows the performance of the models for one building in each category. From Figure 4.6, we observe that GCRF and LSTM outperform the baselines significantly. We also observe that RMSE values for linear regression and ARIMA increase considerably with each predicted hour into the future. In comparison, the RMSE values increase gradually for the GCRF and LSTM models, which demonstrates that our models are able to predict considerably better into the future. We attribute this to the

[3]https://pypi.org/project/pyramid-arima/

Table 4.2: RMSE

(a) Hour 1

Building	**GCRF**	**LSTM**	**ARIMA**	**LR**
BN	**130.69**	140.08	175.71	314.12
BR	68.62	**65.44**	92.6	166.06
C4	101.64	**99.56**	123.83	233.66
DE	**151.17**	169.45	192.74	330.34
DG	**141.86**	168.11	200.16	330.06
EB	59.16	**58.64**	63	89.36
FA	**48.48**	57.91	63.27	85.49
GE	**119.88**	128.27	140.88	184.78
JS	**127.04**	173.75	183.13	361.19
LH	**80.55**	83.2	109.29	180.39
RA	**161.22**	178.15	230.01	400.57
S2	**18.78**	22.19	24.12	32.88
S3	**196.37**	259.56	269.45	400
SN	**112.79**	116.23	119.68	125.89

(b) Average

Building	**GCRF**	**LSTM**	**ARIMA**	**LR**
BN	**151.2**	154.95	345.31	406.11
BR	88.28	**84.89**	171.29	206.36
C4	**148.26**	164.81	256.28	316.19
DE	**184.95**	201.96	364.89	418.18
DG	**168.83**	190.53	329.99	398.63
EB	**70.5**	75.46	103.09	114.35
FA	**58.17**	81.93	107.35	111.26
GE	**138.81**	145.61	195.25	237.16
JS	**162.15**	190.96	380.72	461.5
LH	**116.77**	122.53	249.26	242.29
RA	**188.31**	203.88	412.93	488.15
S2	**23.26**	26.01	40.23	44.11
S3	**238.64**	313.26	516.58	536.86
SN	**124.43**	134.13	144.42	155.8

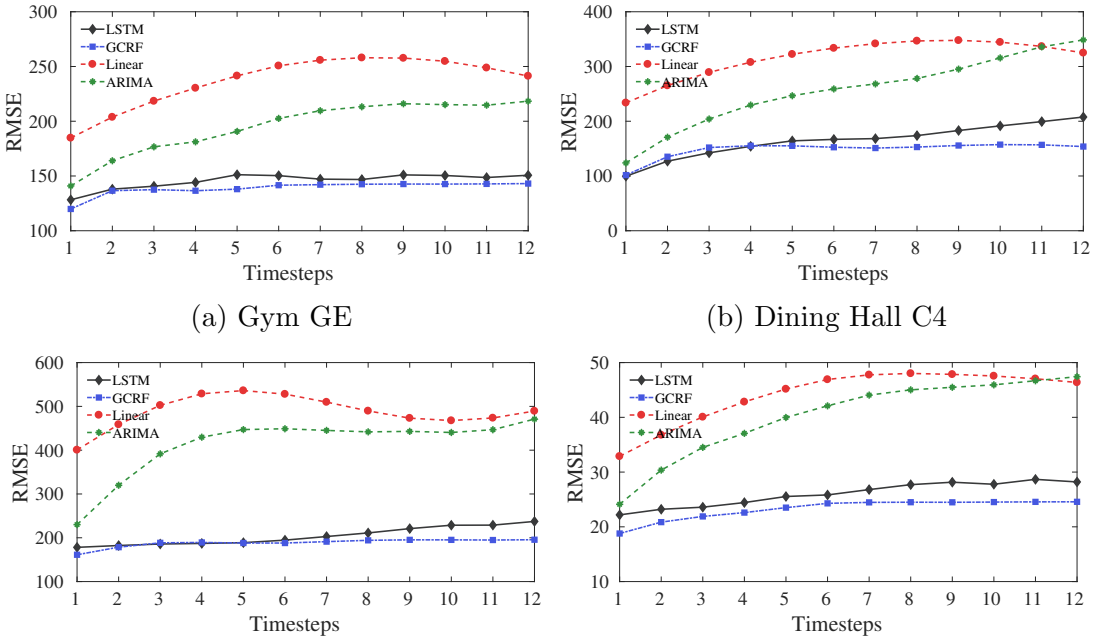
Table 4.3: MAE

(a) Hour 1

Building	GCRF	LSTM	ARIMA	LR
BN	**98.51**	105.04	138.36	254.13
BR	52.5	**49.44**	73.14	132.17
C4	68.52	**67.05**	82.81	185.04
DE	**105.55**	121.29	147.45	266.17
DG	**108.11**	122.9	158.01	262.01
EB	**26.48**	30.02	33.07	58.62
FA	**28.08**	36.51	38.51	64.97
GE	**77.25**	94.86	97.35	144.4
JS	**98.73**	134.3	146.78	291.42
LH	**51.54**	52.84	65.61	129.14
RA	**124.78**	132.38	179.4	316.81
S2	**13.54**	16.32	17.18	25.11
S3	**134.77**	165.3	179.33	295.57
SN	**66.02**	70.38	77.5	87.94

(b) Average

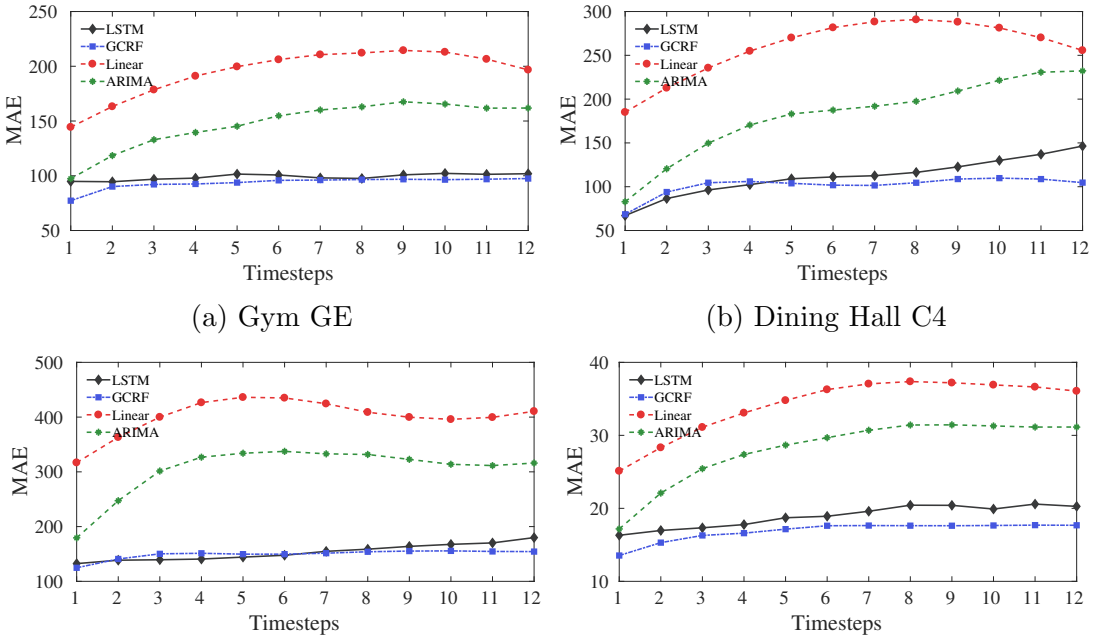
Building	GCRF	LSTM	ARIMA	LR
BN	**112.94**	115.82	257.6	341.42
BR	65.84	**63.82**	132.02	169.25
C4	**101.38**	111.46	181.42	259.52
DE	**133.9**	147.14	266.29	345.39
DG	**129.51**	145.06	259.23	327.47
EB	**37.09**	41.76	63.51	81.45
FA	**36.21**	51.58	71.64	89.36
GE	**93.48**	99.01	147.34	194.71
JS	**125.53**	149.52	291.68	383.13
LH	**74.62**	77.7	150.07	178.77
RA	**149.31**	153.18	304.59	401.4
S2	**16.87**	18.94	28.13	34.16
S3	**161.37**	203.31	325.96	414.75
SN	**80.42**	89.05	100.92	115.11



(a) Gym GE (b) Dining Hall C4

(c) Residence Hall RA (d) Academic Building S2

Figure 4.6: Average RMSE of 4 buildings through 12 predicted time-steps



(a) Gym GE (b) Dining Hall C4

(c) Residence Hall RA (d) Academic Building S2

Figure 4.7: Average MAE of 4 buildings through 12 predicted time-steps

sequence-to-sequence modeling aspect of these models.

Table 4.2 shows RMSE results for hour 1 and the average over the 12 predicted hours for all buildings. We observe from the table that for all the buildings, GCRF and LSTM outperform the baselines. The overall performance improvement of GCRF over ARIMA and linear regression is in the range of 14% to 65%, while the gains of LSTM over ARIMA and linear regression is in the range of 7% to

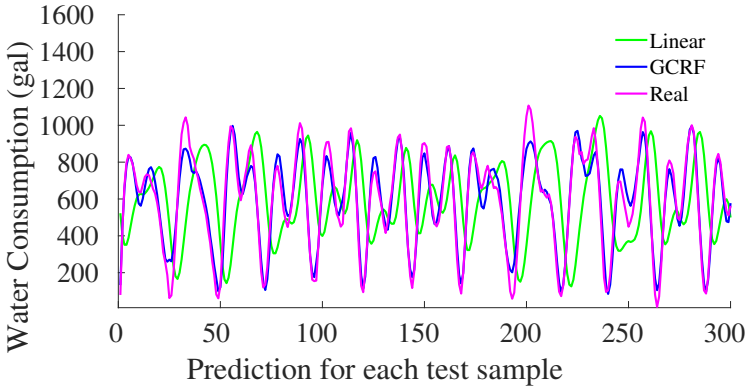
62%. We also see that for most buildings GCRF performs better than LSTM. We believe that the sparse nature of the L_1-regularized GCRF model helps in learning the dependencies that positively affect the prediction performance, while excluding those that do not matter. This helps in yielding a model that is better suited to the data.

4.4.2 MAE

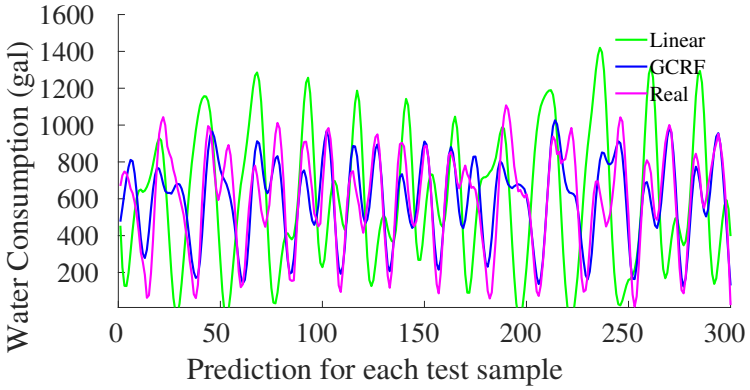
In this subsection, we discuss the MAE results for all models. Figure 4.7 shows the performance of the models for one building in each category. In comparison, Table 4.3 shows the 1 hour and the average (taken over predictions for the next 12 hours) MAE results. We observe that the GCRF and LSTM models outperform the baselines for all buildings with respect to MAE. The performance improvement of GCRF over ARIMA and linear regression is in the range of 20% to 67%, while improvement of LSTM over ARIMA and linear regression is in the range of 12% to 66% with respect to the average MAE. Once again, we see that for most buildings GCRF achieves a better performance than LSTM. The performance improvement of GCRF over LSTM is approximately 10%.

4.4.3 Qualitative Results

In this subsection, we compare the qualitative prediction performance of the GCRF and LSTM models with the baselines to help the reader appreciate the superior performance of our models. Figures 4.8a and 4.8b show the 1 hour and 12 hour predictions for GCRF and linear regression, while Figures 4.9a and 4.9b show the 1 hour and 12 hour predictions for LSTM and ARIMA for residence hall RA. For the 1 hour prediction, we observe that as linear regression tends to closely follow the actual values in the previous time step, it provides poor prediction performance as the recent past may not mirror the future. In comparison, GCRF generates smoothened predictions as it is trained on entire input sequences and thus provides superior performance. Additionally, we observe from Figure 4.8b that the 12 hour prediction for linear regression is notably worse than its 1 hour prediction. In comparison, as GCRF takes entire sequences into account and captures the underlying variations in the data, its 12 hour prediction performance does not deteriorate significantly. Similar to GCRF, as LSTM is also a sequence-to-sequence model and elegantly capture the dependencies in the data, its prediction performance does not decrease with larger time steps (Figures 4.9a and 4.9b).



(a) Residence Hall RA: Hour 1



(b) Residence Hall RA: Hour 12

Figure 4.8: Qualitative Results: GCRF vs Linear Regression

4.4.4 Adding Temporal Features

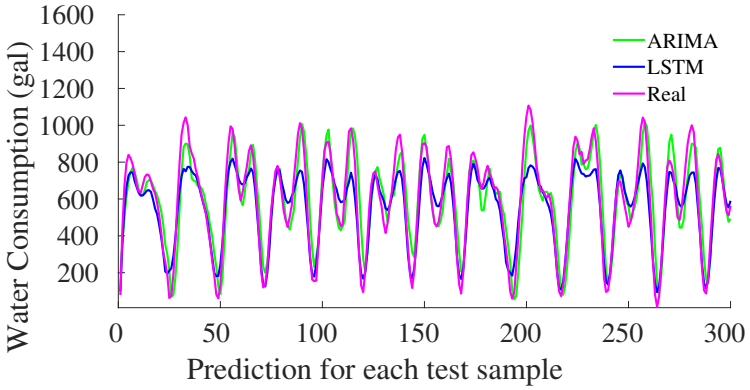
In the experimental results reported so far, we have only used the previous water consumption data to predict future water consumption. In this subsection, we investigate the performance improvement of augmenting our GCRF and LSTM models with temporal features. To this end, we add two features— i) day of the week and ii) hour of the day in our model. Day of the week takes values from 1 to 7, where 1 denotes Sunday. Hour of the day take values from 1 to 24, where 1 denotes the time period from 12 am to 1 am. Table 4.4 shows the average performance improvement over 12 predictions obtained by our augmented models over their respective baseline GCRF and LSTM models. We observe from the table that including the temporal features improves performance for most buildings for both GCRF and LSTM. The average performance improvement for GCRF and LSTM are 8.42% and 10.31%, respectively. The highest improvement is observed in academic buildings where the performance is enhanced by around 15% for building S3 in GCRF and 24% for building LH in LSTM.

Table 4.4: Percentage improvement after adding features

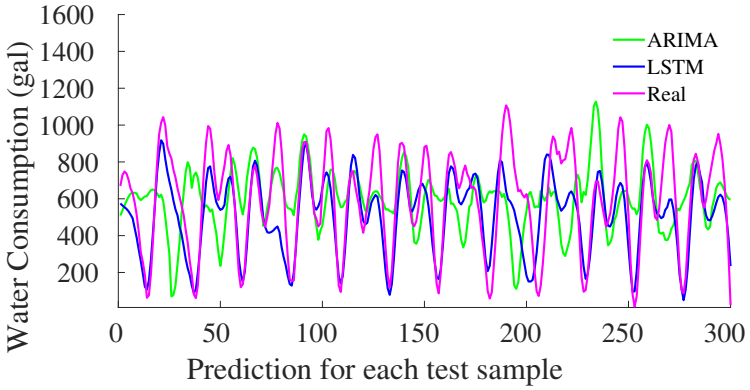
Building	**GCRF**	**LSTM**
BN	6.68%	10.49%
BR	8.36%	10.54%
C4	5.01%	-
DE	7.66%	6.67%
DG	7.6%	15.6%
JS	8.34%	7.51%
LH	-	24.34%
RA	6.43%	5.92%
S2	11.13%	6.57%
S3	14.58%	5.21%

Table 4.5: RMSE Varying Sequence Length

Building	**GCRF**			**LSTM**		
	12	**18**	**24**	**12**	**18**	**24**
BN	252.71	175.74	151.2	176.83	165.27	154.95
BR	134.87	101.33	88.28	95.81	80.69	84.89
C4	184.67	151.78	148.26	173.33	158.43	164.81
DE	272.67	208.73	184.95	220.61	217.21	201.96
DG	264.93	198.56	168.83	221.84	189.84	190.53
EB	79.56	72.18	70.5	80.91	80.37	75.46
FA	71.76	60.57	58.17	90.91	87.39	81.93
GE	163.67	142.03	138.81	226.45	148.29	145.61
JS	283.89	194	162.15	244.38	187.51	190.96
LH	135.44	118.01	116.77	126.84	138.97	122.53
RA	312.18	225.17	188.31	259.07	204.04	203.88
S2	28.77	24.11	23.26	31.65	28	26.01
S3	316.76	247.7	238.64	330.36	326.43	313.26
SN	130.51	125.12	124.43	135.75	133.03	134.13



(a) Residence Hall RA: Day 1



(b) Residence Hall RA: Day 12

Figure 4.9: Qualitative Results: LSTM vs ARIMA

4.4.5 Varying Sequence Length

In this subsection, we discuss the impact of varying sequence length and the rationale behind choosing 24 time steps as the input sequence length. Table 4.5 shows the average RMSE results for input sequence lengths 12, 18 and 24. We observe that for both models RMSE values are the worst for all buildings when the sequence length is 12. We also see that for most buildings having sequence length of 24 provides better performance than sequence length of 18. This is because a sequence length of 24 captures water consumption behavior for all hours of the day. Having sequence lengths greater than 24 does not significantly improve performance as longer sequences only reinforce previously learnt structure in the data.

4.4.6 Discussion on SWaP's practicality

The above experiments demonstrate that the GCRF-based **SWaP** overall outperforms the LSTM-based **SWaP**. Therefore, we recommend using the GCRF-based **SWaP** due to its superior prediction performance. Employing the GCRF-

based **SWaP** also provides the system with greater interpretability as GCRF is a probabilistic graphical model and it is easy to understand and appreciate which inputs/past outputs are instrumental in arriving at the predictions. These insights can help in understanding the inherent patterns in the data and explain the predictions, when necessary.

Additionally, in comparison to DL models, GCRF models require significantly less time (around 5 minutes when compared to few hours for deep learning models) and limited computational resources to train. This further means that in a deployed system, as new data becomes available, it is relative easy to re-train the model. Also, we observe that both the models perform well during test time on > 30 days of consecutive data without the need for re-training. Thus, it is only required to re-train both the models at comparatively infrequent intervals, aiding in practical deployment.

Another attractive aspect of **SWaP** is its low data and computational power requirement at test time. A well-trained **SWaP** system only requires 24 prior data points at test time to make strong predictions. Moreover, both GCRF and deep models are highly computationally efficient at test time, which means that it can generate the predictions quickly, a desired attribute in a practical system. These characteristics of **SWaP**, in particular the GCRF-based one, make it a useful system for managing water consumption. These qualities also make the system potentially extensible to other water management scenarios.

4.5 Conclusion

In this chapter, we investigated the hourly water consumption prediction problem using data collected from multiple buildings in a university campus. We designed **SWaP**, a **S**mart **Wa**ter **P**rediction system to accurately predict future hourly water consumption based on historical data. To enable **SWaP** make good predictions, we designed discriminative probabilistic graphical and deep learning models, in particular sparse GCRF and LSTM based deep models that successfully capture dependencies in the water consumption data. Our experimental evaluation shows that **SWaP** achieves superior prediction performance for all buildings, when compared to linear regression and ARIMA baselines in terms of RMSE and MAE. Additionally, we observed that a GCRF-based model provides better performance than an LSTM based deep learning model. Therefore, we recommend adopting the computationally efficient and interpretable GCRF-based **SWaP**, which makes our model practically attractive.

5 Resolution Time Prediction of Emergency Events

A number of emergency incidents (e.g. fire, building collapse, etc.) are reported on a regular basis in cities around the world. It is important that city officials are able to allocate sufficient resources to ensure public safety, smooth functioning of cities and address such incidents in a timely manner. To engage the public in coordinating these emergency response services smoothly, governments and city officials have made such data openly available to everyone. By adopting a data-driven approach, cities can efficiently allocate resources, plan prudently, and thus minimize the loss to human life and property and improve resolution time.

One of the major challenges in this regard is estimating the resolution time for emergency events in the future. While emergencies are unpredictable by nature and often happen unexpectedly, it is possible to leverage past resolution time data to predict the resolution time of future events. This is because the nature of the event (e.g., fire), the extent of damage, and the number of personnel and equipment available on site are keys factors that dictate the total time needed to address the issue. For example, if multiple emergencies occur in a colocated manner, then it is likely that the time needed to address each of these issues will be higher than usual because of the division of resources. Therefore, if a data-driven analysis suggests that resolution time for future events will be higher than a desired value for a particular incident type, then this analysis can provide insights into budget spending, personnel hiring, and resource allocation.

Hence, we design **DeepER**, a deep learning based emergency resolution time prediction system that predicts the future resolution time of incidents based on historical data. We consider three important emergency incident types, namely, *Fire*, *Law* and *Structural*. We model emergency resolution time prediction as a time series prediction problem. At the core of DeepER there is a sequence-to-sequence encoder-decoder neural network architecture. Both the encoder and the decoder in DeepER are Recurrent Neural Networks (RNNs) and the basic cell is an LSTM cell. The encoder receives the previous resolution times as input and encodes them into a hidden context vector. This hidden vector is given as an input to the decoder, which generates future resolution times.

To evaluate the performance of DeepER, we perform extensive experiments on

the publicly available NYC Emergency Response Incidents dataset [64]. We use the data for a period of approximately eight years for the three incident types. This dataset is challenging from the perspective of time series analysis and prediction because emergency events by nature occur at random times (i.e., lack periodicity), have limited correlation to each other and may not follow seasonal trends. Because of this reason, we design DeepER as a sequence-to-sequence model so that it can unearth the dependencies among the data points in the sequence even though the time period between two consecutive events in the sequence is varying. DeepER leverages these hidden patterns in data to make superior predictions.

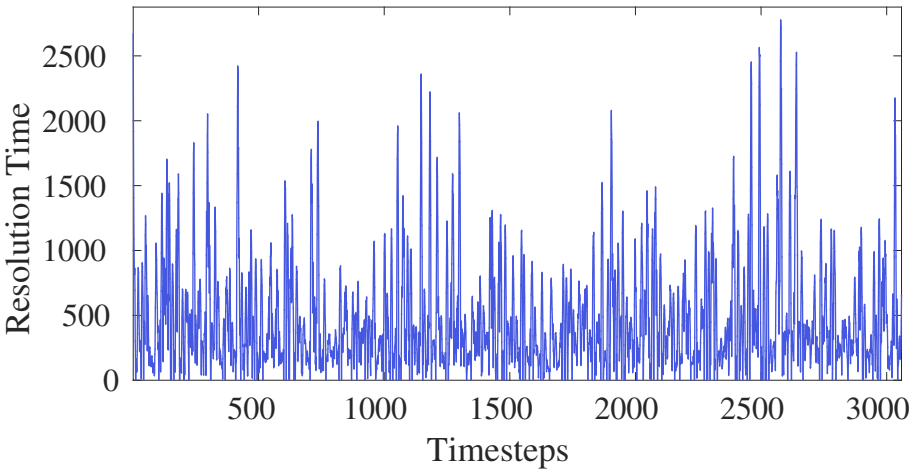
We compare the performance of DeepER with two widely used baselines— Linear Regression and Auto Regressive Integrated Moving Average (ARIMA). We use two metrics to evaluate the models— Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). DeepER achieves an average performance improvement of 3% and 16% with respect to RMSE and 10% and 27% with respect to MAE over ARIMA and Linear Regression, respectively. Our results demonstrate that DeepER is a practically viable system that provides superior prediction performance and can be used to aid city planning and management. We conclude the chapter with a discussion of some of the insights we obtain while conducting this investigation.

The rest of the chapter is organized as follows. In section 5.1, we present related work. We present the dataset and discuss the problem investigated in section 5.2. We then describe the DeepER system in section 5.3 and the implementation details in section 5.4. We present experimental results in section 5.5 and discuss some of our insights from this work in Section 5.6.

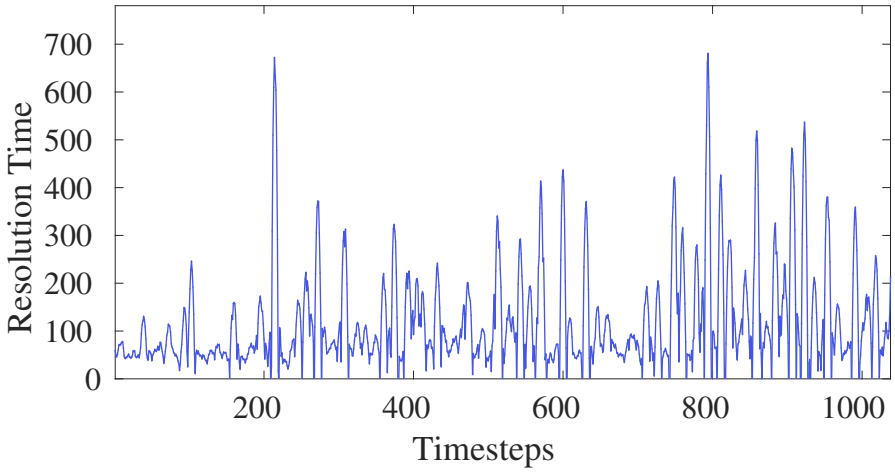
5.1 Related Work

With the growth and development of smart cities and cyber-physical systems, a variety of machine learning approaches have been adopted to address different problems in these domains [65]–[70]. In this section, we first present work related to assisting the operations of emergency and non-emergency services and then discuss prior research related to smart cities.

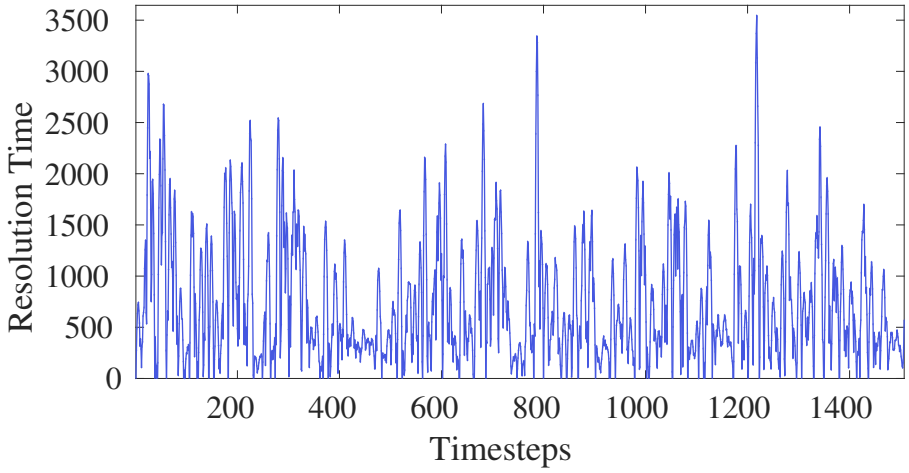
In the recent years, a number of research papers have adopted data-driven approaches to aid the functioning of emergency and non-emergency services in cities. For example, DeFazio et al. use Gaussian Conditional Random Fields (GCRFs) to predict response times of non-emergency 311 calls in NYC [40]. The authors in [71] adopt a rolling forecast model to predict the number of emergency calls based on the number of 911 calls in NYC. Similarly, the authors analyze NYC non-emergency call requests and present a Random Forest model to predict



(a) Fire



(b) Law

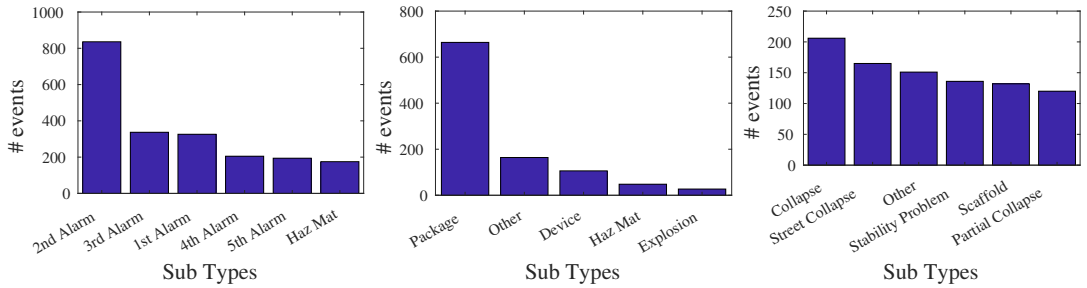


(c) Structural

Figure 5.1: Trends in datasets

the number of requests [72].

In [73], authors analyze the intra-region temporal correlation and the inter-region spatial correlation of data collected from NYC and build a framework to predict the number of crimes for certain regions. Similarly, the authors propose a



(a) Fire (b) Law (c) Structural

Figure 5.2: Incident Types

neural network based continuous conditional random field model for fine-grained crime prediction in Chicago and NYC [74]. The potential of DL models for a variety of time series prediction tasks has also been explored recently. For example, deep learning models have been adopted for emergency event prediction in [75]. Similarly, the authors use LSTM based deep models for gas consumption and occupancy detection using WiFi beacons in [76] and [77], respectively. Recurrent Neural Network (RNN) based encoder-decoder models similar to the one designed in this chapter have also been used for prediction problems in a variety of different domains. For example, such models have been used for water consumption, gym center occupancy, wireless channel quality and air pollution prediction **SWaP**, [78]–[80].

In contrast to existing work, we design DeepER, a deep learning model to predict the resolution time of emergency services and validate the efficacy of the model using the emergency incidents response data from NYC collected over a period of approximately eight years.

5.2 Data

We use the emergency response incidents data from NYC Open Data provided by the Office of Emergency Management [64]. We use around 8 years of data starting May 2011 to December 2019. The dataset consists of 13 incident types with 7 attributes each. We use three attributes from the dataset — Incident type, Creation date, and Close date. In this study, we focus on three of the most important and frequent emergency types — *Fire*, *Law*, and *Structural*. We calculate the *resolution time* for each incident by subtracting the creation date from the close date and converting it to minutes. Figure 5.1 shows the resolution times for these incidents. We observe that the time required to resolve events related to *Law* is the least followed by *Fire* and *Structural*, respectively. We also observe from Figure 5.1 that there is significant variation in resolution time for events belonging to the same incident type.

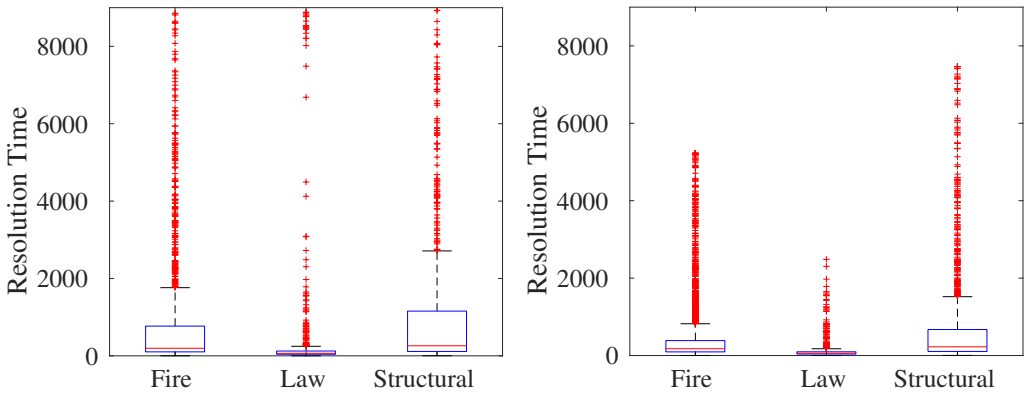
Additionally, we observe from the data that each of these three incident types have multiple subtypes, which we extract from the type description. *Fire*, *Law*, and *Structural* have 52, 29, and 73 subtypes, respectively. Figure 5.2 shows the incident subtypes that contribute the most events for each of the most general incident types. We observe that *2nd alarm*, *Suspicious Package (denoted as Package)*, and *Collapse* are the subtypes that account for the highest number of events in *Fire*, *Law*, and *Structural*, respectively.

5.2.1 Preprocessing

As is the case with most data-driven solutions to real-world problems, the first step involves pre-processing the data to identify missing values and outliers. We observe that the dataset contains non-trivial number of missing values for events. A missing value is encountered when an event does not have a valid close date. In the entire dataset, we observe that *Fire*, *Law*, and *Structural* have 32%, 12%, and 23% missing values, respectively. For each incident type, we replace these missing points by sampling from the actual distribution of the remaining points. To determine the actual distribution for each incident type, we fit the data to more than 80 different distributions. We perform the Kolmogorov-Smirnov goodness of fit test (KS test) and use the *p-value* of the KS test to pick the best distribution for the dataset under consideration.

We also observe that the dataset contains some outliers—values that are significantly different from the rest of the data points. By studying the values, we believe that such values might be the result of manually closing some unfinished entries at a later date. For example, we observe some extreme outliers in the dataset (greater than 100,000 minutes). We identify outliers as those points whose resolution time is greater than the quantile 90 of that incident type. For *Fire*, *Law*, and *Structural*, we observe that there are 7%, 9%, and 8% outliers, respectively. Figure 5.3 shows the distribution of the three different incident types before and after preprocessing. We observe from the figure that the raw dataset has a large number of outliers. We once again replace these outliers by sampling from the distribution of the valid data points. In comparison to Figure 5.3a, we observe that Figure 5.3b presents a significantly refined distribution.

We observe some other interesting issues in the dataset. We observe that for *Fire* and *Structural* most of outliers are located in the first few years of the dataset. In comparison, most of the missing points are located during the last few years. Additionally, for *Fire* and *Structural* we observe larger resolution times during the initial years than the last few years. However, the opposite is true for *Law*, where the resolution times during the initial years is lower than the resolution times in the later years.



(a) Before preprocessing (b) After preprocessing

Figure 5.3: Datasets before and after preprocessing

5.3 Model

In this section, we first discuss the future resolution time prediction problem studied in this chapter and then describe DeepER, a deep learning based system that predicts future resolution time based on past data.

5.3.1 Problem Statement

Our aim is to design a system that accurately predicts future resolution times of incidents from historical data. For this purpose, we cast the problem as a time series prediction problem. We consider a sequence of n events with resolution times X = x_1 , x_2 ..... x_n, and predict the resolution time of the next k events Y= y_1, y_2, ... y_k. What makes this problem challenging and different from classic time series prediction problems is that though these events occur chronologically, the actual time elapsed between two consecutive events varies. This is because each event corresponds to an emergency (i.e., unplanned) and thus the time when it occurs is completely random. Hence, in some cases one may have considerable time between two consecutive events, whereas in other cases multiple events can occur in a short duration of time. Therefore, our goal is to design a flexible model that examines the resolution time of a sequence of prior events and predicts the resolution time of future events and does not depend on the actual time frame in which the events occurred.

5.3.2 DeepER System Details

In this subsection, we describe DeepER, a sequence-to-sequence based encoder-decoder model that considers the resolution times of a sequence of prior events to predict the resolution time of future events. Figure 5.4 provides an overview of the DeepER system. DeepER consists of a data preprocessing block that splits

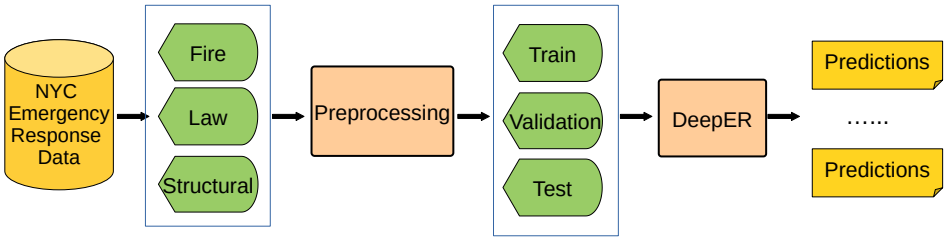


Figure 5.4: DeepER System Overview

the data by incident types and prepares the training, validation, and test datasets (details in Section 5.4). The preprocessing block also replaces the outlier and missing values according to the steps outlined in Section 5.2.1. The system then presents the data sequences as input to the deep learning model that uses them to generate the predictions.

Sequence-to-Sequence Models

Before delving into the system details, we discuss the appropriateness of sequence-to-sequence models for the emergency resolution time prediction problem studied here. In comparison to classic statistical regression models, sequence-to-sequence models are better suited for this problem as they map entire input sequences to output sequences and do not just focus on capturing simple trends in the data. Additionally, as the points in the dataset are not equally spaced in time and the events lack a seasonal and periodic behavioral pattern, we design deep learning based sequence-to-sequence models because such models are capable of learning the underlying dependencies and correlations in the data using an interconnected neural network architecture during the training phase and are especially useful when the dependencies are not apparent and cannot be easily defined. The trained model leverages this knowledge to make accurate predictions at test time by considering the current input sequence.

Encoder-decoder based RNN Model

DeepER consists of two components, an encoder and a decoder as explained in Figure 2.2. Both the encoder and the decoder comprise of recurrent neural networks (RNN). An RNN is a network of neural nodes that are arranged in layers. Internally, the RNN has a hidden state h_t that is updated at each time step t using the input x_t and the previous hidden state h_{t-1}. At each time step t, the hidden state of the RNN is given by,

$$h_t = \phi(h_{t-1}, x_t) \tag{5.1}$$

Sequence Length	Learning Rate	Units in Hidden Layer
10-3	0.01	10
10-3	0.001	50
10-3	0.0001	100
15-5	0.01	10
15-5	0.001	50
15-5	0.0001	100

Table 5.1: Hypararameter combinations for experiments

where, ϕ is any non-linear activation function and $1 \leq t \leq n$.

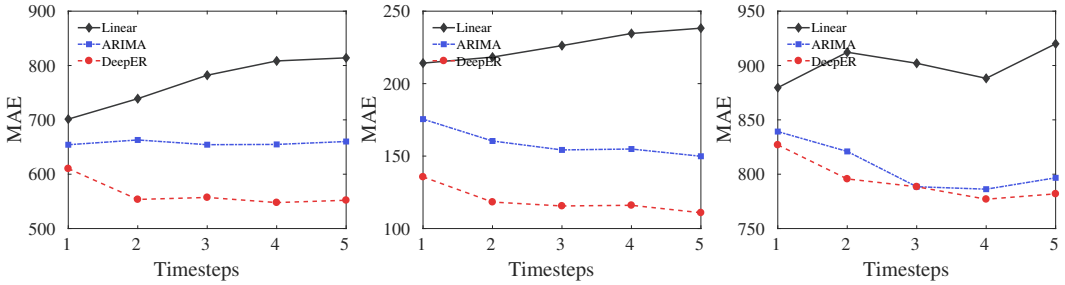
5.4 Implementation Details

In this section, we discuss implementation details regarding training, validation, and testing as well as important design decisions (e.g., hyper-parameter selection).

From our discussion in the Section 5.2.1, we observe that missing points and outliers are not spread uniformly throughout the duration of the dataset. Additionally, the entire duration of the dataset is approximately nine years and therefore, we observe gradual changes in the average resolution times of events for the same incident type. We attribute these variations to possible changes adopted by the different emergency management agencies. These issues inherent to the dataset necessitate some important design decisions. As the underlying characteristics of the data change over time, if we adopt a simple approach and split the data chronologically into training and test, then we will end up training solely on the data for the initial few years and testing on the last few years. This is unlikely to provide good performance because the distribution of the test data sequences are different from the distribution of the training sequences. Hence, we adopt a more careful approach where we ensure representation of data from each year in training, validation, and test datasets.

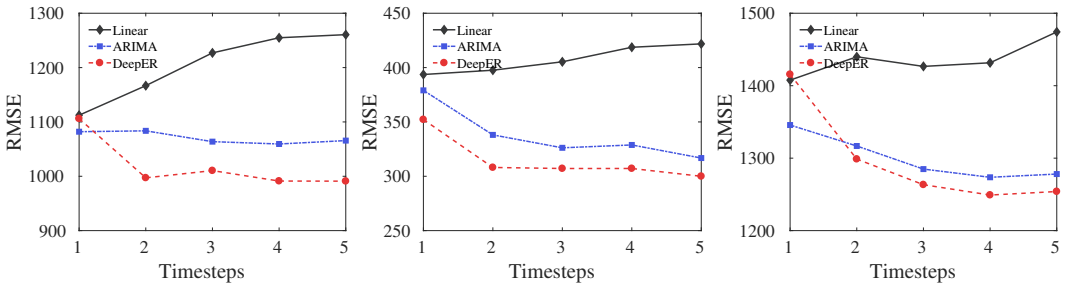
To do so, we divide the entire dataset into eight periods: seven periods of one year each and one period of approximately one and half year, approximately. We split each of these years in training, validation, and test sets following the usual percentages of 50%, 25%, and 25%, respectively. This ensures that the training, validation, and test sets contains data from all the years. Additionally, to remove any form of seasonal dependencies that may exist in the dataset, we permute the split order of training, validation and test within each year. Such permutation ensures that the training, validation, and test data contain samples from all months of the year; in the absence of such reordering the training data

will be confined to primarily the first few months, the validation confined to the middle months, and test containing data from the last few months of each year. In addition to helping in achieving good prediction performance across the entire duration of the dataset across the years, this important pre-processing step also makes our model more readily extensible to real-world deployment as the model is trained on the different variations that may be present in the data.



(a) Fire (b) Law (c) Structural

Figure 5.5: MAE Results for the 15-5 setting



(a) Fire (b) Law (c) Structural

Figure 5.6: RMSE Results for the 15-5 setting

5.4.1 Training, Validation, and Testing

We use Pytorch to implement the deep learning model. We train our models on a Linux machine with 8-core Intel i7 processor and 64 GB RAM. We use a sliding window approach with a stride of 1 to transform the time series into instances of sequences of length n and prediction of length k. We use unguided training as the training methodology. In this approach, the previous predicted output is used by the decoder as an input to the next step of the decoder during both training and test. Unguided approach is likely to provide better results at test time because it allows greater exploration of the state space during training. The loss function used to guide the training is the mean squared error.

During the training phase, we experimented with various hyperparameters and then finally decided upon 6 hyperparameter combinations that are best suited for our dataset (Table 5.1). A sequence length of 10-3 in Table 5.1 means that the

model takes 10 points are input and predicts 3 points into the future. For each hyperparameter combination, we iterate over 100,000 epochs, saving the state of the model after every 5000 iterations. We then select the particular model combination that provides the best performance on the validation set.

For quantifying the best performance, we do not simply pick the model with the lowest loss. Such an approach usually works well when the data has overall less variation and exhibits seasonality. However, in our dataset, events occur at random times and are of varying intensity (as each event is an emergency), thus resulting in higher variation among the values. This makes our dataset challenging to predict, resulting in the model adopting a safe approach and predicting values close to the mean value. Therefore, in addition to the loss function, we also consider another heuristic, the magnitude of the standard deviation among the predictions on the validation set, to select the best model. This approach ensures that the trained model provides superior quantitative and qualitative performance. Additionally, testing on a validation set and selecting from the myriad of combinations ensures that we do not overfit the model to the training dataset.

5.5 Results

In this section, we compare the performance of DeepER with two baselines: i) Linear Regression and ii) Auto-Regressive Integrated Moving Average (ARIMA).

Linear Regression is a statistical model that fits the best straight line to the given data.

ARIMA is a statistical model that has three components — AR (autoregressive term), I (differencing term), and MA (moving average term), specified by the parameters p, d, and q, respectively. We use the *pmdarima* toolkit in python for our experiments, which picks the optimal combination of the parameters for the input data.

We use two well-known metrics for evaluation, Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). Equations 5.2 and 5.3 show how they are calculated, where y_{ij} is the i^{th} test sample for j^{th} time step, $\hat{y}_{ij}$ is the predicted value of y_{ij}, and h is the total number of test samples.

$$RMSE_j = \sqrt{\frac{\sum_{i=1}^{h} (\hat{y}_{ij} - y_{ij})^2}{h}} \tag{5.2}$$

$$MAE_j = \frac{\sum_{i=1}^{h} |\hat{y}_{ij} - y_{ij}|}{h} \tag{5.3}$$

5.5.1 RMSE and MAE

In this section, we discuss the RMSE and MAE results for all the models. Table 5.2 shows the average RMSE and MAE results for sequence lengths 10-3 and 15-5, where the average performance is calculated over 3 and 5 time steps, respectively. Recall that a sequence length 10-3 means that the model takes 10 events as input and predicts 3 events as output into the future. We see that DeepER outperforms the baselines for all incident types on average for the 15-5 setting. For the 10-3 setting, DeepER outperforms both baselines for *Fire* and *Law*. However, for *Structural*, it outperforms Linear but not ARIMA. Therefore, from Table 5.2, we observe that DeepER provides overall better performance for the 15-5 setting.

Incident Type	**Model**	**10-3**		**15-5**	
		MAE	**RMSE**	**MAE**	**RMSE**
Fire	Linear	786	1256	769	1204
	Arima	660	1087	657	1071
	DeepER	**584**	**1069**	**564**	**1019**
Law	Linear	186	381	226	407
	Arima	133	323	159	338
	DeepER	**116**	**309**	**119**	**315**
Structural	Linear	965	1504	900	1436
	Arima	**818**	**1299**	806	1300
	DeepER	839	1307	**794**	**1296**

Table 5.2: Average MAE and RMSE

Figures 5.5 and 5.6 shows the MAE and RMSE results for the three incident types for the 15-5 sequence setting as it provides the best predicitons. We observe that DeepER significantly outperforms the baselines with respect to both MAE and RMSE. Interestingly, from the figures, we observe that DeepER is able to better predict the resolution time of events further into the future. This is in contrast to most time series prediction problems where the prediction performance deteriorates as the model predicts further into the future. The primary reason behind this behavior is that the data points in our dataset correspond to emergency events and hence lack seasonality, strong correlation, and trends. DeepER is still able to generate better predictions for our challenging problem than the baselines because of its sequence-to-sequence behavior that maps entire input sequences to output sequences and ability to glean complex underlying dependencies in the data that are not apparent.

5.6 Discussion

In the previous section, we demonstrated that DeepER provides superior prediction performance than the baseline models. In this section, we discuss some additional learnings from our exploration of this dataset.

5.6.1 Qualitative Results

We next discuss the qualitative prediction performance of DeepER as well as the baselines. Figure 5.7 shows the 1-step prediction performance for DeepER, ARIMA, and linear regression for *Fire*. From the figure, we observe that all models struggle to predict the values accurately. From our experience of working with similar models in the past **SWaP**, [78], [79], we have observed that sequence-to-sequence models are generally able to make really superior predictions. This does not appear to be the case always for this prediction task primarily because of the challenging non-periodic and non-seasonal nature of the emergency events dataset.

We observe from Figure 5.7 that the resolution times for some events is significantly higher in comparison to majority of the points. Because of this pattern, any prediction model will find it difficult to accurately predict such high peaks. But, despite this challenging nature of the dataset, we observe that DeepER provides a significantly smoothened prediction performance in comparison to the baselines and accurately predicts the underlying pattern. If we overlook the peaks, we can see that the prediction performance of DeepER for the remaining data points is good.

In comparison, we observe that the next step predictions for both ARIMA and Linear Regression closely mirror the actual resolution time of the previous time step. This occurs because both these baselines only use the past trend to predict the future. This is the root cause behind their poor performance because the resolution time of the current request is significantly different from the previous one.

5.6.2 Further Insights into Data Preprocessing

As mentioned in Section 5.2.1, we ensure that the training, validation, and test data include sequences from all years of the dataset. While cross-validation is commonly used to establish the significance of the results, we design this well-crafted split of the dataset to render greater credibility to our results primarily because of the fairly limited number of data points. Additionally, as noted earlier, the dataset contains a non-trivial number of outliers and missing values. Table 5.3 shows the number and percentage of missing and outlier points in each of

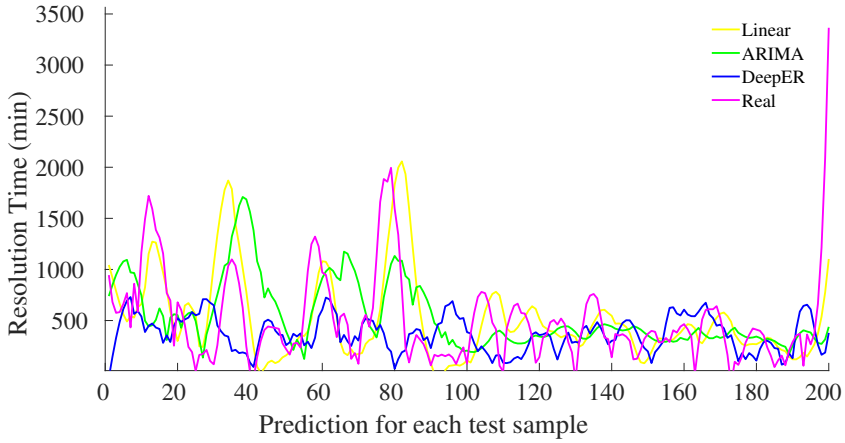


Figure 5.7: Fire: One step Real vs Predicted Performance

the training, validation and test sets if the data is split chronologically. This uneven distribution of the outlier and missing values further necessitates a carefully constructed split to ensure that training, validation, and test sets contain a more uniform distribution of such points. We note that while the dataset is relatively small in size, each event corresponds to an emergency and therefore, it is crucial to use all available data points and generate superior predictions because such predictions are critical to improving human safety.

Incident Type		**Training**	**Validation**	**Testing**
Fire	Total	1529	764	764
	Outliers	8%	6%	5%
	Missing	14%	47%	53%
Law	Total	519	260	260
	Outliers	8%	6%	13%
	Missing	2%	17%	27%
Structural	Total	754	377	377
	Outliers	8%	6%	13%
	Missing	2%	17%	27%

Table 5.3: Statistics of Outliers and Missing values in a Chronological Split

5.6.3 Limitations of Enriching DeepER

We observe from Section 5.2 that each incident type consists of multiple subtypes. We attempt to perform resolution time prediction at the subtype level, but realize that because this is an emergency events dataset, the number of data points is not sufficient for training, validation, and testing of deep learning models for each subtype separately. We also use these subtypes as features in DeepER, but observe that this enhanced model did not improve prediction performance. We believe that dearth of data at the subtype level is the primary reason behind it not contributing to DeepER's prediction performance.

5.6.4 Practicality of DeepER

With the increase in computational power over the last decade, deploying deep learning based systems to solve real-world problems is becoming relatively easy. As is the case with most deep learning models, DeepER requires some computational time for training. However, once trained, DeepER requires limited amount of time to generate predictions, a desired attribute in a practical system. Additionally, as more data becomes available, DeepER can be easily retrained thus enabling it to adapt to changing situations. We anticipate DeepER to be retrained at comparatively infrequent intervals (i.e., only when significant number of new emergency events have been resolved).

5.7 Conclusion

In this chapter, we presented DeepER, a deep learning based emergency resolution time prediction system that predicts future resolution times based on past data. We performed experiments on the NYC Emergency Response Incidents data provided by NYC Open Data. Missing values and outliers make the dataset challenging and thus necessitate effective preprocessing of the data before executing our experiments. We compared the performance of DeepER with ARIMA and Linear Regression using two metrics— Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). DeepER achieved an average performance improvement of 3% and 16% with respect to RMSE and 10% and 27% with respect to MAE over ARIMA and Linear Regression, respectively. We also draw upon important learnings and insights from the data, which can be utilized for designing deep learning models for data in the emergency response domain and other related domains where the data can lack an overt predictable trend. As part of our future work, we plan to extend this analysis to other cities so that it gives greater validity to our results. We want to also engage with city officials so that DeepER can be adopted to aid the planning and preparation of city emergency response systems.

6 Joint disaggregation and prediction for energy consumption

Designing machine learning models for smart energy consumption is an important research problem, having a tremendous impact on society. A crucial subproblem in facilitating smart energy consumption is being able to accurately disaggregate energy signals into their component appliance signals. This process is also known as energy disaggregation/non-intrusive load monitoring (NILM). This exercise provides residents with an accurate view and understanding of their energy consumption and can potentially help in reducing the peak energy consumption and facilitating efficient usage and conservation of energy. Recent advances in variational inference for deep learning have resulted in more expressive deep generative models such as variational autoencoders and variational recurrent neural networks that possess the ability to encode continuous latent variables. These latent variables provide the models with a powerful layer of abstraction that captures the variations in the input data and helps in generating the output data. These models map the input sequence into continuous latent variables using an inference network (referred to as an encoder), and then use the generative network (referred to as a decoder) to reconstruct the input sequence by sampling from the latent variables. Chung et al. [5] propose variational recurrent neural networks (VRNNs), which extend VAEs to model sequences by introducing high-level latent variables in RNNs. Deep generative models such as VRNNs and VAEs have achieved state-of-the-art performance in many sequence-to-sequence language tasks such as machine translation, paraphrase generation, and textual entailment, but have not been explored for the problem of energy disaggregation.

In this work, we present a novel deep generative architecture for disaggregation that leverages and adapts VRNNs to jointly disaggregate the total energy consumption into individual component appliance signals. Our proposed approach learns the abstraction of the aggregated energy consumption over latent variables at training time and then generates all the individual appliance signals jointly by sampling from the latent variables at test time. Hence, at test time our model only depends on the aggregated signal and the latent variable abstractions learned during training and does not depend on contextual information and appliance data from previous time steps, making it a meaningful model for energy disaggregation.

Specifically, we make the following contributions:

1. We present a novel deep generative architecture for performing sequence-to-many-sequence prediction (aggregated consumption to appliance consumptions) needed for energy disaggregation by leveraging and adapting variational recurrent neural networks (VRNNs). Our model generates continuous power consumption signals as opposed to state-of-the-art approaches that model consumption through discrete appliance states.

2. We model the structure among the different appliances in a household by jointly predicting each of them at the same time from the aggregated signal. We cast the different appliance energy signatures as a structured prediction problem, modeling the structure among the different appliance energy consumption signals over time, to effectively represent and reason about their dependence.

3. Our model achieves a performance improvement of *29*% and *41*% for the REDD and Dataport datasets, respectively, when compared to two recent state-of-the-art energy disaggregation approaches that use extensive additional past temporal and contextual information [81], [82]. Further, our model achieves a superior prediction performance on low power consuming appliances, which are harder to predict and are often ignored by most existing approaches.

4. Through qualitative analysis, we demonstrate that our models can achieve a superior disaggregation for both high and low energy consumption states and accurately discerns which appliance(s) contribute to the aggregated power consumption, thus providing a more useful and meaningful disaggregation model.

5. We demonstrate the extensibility of our model in predicting individual appliance consumption on previously unseen data by testing on a building that is left out while training. We observe that our model achieves a superior prediction performance on two buildings in REDD, thus making it potentially extensible to new datasets.

6.1 Related Work

Hart et al. [83] was the first to introduce the problem of energy disaggregation. Perhaps the most popular approach for energy disaggregation is using factorial hidden Markov models (FHMMs) [84], which generalize HMMs by using a distributed representation and its variants [85]–[88]. Shaloudegi et al. [82]

propose a scalable algorithm for this problem that extends FHMMs. Supervised machine learning models such as Support Vector Machines (SVMs) and k-Nearest Neighbors (k-NNs) and unsupervised models that use prior appliance models have also been applied to this problem [89]–[92]. Some other models have relied on other information apart from the aggregated consumption to model relationships with user's behavior and climate [81], [93], [94]. Tomkins et al. [81] propose a structured probabilistic framework for energy disaggregation. Recent advances in deep learning have spurred deep-learning based energy disaggregation models [95]–[101].

In this work, we propose a deep latent generative model based on VRNNs that combines the advantages of the modeling complexity of deep neural networks and the rich representational power of latent variables in probabilistic models such as FHMMs. Our model learns to predict all individual appliance signals *jointly* from the aggregated signal. We compare our approach to two recent state-of-the-art approaches for energy disaggregation: a) ADMM-RR, a scalable variant of FHMMs [82], and b) Tomkins et al.'s [81] joint probabilistic approach to energy disaggregation, and show that our approach achieves superior prediction performance.

6.2 Deep Latent Generative Models for Energy Disaggregation

In this section, we describe the energy disaggregation problem and the suitability of VRNNs for the same. Then, we present our deep latent generative energy disaggregation architecture.

6.2.1 Energy Disaggregation Problem

The problem of disaggregation is to calculate the energy consumption of individual component appliances given the total aggregated power consumption. Let $\mathbf{x} = (x_1, x_2, ..., x_T)$ be the aggregated energy consumption of a house over T time steps, where $x_t \in \mathbb{R}_+$. Let I be the number of appliances. The individual energy consumption of appliance i is denoted by $\mathbf{y^i} = (y_1^i, y_2^i, ..., y_T^i)$, where $y_t^i \in \mathbb{R}_+$. Consequently the aggregated energy signal at a given time can also be expressed as $x_t = \sum_{i=1}^{I} y_t^{(i)}$. We use y_t to denote the consumption time t for all the appliances: $y_t = \{y_t^1, y_t^2, ..., y_t^I\}$. *Our goal in this work is to develop a deep latent generative energy disaggregation framework that can learn to infer the continuous-valued appliances' consumption given the aggregated energy consumption.*

6.2.2 Variational Recurrent Neural Networks

Variational Recurrent Neural Networks (VRNNs) [5] are a recently developed deep neural network architecture that introduce latent variables and temporal dependencies between them in the different time steps in the RNN architecture. The core of a VRNN is a VAE [19], [102]. VAEs and VRNNs are variants of autoencoders (AE) and recurrent neural networks (RNNs) that encode latent variables and probabilistic transition functions. The principal difference between VAE and VRNN is that VRNN models the dependencies between latent variables across subsequent time steps, thus providing us with the ability to accurately abstract highly non-linear dynamics in sequential data. Since the prior distribution at timestep t is dependent on all the preceding inputs via the RNN hidden state h_{t-1}, the introduction of temporal structure in the prior distribution is expected to improve the representational power of the model. We first discuss the suitability of VRNNs for the energy disaggregation problem and then present our deep generative architecture.

Why are VRNNs suitable for the energy disaggregation problem?

As Chung et al. [5] note, VRNNs are best suited for modeling highly variable and highly structured (having a high signal-to-noise ratio) sequential data. Highly variable data exhibits high sudden variations that vanilla RNNs do not accurately represent. The deterministic nature of transition functions in RNNs limit their capability in modeling variability in the outputs. The presence of latent variables in VRNNs allows them to represent latent state spaces similar to models such as hidden Markov models (HMMs) and Kalman filters in a deep neural network architecture such as RNNs, thus achieving the combined benefits of both these classes of models.

Energy consumption signals are highly structured, i.e., they have a high signal to noise ratio; the variations in the data are due to signal itself rather than noise. Thus, the presence of structured output functions in VRNNs along with their ability to represent complex non-linear data make them ideal for modeling this domain.

The structured output functions present in VRNNs aid the joint prediction of disaggregated appliance signals from the aggregated consumption. Kelly et al. (2015) use RNNs for the energy disaggregation problem, but their model does not disaggregate all appliance signals at once. Instead, they train a separate model for each appliance. Due to the lack of probabilistic transitions between latent variables and structured output functions, this approach fails to capture the dependencies between the different appliance signals and thus lacks the ability to accurately

identify the contributing appliance signals in an aggregated signal.

In the energy disaggregation problem, usually power consumption is mapped to discrete appliance states [81], [82]. This, however, ignores the fine-grained variations in the signals. The deep structured construction of our model and the presence of latent variable abstractions and probabilistic transitions between them provide us with the ability to model the exact consumption of appliances as continuous values and detect fine-grained variations in the signals. Since we do not approximate signals into consumption states and model the exact continuous values, our approach requires minimal pre-processing and is able to model these minute variations.

6.2.3 VRNN-DIS-ALL: A Deep Generative Energy Disaggregation Framework

We bring out the modeling power of VRNNs by adapting them to disaggregate individual appliance signals jointly from the aggregated power consumption signal. In the following sections, we present the generative process, inference, and learning in our model, VRNN-DIS-ALL. We also highlight the adaptations to the original VRNN for the energy disaggregation problem.

Generation

The VRNN contains a VAE in each time step but the prior on the latent variable follows a distribution that is conditioned on the hidden state at time $t-1$, h_{t-1}. We augment the prior distribution to include both h_{t-1} and the aggregated consumption at time t, denoted by x_t. Hence, the random variable z_t follows the distribution:

$$z_t \sim N(\mu_{0,t}, diag(\sigma_{0,t}^2)) \tag{6.1}$$

where, $[\mu_{0,t}, \sigma_{0,t}]$ denote the parameters of the distribution $\phi_\tau^{prior}(h_{t-1}, x_t)$. While for the generation task described in Chung et al. [5], the prior distribution only depends on h_{t-1}, we adapt it to include the aggregated signal as we are interested in generating the disaggregated appliance signal from the aggregated signal. Next, y_t (disaggregated signal) is generated given z_t and h_{t-1} from the distribution:

$$y_t|z_t \sim N(\mu_{y,t}, diag(\sigma_{y,t}^2)) \tag{6.2}$$

where, $[\mu_{y,t}, \sigma_{y,t}] = \phi_\tau^{dec}(\phi_\tau^z(z_t), h_{t-1})$. Chung et al. [5] note that ϕ_τ^{prior} and ϕ_τ^{dec} can be any highly flexible functions and are essential for learning complex dependencies. In our models, ϕ_τ^{prior} and ϕ_τ^{dec} are neural networks with one-hidden layer

with standard activation functions. The hidden layer has a hyperbolic tangent ($tanh$) and the output layers for $\mu_{y,t}$ and $\sigma_{y,t}$ have linear and softplus activations, respectively. The RNN hidden state calculation is given by

$$h_t = f(\phi_\tau^x(x_t), \phi_\tau^y(y_t), \phi_\tau^z(z_t), h_{t-1}) \tag{6.3}$$

where, f is the transition function between hidden states. The feature extractors, ϕ_τ^x, $\phi_\tau^y(y_t)$, and ϕ_τ^z, can be any expressive function. We use a 1-hidden layer neural network for the same. The learning problem is to learn the prior distribution, $\phi_\tau^{prior}(h_{t-1}, x_t)$, to be as close as possible to the approximate posterior $\phi_\tau^{enc}(\phi_\tau^x(x_t), \phi_\tau^y(y_t), h_{t-1})$.

Inference

At training time, VRNN works as an encoder, learning the approximate posterior as a function of x_t, y_t and h_{t-1}.

$$z_t|x_t, y_t \sim N(\mu_{z,t}, diag(\sigma_{z,t}^2)) \tag{6.4}$$

where, $[\mu_{z,t}, \sigma_{z,t}]$ denote the parameters of the distribution $\phi_\tau^{enc}(\phi_\tau^x(x_t), \phi_\tau^y(y_t), h_{t-1})$. In addition to the feature extractors from x_t and z_t, we also include $\phi_\tau^y(y_t)$ that extracts the features of the disaggregated signal y_t at training time. Inference at training time is done by sampling z_t from this approximated posterior distribution. At test time, z_t is sampled from the learned prior distribution that is learned during training. This difference in the z_t can be appreciated in Figure 6.1 where we can see how the distribution ϕ_τ^{prior} replaces the encoder distribution ϕ_τ^{enc}.

Learning

Learning is performed by minimizing the sum of two components: distance between the posterior and the prior distribution and the log-likelihood of the output. In the first term in Equation 6.5, we minimize the Kullback-Leibler divergence distance (KL divergence) between the approximate posterior in Equation 6.4 (denoted by q in Equation 6.5) and the prior distribution (denoted by p in Equation 6.5), where z_t depends only on aggregated signal ($x \leq t$) and the latent variable states at previous time steps ($z < t$). The second term captures the negative log-likelihood of the output distribution from which we sample y_t.

$$KL(q(z_t|x \leq t, y \leq t, z < t)||p(z_t|x < t, z < t)) + logp(y_t|z \leq t, x < t) \tag{6.5}$$

Training

At training time, our goal is to learn an approximate function that is very similar to the conditional distribution $p(z|y)$ by minimizing the KL divergence between the prior distribution (ϕ_τ^{prior}) and the approximate posterior or the encoder distribution (ϕ_τ^{enc}) (Figure 6.1a). We follow a curriculum learning strategy as proposed by Bengio et al. [103] that involves gradually migrating during training from considering the ground truth to the output predicted by the model in the previous step in order to bridge the gap in inference between training and testing. The scheduled sampling algorithm used by this learning strategy will decide at training time whether to sample from the ground truth (y_t) or from the predictions generated by the model ($\hat{y}_t$). In our models, we use an inverse sigmoid decay. It is defined as: $p_i = k/(k + exp(i/k))$ where, p_i is the sampling probability and $k >= 1$ gives the speed of convergence. This probability is calculated at each time step.

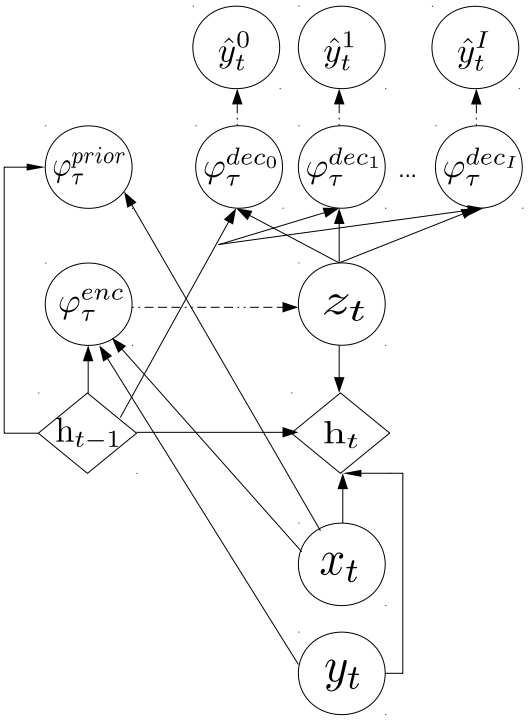
Testing

At test time, we only input the aggregated energy consumption information x_t. We no longer use the encoder distribution but the parameters of the learned prior distribution to sample the latent variables z_t, i.e., $z_t \sim N(\mu_{0,t}, diag(\sigma_{0,t}^2))$. Then, we calculate the parameters of the distribution of each appliance using $y_t^i|z_t \sim N(\mu_{y,t}^i, diag({\sigma_{y,t}^i}^2))$, where $[\mu_{y,t}^i, \sigma_{y,t}^i]$ is now calculated from the learned prior distribution. We calculate the next recurrent hidden layer h_t as a function of feature extractor neural networks for x_t, $\hat{y}_t$, and z_t, and previous hidden state h_{t-1}, i.e., $f(\phi_\tau^x(x_t), \phi_\tau^z(z_t), \phi_\tau^y(\hat{y}_t), h_{t-1})$. Note that here we use the predicted $\hat{y}$ instead of the actual y (Figure 6.1b).

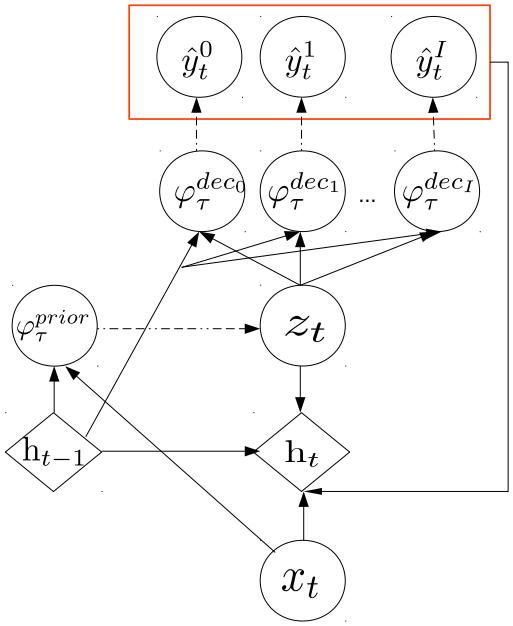
Implementation Details

We develop our model[1] on the original VRNN implementation [5] in Theano. Figure 6.2 captures the model architecture. It shows the different hidden layers, number of nodes in the hidden layers, identifies the components corresponding to the hidden layer, and captures the interactions between them for one iteration of training from time $t-1$ to t. The name of each component and the activation function applied to nodes in that hidden layer is mentioned at the top and the number of nodes is indicated in the bottom of each hidden layer. The architecture shows the distribution from where z_t will be sampled at training time (from the encoder, marked in green) and at inference/test time (from the prior, marked

[1] https://bitbucket.org/gissemari/disaggregation-vrnn



(a) VRNN-DIS-ALL at training



(b) VRNN-DIS-ALL at test

Figure 6.1: Graphical illustrations of VRNN-DIS-ALL training to reconstruct disaggregated appliance signals from the aggregated and disaggregated signals and as a generative model of disaggregated appliance signals from only the aggregated signal at test time.

in red). The weight matrices of all layers are randomly initialized using a uniform distribution. The LSTM-cell diagonal matrix that captures the interaction between the recurrent states h_{t-1} and h_t is initialized randomly from a normal

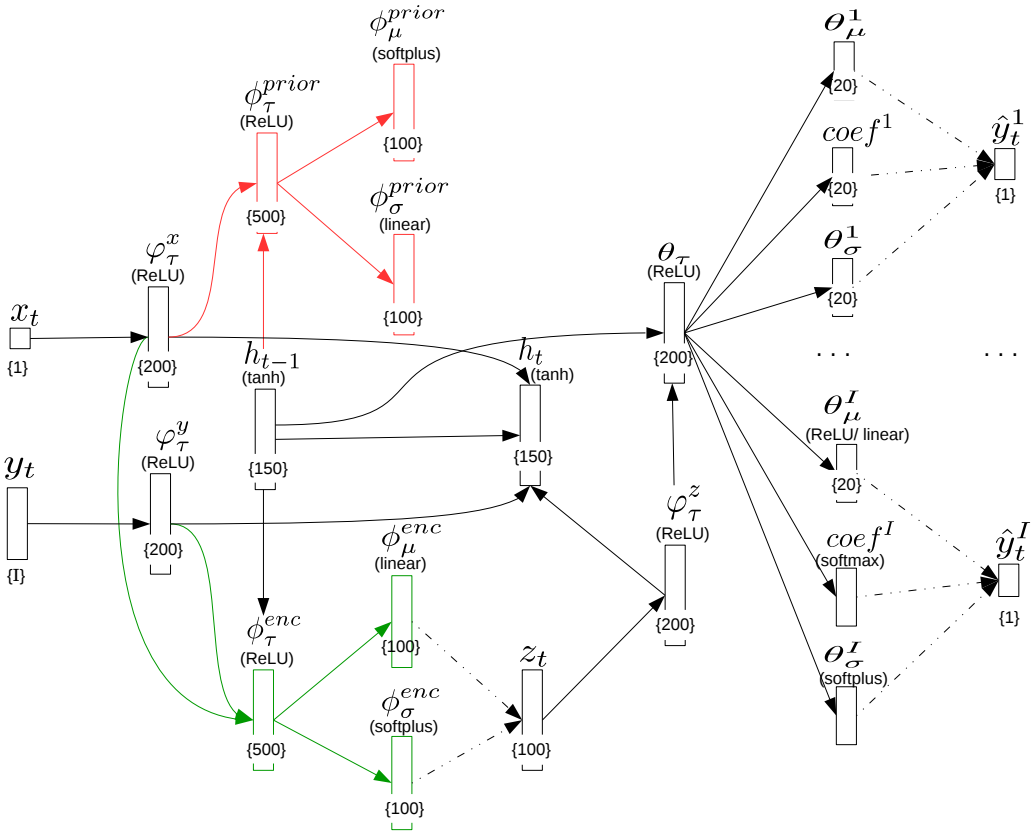


Figure 6.2: Architecture of the VRNN-DIS-ALL model. Solid lines represent fully connected layers and dashed lines represent the sampling process.

distribution ensuring its orthogonality. The initial hidden state of the recurrent neural network is initialized to 0.

We experiment with different activation functions for θ_μ and find that ReLU activation function for θ_μ works better for some buildings while for others the linear activation function works better. For θ_σ and $coef^1$, we apply a *softmax* and *softplus* activation functions, respectively. These parameters are calculated for each appliance, so the final layer has as many Gaussian mixture models (GMMs) as appliances.

6.3 Experimental Evaluation

We conduct experiments to answer the following questions:

1. How well do our deep generative models perform in energy disaggregation?
2. Are our models able to effectively identify which appliance(s) are contributing to the aggregated consumption?

Model	**Disaggregation Representation**	**Temporal Dependencies**	**Context / Heuristics**
ADMM-RR (Shaloudegi et al. 2016)	Discrete states	Encoded	✓
INTERVAL (Tomkins et al. 2017)	Discrete states	Encoded	✓
INSTANCE (Tomkins et al. 2017)	Discrete states	Encoded	✓
CONTEXT (Tomkins et al. 2017)	Discrete states	Encoded	✓
vrnn-dis-all (our approach)	Continuous	Learned	✗

Table 6.1: A comparison table between our model and the state-of-the-art energy disaggregation approaches

In this section, we present results from our experimental evaluation to answer the above-mentioned questions on two well-known real-world energy disaggregation datasets. We demonstrate the efficacy of our models by comparing them with two recent state-of-the-art energy disaggregation approaches: a) ADMM-RR [82], and b) Interval, Instance, and +Context models from Tomkins et al. [81]. Table 6.1 gives a comparison of our approach with the state-of-the-art energy disaggregation models. Our approach uses a continuous value representation, does not explicitly encode any domain-specific variables or capture any dependencies among them, and does not require any additional contextual information (such as

temperature of the day, day of the month/year, user-specific contextual information). Our model automatically learns these dependencies from training data. In our experiments, we demonstrate that our models outperform ADMM-RR across most appliances in both the datasets and outperforms Tomkins et al.'s best model on one dataset and achieves comparable performance on another despite using no temporal, domain-related, or contextual information.

We present two metrics of evaluation for both the datasets: i) mean absolute error (MAE), and ii) percentage of total energy estimated by each appliance compared to percentage of total energy in the original data. The MAE is calculated by computing the absolute value of the difference between the predicted disaggregated appliance consumption ($\hat{y}_t$) and the actual consumption (y_t). Percentage of energy estimated is calculated by taking the ratio of predictions associated with the appliance to original aggregated signal. This percentage is compared with the actual percentage of energy consumption of the appliance in the aggregated energy consumption. We evaluate our percentage predictions in the following ways: i) first, we compare the actual percentage numbers between our predictions and the actual data, ii) second, we compute the percentage/range of error between the predicted and the actual by taking the ratio of the difference in the percentages with the actual percentage, and iii) third, we compare our deviation in percentages (percentage of error) to the deviation in percentages reported for the same building by Tomkins et al.

6.3.1 Datasets

We evaluate our model on two real-world energy datasets: i) Pecan Street Inc. Dataset (DataPort) [104], and ii) Reference Energy Disaggregation Dataset (REDD) [105]. These datasets have been used in several previous works [81], [82], [92].

DataPort

The Pecan Street dataset (DataPort) consists of energy consumption readings at 1-minute and 1-hour intervals. We evaluate on the finer-grained 1-minute readings. As there are missing values, we work on the same subset of buildings (2859, 3413, 6990, 7951, 8292) that Tomkins et al. [81] use in their work. We consider data for the following appliances: *air conditioner*, *furnace*, *refrigerator*, *dishwasher*, *kitchen outlet*, *dryer*, *microwave*, and *clothes washer*.

REDD

The REDD dataset contains data for 10 houses from the greater Boston area for approximately two months. We again consider the same five houses (houses 1, 2, 3, 4, and 6) that Tomkins et al. [81] and Makonin et al. [92] consider so that we can make a fair comparison. We also consider the same four appliances for houses 1, 2, and 3: *refrigerator*, *dishwasher*, *light*, *microwave*. For house 6 we exclude *microwave* as the data for that appliance is unavailable. We use the non-intrusive load monitoring toolkit [106] to get a sampling rate of every 6 or 60 seconds.

6.3.2 Data Preprocessing

To preprocess the data for our model, we first determine the minimum activation threshold for each appliance in each dataset. Then, we use a non-overlapping sliding window on the entire original time series data to construct sequences of fixed length from them. From these sequences, we filter the ones where at least one data point in the sequence is greater than the minimum threshold activation for each appliance. We treat each sequence as one data instance. We split the total number of instances into training, testing, and validation sets in the ratio 50%:25%:25%, respectively. We record the performance metrics in the validation set every ten epochs to detect and prevent overfitting.

We construct batches of instances (which we refer to as *mini-batch*) and train the model for many epochs for each mini-batch. This enables the model to see a smaller number of instances for a longer training period, enabling it to model the structural dependencies in the data. We use 5-30 mini-batches. We report the average scores from three different train-test-validation splits across both datasets. Note that our approach uses very minimal pre-processing and domain knowledge when compared to the existing state-of-the-art approaches.

6.3.3 Energy Disaggregation Results on DataPort

Table 6.2 shows the MAE of each appliance in each building in DataPort dataset. Our model is able to achieve low MAE for appliances that consume higher energy in average such as *clothes washer* and *air conditioner*. The first one shows a MAE of 1.5, 6, 8.5, and 2.5 in buildings 2859, 6990, 8292 and 3413, respectively and the *air conditioner* obtains a MAE of 9.5 for building 2859. In addition to that, appliances which consume less energy on average such as *dishwasher*, *kitchen appliance*, and *microwave* show an average MAE of less or equal than 15.5 among all buildings. It is interesting to note that Tomkins et al.'s prediction performance of appliance states for appliances that consume lesser power on average and are intermittent is lower as indicated by their lower values of precision, recall and F1

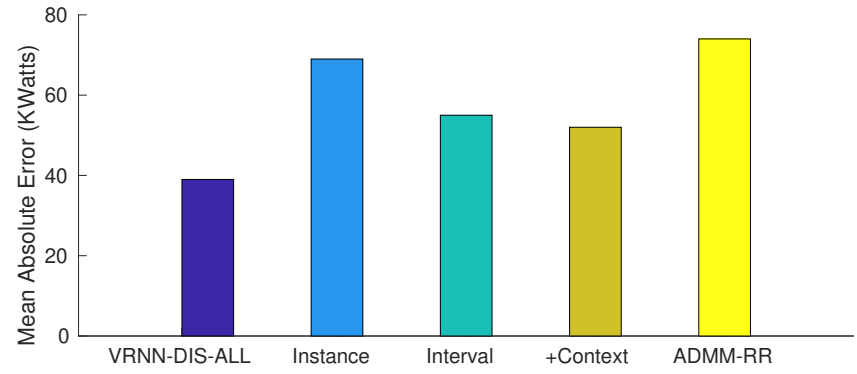
Appliance	Building					
	2859	**6990**	**7951**	**8292**	**3413**	**AVG/Appliance**
Air	9.50	199.50	152.50	98.50	64.00	104.80
Furnace	40.50	110.50	59.00	56.50	32.50	59.80
Refrigerator	32.50	70.00	77.50	60.00	71.50	62.30
Clothes washer	1.50	6.00	24.00	8.50	2.50	8.50
Dryer	4.00	52.00	33.00	78.50	35.50	40.50
Dish washer	1.00	8.00	14.50	25.00	9.50	11.60
Kitchenapp	1.00	3.00	14.00	17.50	1.00	7.30
Microwave	10 .00	12.50	40.00	9.00	6.00	15.50
AVG/Building	12.50	57.69	51.81	44.13	27.81	**38.79**

Table 6.2: VRNN-DIS-ALL results on DataPort showing the MAE for each appliance for the five buildings.

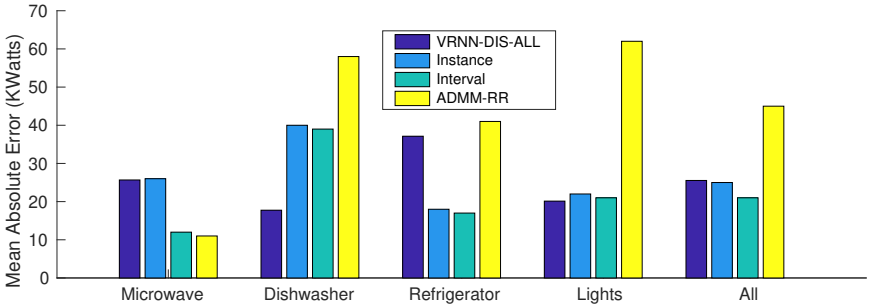
scores. Thus, our model is able to discern patterns of consumption in both kinds of appliances and hence perform a more accurate disaggregation.

Figure 6.3a shows the comparison of average MAE value across the buildings and appliances between our model VRNN-DIS-ALL and two existing state-of-the-art approaches. Since we consider the same set of buildings and appliances, we make a direct comparison to the results presented by Tomkins et al. [81]. We observe that our model achieves *29%* performance improvement in MAE over the +context model (Tomkins et al.'s best model) and *41%* improvement over ADMM-RR. It is important to note that our model achieves this performance improvement without any contextual information. The deep nature of the model and the presence of neural network feature extractors help in extracting complex features and learning structural dependencies among them. This eliminates the necessity to encode domain-specific information and their relationships as in existing probabilistic energy disaggregation approaches. Hence, our approach requires less manual effort and can scale easily to new datasets without the need for careful encoding of graphical structure among variables.

Figure 6.4 gives the percentage of total energy consumed by the appliance as predicted by our model compared with the actual percentage of total energy consumed by the appliance in the original data. We group the appliances into *air conditioner* (air), *furnace*, *refrigerator*, *dryer*, and *others*, to enable an easy comparison to Tomkins et al.'s percentage calculations. Comparing the predicted percentage of total energy with the actual for *air conditioner* across all buildings, we observe that our model predicts within 4% of the actual percentage of energy consumed by the appliance for 4 out of 5 buildings. Similarly, for *dryer*, our model's predictions lie within 11% for 4 out of 5 buildings, and for *furnace*, our model's predictions lie within 6% for 3 out of 5 buildings. Comparing the percent-



(a) MAE: DataPort



(b) MAE: REDD

Figure 6.3: MAE comparing our proposed model VRNN-DIS-ALL with existing state-of-the-art models (Interval, Instance, +Context, and ADMM-RR)

ages for building 3413 with the percentages reported by Tomkins et al., we observe that our model's percentage prediction for *air conditioner* deviates by only 3.6% from the actual percentage, while theirs deviates by 7.3%. Similarly, comparing the percentages for *furnace* we observe that ours deviates by 1.5% while theirs deviates by 10%. For the rest of the appliances, our model achieves comparable differences in percentages between the predicted and the actual values to their model.

6.3.4 Energy Disaggregation Results on REDD

Figure 6.3b gives the comparison for average MAE of each appliance across buildings between VRNN-DIS-ALL and existing state-of-the-art approaches. Here, we only compare against INTERVAL and INSTANCE models from Tomkins et al. as the +CONTEXT model cannot be used due to absence of contextual information in the dataset. We observe that our model achieves superior performance on *dishwasher* and *lights*, which are harder to predict due to their unpredictability. We get performance improvements of 69% and 68%, respectively, over ADMM-RR and 56% for dishwasher over Tomkins et al. For the other appliances: *microwave* and *refrigerator*, we achieve comparable performance to one of the existing approaches. Comparing our overall MAE averaged over all buildings

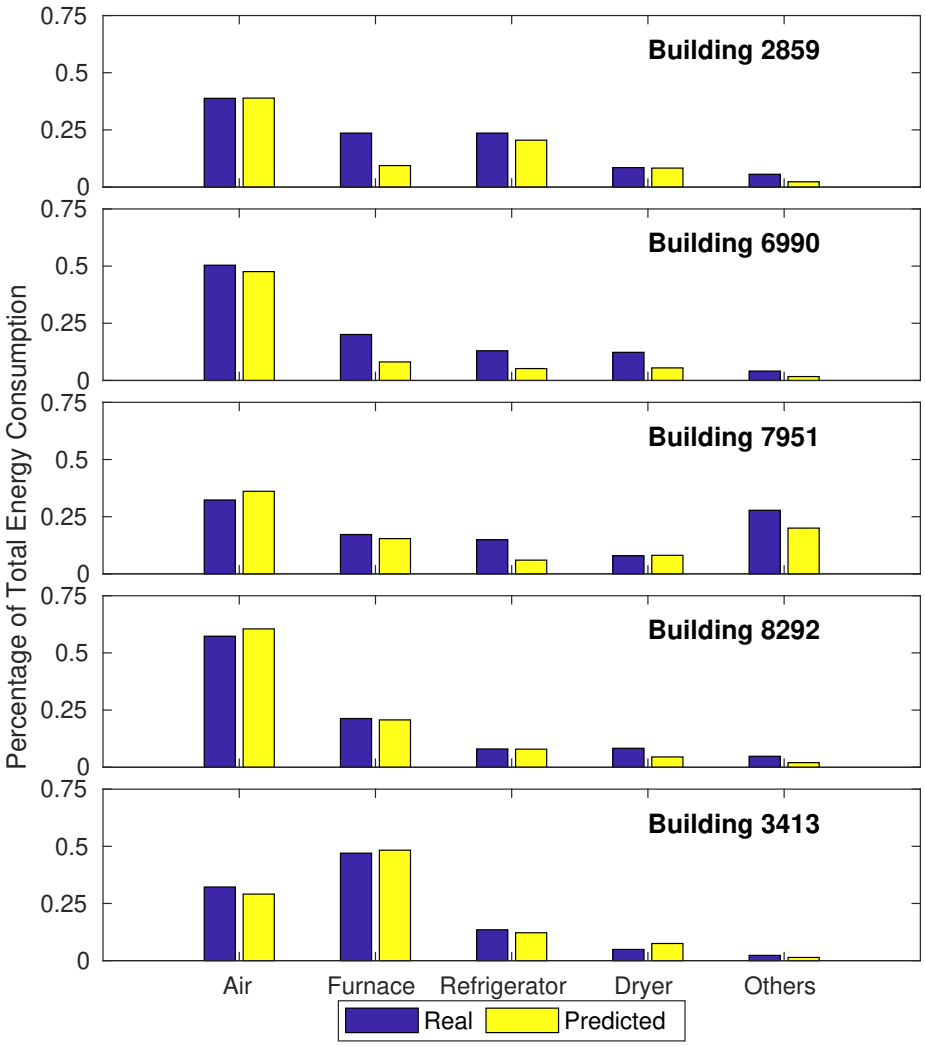


Figure 6.4: Percentage of total energy consumption of each appliance for Dataport homes

and all appliances with ADMM-RR, we observe that VRNN-DIS-ALL achieves a performance improvement of 41%. Our overall MAE is comparable to Tomkins et al.'s models, despite having no careful encoding of domain-specific temporal, contextual, and structural dependencies using graphical templates, paving the way for a model that can be extended easily to other settings.

Figure 6.5 compares the percentage of energy consumption by each appliance with the actual energy consumption percentages. Our actual percentage values for *refrigerator* differ by less than 1% for buildings 1 and 6. We observe a similar trend for *light*, where VRNN-DIS-ALL's predictions achieve the exact same percentage for building 6 and only an actual difference in percentage values of $< 4\%$ for building 1. Again, comparing the percentages for building 3 with the percentages reported by Tomkins et al., we observe that our model's percentage prediction for *refrigerator* deviates by 13% from the actual percentage, while theirs deviates by 16.5%. Similarly, for *dishwasher*, our predictions deviate by 30% while Tomkins

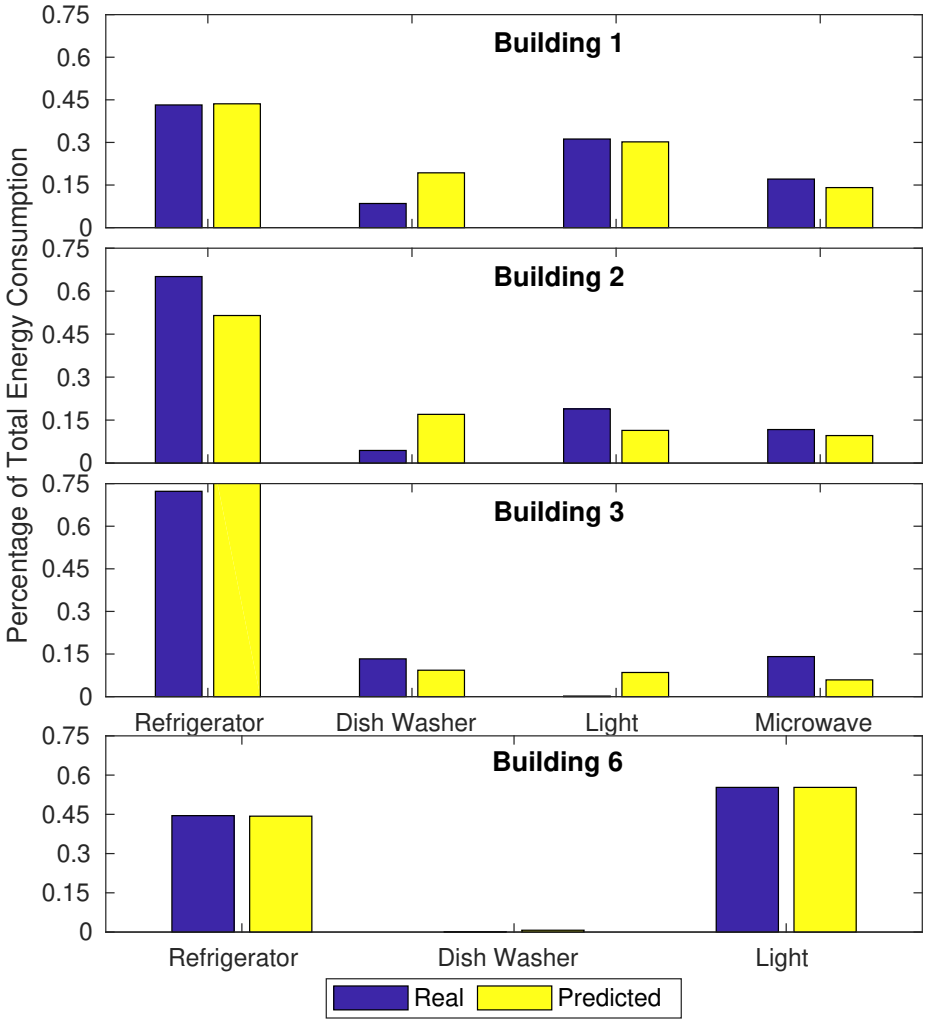


Figure 6.5: Percentage of total energy consumption of each appliance for REDD homes

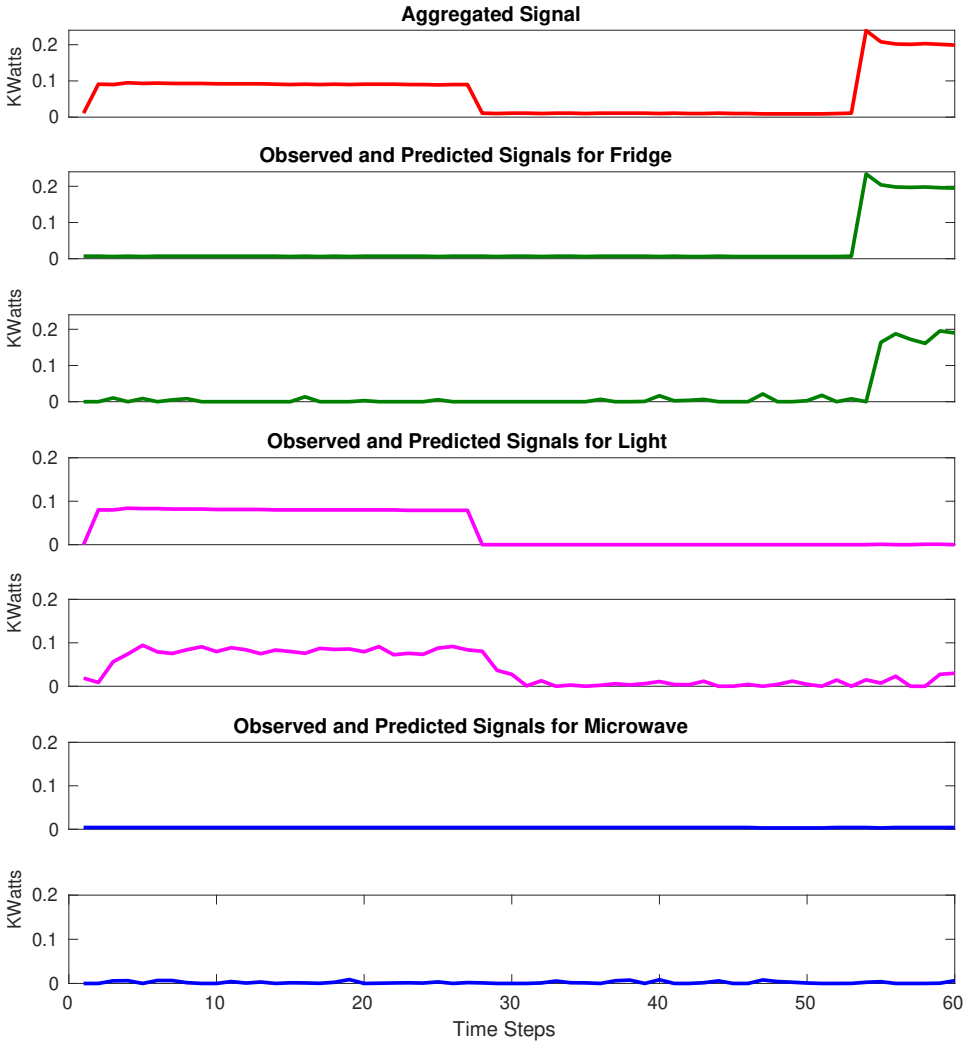


Figure 6.6: Figures showing an example disaggregation by VRNN-DIS-ALL in REDD using aggregated and disaggregated ground truth and predicted signals.

et al.'s deviate by 49%.

Further, the latent variable abstractions help our model discern which appliance(s) contribute to the aggregated power consumption and distinguish between appliance signatures, demonstrating the ability to perform blind source separation [107]. In Figure 6.6, we show an example of disaggregation for REDD. We observe that the aggregated energy consumption (first subfigure from top) is significantly contributed by *light* and *refrigerator*. Our model accurately detects both these phenomena in the predictions by identifying the presence of two peaks in this time period and the respective appliances responsible for them. These qualitative results demonstrate that our model is indeed learning to split the aggregated energy consumption into its component appliance signals.

Training	Building				
	1	**2**	**3**	**6**	**AVG**
Same building	32.00	31.63	25.00	7.50	**25.17**
Unseen building	40.00	25.25	17.75	71.33	**35.54**

Table 6.3: VRNN-DIS-ALL MAE results on different data seen for training the REDD dataset

Testing on Unseen Data

We evaluate the performance of our model training on all buildings leaving one building out and testing on that building. From Table 6.3, we can see that the MAEs for buildings 2 and 3 improve while building 1 gets a comparable MAE. The overall MAE across all buildings and appliances is also comparable to the result obtained when training on the same building with a difference of only ~ 10 in MAE and still achieving a superior prediction performance than ADMM-RR, illustrating the ability of our models to be extensible across buildings of the same dataset.

6.4 Discussion

In this chapter, we presented a novel deep generative framework that adapts a very recently developed generative model, VRNNs for energy disaggregation. We demonstrated that our model is capable of performing sequence-to-many-sequence prediction to disaggregate the aggregated energy consumption into individual appliance consumption signals. We further demonstrated that our models are capable of achieving superior performance in two well-known real-world energy disaggregation datasets DataPort and REDD, achieving 29% and 41% improvement in MAE from the existing state-of-the-art approaches. We also demonstrated the capability of our framework in accurately predicting energy consumption of appliances that consume less power and have no discernible repeating pattern, thus paving the way toward a fine-grained and informed energy disaggregation. Further, the latent variable abstractions help in achieving good prediction performance on previously unseen data. There are many exciting future directions. The generative nature of our models facilitates generating synthetic data that captures the minute variations in the signal. The latent variable abstractions can potentially be tuned to achieve good prediction performance across energy signals from different locations.

7 Conclusions and Future Work

Due to the enormous amount of data that is being created at different rates in cities, there is a great opportunity for machine learning models to transform solutions to improve the quality of life of residents in the cities. This thesis presented several machine learning (ML) models for different problems for smart cities. We compared and combined different machine learning approaches, which we adapted carefully for problems such as classification, prediction and disaggregation. Moreover, we analyzed and provided insightful results for the comparison and the combination of two machine learning approaches such as probabilistic graphical models and deep learning. In this section, we summarize the key contributions, present future directions that our research can take in the short and long term and explain the future challenge on productionizing machine learning models for smart cities.

7.1 Key Takeaways

We started implementing ensemble learning models to predict pump operation status from two countries in Africa. Our XGBoost and Random Forest models are compared to Super Vector Machine (SVM) and Linear Regression models. We found that in most of the metrics, Precission, Recall and F1, our ensemble models perform better than the base lines. Besides, we identified the individual and group features that contribute the most in improving the results in spite of some class imbalances.

Then, we explored two multi-step prediction model for hourly water consumption forecasting in fourteen buildings at Binghamton University campus. Although these models correspond to two different approaches such as PG and DL models, both of them are discriminative and suitable for prediction tasks. We found that our two models perform better than baselines such as ARIMA and Linear Regression. Moreover, our probabilistic approach, SGCRF, presented better results than a sequence-to-sequence LSTM-based model in most of the buildings from campus. We also tested the LSTM-based encoder-decoder for another problem such as the time resolution prediction for emergency events, which can help improve assignment of resources. This work was especially interesting as we show how to adapt

a sequence-to-sequence deep learning model to a non-equidistant dataset..

Finally we developed a variational model that combines the best qualities of two types of models to perform energy disaggregation. For this task, we trained our model on previous total energy house consumption in a period of time and its respective disaggregated signals, which represent different appliances' consumption. In this case, we combined the best of PG and the DL models through the use of latent variables. This latent space contributed to the disaggregation and generation jointly while allowing variability in the output and learning the essence of the signals.

7.2 Inmediate Work Extensions

As future work, we plan to continue adapting our prediction models to other smart cities problems, to work with more variables in a sequential manner and to extend our disaggregated model can be applied to other signal processing areas. The following subsections explain with more detail the inmediate extensions of our work and other works addressing the same problems and using similar approaches.

7.2.1 Sequential Models for Other Domains

Models in this thesis work with equidistant and unequidistant sequential datasets for prediction of more than one time step in the future. For all this variants of inputs, our models, representational learning and preprocessing stages can be tailored in order to obtain impactful results. For example, multi-step prediction can help monitor quality air measuarements, waste volumen generation, flood and traffic alerts, wind speed and other smart city problems [108]–[110]. These monitoring results can suggest updates design and planning of systems such as utilities production in smart grids or schedules for public transportation routes. In other words, this type of learning can also be supported by reinforcement learning models. Reinfocement Learning can optimize the performance of a system thorugh the selection of the most convenient actions to change its status over time, such as the work of [111] to control traffic signal.

7.2.2 Joint Prediction

We test our sequential models encoder-decoder to do prediction of water or energy consumption based on historic consumption of both resources. Our hypothesis was that an attention mechanism can improve the performance of the seq2seq model used in 4.5. In our initial experiments, we did not observe a significant performance improvement compared to an approach without attention.

However, we believe that this problem is worth investigating as energy and water have an underlying connection via the appliances that use them both such as washing machine or dishwasher. For example, recent work [112] showed interesting results when implementing an attention mechanism based on a bi-directional long short-term memory model and testing it in five multivariate datasets.

7.2.3 Other Disaggregation Problems

Disaggregation models can be applied to many other signals that need to be split to know what are their specific sources or components. For example, water consumption can also be disaggregated to know the spots or appliances inside a house that are using it the most. From a larger perspective, environment water flows can be measured in specific spots to learn the stream and confluences contribution in certain water reservoirs. On the other hand, this model can be useful in other domains and for a larger number of aggregated signals such as the very well known cokctail party or blind source problem. In this type of problem, there exist several microphones receiveing more than one voice or sound signal. The task consists on disaggregating the sources signals using the different aggregated input values. We believe this problem presents the next challenge in the speech recognition field and other related signal processing problems such as in [113], [114].

7.3 Long Term Extensions

Our machine learning models have proved to work well in individual tasks for smart cities. However, smart homes data collection can improve the solutions for this individual tasks and benefit from one another to solve larger problems. For example, when being able to predict the demand of water, energy and other utilities, there are possibilities to optimize their use in-house. Moreover, prediction might rely not only on one or two signal, as in our joint water and energy consumption prediction, but on several other variables that can contribute to one another in different periods of time. On the other hand, utilities market decision makers that access this type of information have more tools to optimize their services delivery and to better forecast the price of the a service, such as electricity in this work [115].

The cost of collecting the amount of data for certain type of models might be too high. For that reason, a future challenge is to develop models which can work well with limited datasets. Moreover, transfer learning have started to make available more opportunities to work with limited data in smart cities [116]. Transfer learning takes advantage of the model learned for one task to perform well in a

related second task, which can also include testing our models on previously unseen data. Other approaches to work with small datasets are few-shot learning and one-shot learning, which can contribute tremendously to escenarios where not enough data has not been collected yet [117].

Finally, more and more applications require to have explanations of the results of the models, and smart cities might need them too. As previously mentioned, probabilistic graphical models imply more interpretable models. However, sometimes they are not able to scale to solve some problems in a production level environment. This type of model still have room to improve computational calculations for training and inference time. At the same time, the combination with deep learning model is paving the way in new approaches known as deep probabilistic graphical models, which we notice are being explored more and more.

7.4 Future Challenge

As in many other domains, smart cities present optimization challenges at both levels, macro or for strategic planning and micro or for real time operation, routing and scheduling. Our work focuses in the first group due to the availability of the data to train models off-line. However, we acknowledge that most of the productionizing challenge focuses on the second group of problems. These kind of problems require an specific approach from a networking and behavioral perspective of the consumption or demand of utilities at a finer granularity. For example, some works are working on real-time problems such as detection of fires and combustion in forests [118], [119], energy-efficienty scheduling [120] and detection of emergency sounds for deaf people [121]. Moreover, one of the most popular application is the control of mobile traffic which are a natural problem for real-time approaches [122]. To improve, monitor and recognize patterns in the delivery of content and services in the smart cities grids, represent a great contribution and the future challenge on machine learning models for smart cities.

Bibliography

[1] G. Bejarano, M. Jain, A. Ramesh, A. Seetharam, and A. Mishra, "Predictive analytics for smart water management in developing regions," in *The 4th IEEE International Conference on Smart Computing (SMARTCOMP)*, 2018.

[2] G. Bejarano, A. Kulkarni, R. Raushan, A. Seetharam, and A. Ramesh, "Swap: Probabilistic graphical and deep learning models for water consumption prediction," in *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, 2019.

[3] G. Bejarano, A. Kulkarni, X. Luo, A. Seetharam, and A. Ramesh, "Deeper: A deep learning based emergency resolution time prediction system," in *2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*, IEEE, 2020, pp. 490–497.

[4] G. Bejarano, D. DeFazio, and A. Ramesh, "Deep latent generative models for energy disaggregation," in *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, 2019.

[5] J. Chung, K. Kastner, L. Dinh, K. Goel, A. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," in *Proceedings of the International Conference on Neural Information Processing Systems (NIPS)*, 2015.

[6] Y. Freund, R. E. Schapire, *et al.*, "Experiments with a new boosting algorithm," in *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, 1996.

[7] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[8] ——, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[9] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the Eighteenth International Conference on Machine Learning*, 2001.

[10] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, "Conditional random fields as recurrent neural networks," in *Proceedings of the IEEE international conference on computer vision*, 2015.

[11] S. Gao and M. R. Gormley, "Training for gibbs sampling on conditional random fields with neural scoring factors," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

[12] G. Chen, Y. Li, and S. N. Srihari, "Word recognition with deep conditional random fields," 2016.

[13] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, pp. 107–116, 1998.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. 2016.

[15] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014.

[16] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[17] A. B. Dieng, "Deep probabilistic graphical modeling," PhD thesis, Columbia University, 2020.

[18] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[19] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *The International Conference on Learning Representations*, 2014.

[20] O. Fabius and J. R. van Amersfoort, *Variational recurrent auto-encoders*, 2015. arXiv: `1412.6581`.

[21] *Humanosphere report*, `http://www.humanosphere.org/world-politics/2014/12/tanzania-failed-fix-water-access-problem/`.

[22] *The taarifa platform*, `http://taarifa.org/`.

[23] R. Cronk and J. Bartram, "Factors influencing water system functionality in nigeria and tanzania: A regression and bayesian network analysis," *Environmental Science & Technology*, vol. 51, no. 19, pp. 11 336–11 345, 2017.

[24] M. B. Fisher, K. F. Shields, T. U. Chan, E. Christenson, R. D. Cronk, H. Leker, D. Samani, P. Apoya, A. Lutz, and J. Bartram, "Understanding handpump sustainability: Determinants of rural water source functionality in the greater afram plains region of ghana," *Water resources research*, vol. 51, no. 10, pp. 8431–8449, 2015.

[25] A. Jiménez and A. Pérez-Foguet, "The relationship between technology and functionality of rural water points: Evidence from tanzania," *Water science and technology*, vol. 63, no. 5, pp. 948–955, 2011.

[26] T. Foster, "Predictors of sustainability for community-managed handpumps in sub-saharan africa: Evidence from liberia, sierra leone, and uganda," *Environmental Science & Technology*, vol. 47, no. 21, pp. 12 037–12 046, 2013.

[27] D. Whittington, J. Davis, L. Prokopy, R. Komives Kristin and Thorsten, H. Lukacs, A. Bakalian, and W. Wakeman, "How well is the demand-driven, community management model for rural water supply systems doing? evidence from bolivia, peru and ghana," *Water Policy*, vol. 11, no. 6, pp. 696–718, 2009.

[28] M. Ali and A. M. Qamar, "Data analysis, quality indexing and prediction of water quality for the management of rawal watershed in pakistan," in *Digital Information Management (ICDIM), 2013 Eighth International Conference on*, IEEE, 2013.

[29] A. J. F. de Palencia and A. Pérez-Foguet, "Quality and year-round availability of water delivered by improved water points in rural tanzania: Effects on coverage," *Water Policy*, vol. 14, no. 3, pp. 509–523, 2012.

[30] Y. Liu, Y. Liang, S. Liu, D. S. Rosenblum, and Y. Zheng, "Predicting urban water quality with ubiquitous data," *arXiv preprint arXiv:1610.09462*, 2016.

[31] S. Deleawe, J. Kusznir, B. Lamb, and D. J. Cook, "Predicting air quality in smart environments," *Journal of Ambient Intelligence and Smart Environments*, vol. 2, no. 2, pp. 145–154, 2010.

[32] D. Tien Bui, B. Pradhan, O. Lofman, and I. Revhaug, "Landslide susceptibility assessment in vietnam using support vector machines, decision tree, and naive bayes models," *Mathematical Problems in Engineering*, vol. 2012, 2012.

[33] B. B. Gulyani, J. A. Mangai, and A. Fathima, "An approach for predicting river water quality using data mining technique," in *Industrial Conference on Data Mining*, 2015.

[34] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, ACM, 2016.

[35] *Executive summary of the national climate assessment*, 2014. [Online]. Available: `https://nca2014.globalchange.gov/highlights/report-findings/extreme-weather`.

[36] *Drought.gov,us drought portal*, 2019. [Online]. Available: `https://www.drought.gov/drought/states/california`.

[37] *Wikipedia: Cape town water crisis*, 2019. [Online]. Available: `https://en.wikipedia.org/wiki/Cape_Town_water_crisis`.

[38] H. Kwon, J. E. Fischer, M. Flintham, and J. Colley, "The connected shower: Studying intimate data in everyday life," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 2, no. 4, p. 176, 2018.

[39] H. Assem, S. Ghariba, G. Makrai, P. Johnston, L. Gill, and F. Pilla, "Urban water flow and water level prediction based on deep learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, 2017.

[40] D. DeFazio, A. Ramesh, and A. Seetharam, "Nycer: A non-emergency response predictor for nyc using sparse gaussian conditional random fields," in *Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, ser. MobiQuitous '18, Association for Computing Machinery, 2018, pp. 187–196.

[41] M. Wytock and Z. Kolter, "Sparse gaussian conditional random fields: Algorithms, theory, and application to energy forecasting," in *International conference on machine learning*, 2013, pp. 1265–1273.

[42] C. Tull, E. Schmitt, and P. Atwater, "How much water does turf removal save? applying bayesian structural time-series to california residential water demand," in *KDD Workshop on Data Science for Food, Energy and Water*.

[43] D. Kofinas, E. Papageorgiou, C. Laspidou, N. Mellios, and K. Kokkinos, "Daily multivariate forecasting of water demand in a touristic island with the use of artificial neural network and adaptive neuro-fuzzy inference system," in *Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks (CySWater)*, IEEE, 2016.

[44] A. Jabbari and D.-H. Bae, "Application of artificial neural networks for accuracy enhancements of real-time flood forecasting in the imjin basin," *Water*, vol. 10, no. 11, p. 1626, 2018.

[45] A. Candelieri, D. Soldi, and F. Archetti, "Short-term forecasting of hourly water consumption by using automatic metering readers data," *Procedia Engineering*, vol. 119, pp. 844–853, 2015, Computing and Control for the Water Industry (CCWI2015) Sharing the best practice in water management, ISSN: 1877-7058. DOI: https://doi.org/10.1016/j.proeng.2015.08.948.

[46] G. Bejarano, M. Jain, A. Ramesh, A. Seetharam, and A. Mishra, "Predictive analytics for smart water management in developing regions," in *The 4th IEEE International Conference on Smart Computing (SMARTCOMP)*, 2018.

[47] A. Endo, I. Tsurita, K. Burnett, and P. M. Orencio, "A review of the current state of research on the water, energy, and food nexus," *Journal of Hydrology: Regional Studies*, vol. 11, pp. 20–30, 2017.

[48] B. Ali, "Forecasting model for water-energy nexus in alberta, canada," *Water-Energy Nexus*, vol. 1, no. 2, pp. 104–115, 2018.

[49] K. A. Rambo, D. M. Warsinger, S. J. Shanbhogue, J. H. L. V, and A. F. Ghoniem, "Water-energy nexus in saudi arabia," *Energy Procedia*, vol. 105, pp. 3837–3843, 2017, 8th International Conference on Applied Energy, ICAE2016, 8-11 October 2016, Beijing, China.

[50] H. Hoff, *Understanding the nexus; background paper for the bonn2011 conference: The water, energy and food security nexus; stockholm environment institute: Stockholm, sweden, 2011*, 2011.

[51] Q. Wang, S. Li, and R. Li, "Forecasting energy demand in china and india: Using single-linear, hybrid-linear, and non-linear time series forecast techniques," *Energy*, vol. 161, pp. 821–831, 2018.

[52] G. Borowik, Z. M. Wawrzyniak, and P. Cichosz, "Time series analysis for crime forecasting," in *2018 26th International Conference on Systems Engineering (ICSEng)*, IEEE, 2018.

[53] K. Mason, J. Duggan, and E. Howley, "Forecasting energy demand, wind generation and carbon dioxide emissions in ireland using evolutionary neural networks," *Energy*, vol. 155, pp. 705–720, 2018.

[54] P. Arjunan, M. Srivastava, A. Singh, and P. Singh, "Openban: An open building analytics middleware for smart buildings," in *Proceedings of the 12th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services on 12th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, 2015.

[55] Z. Zhang and K. P. Lam, "Practical implementation and evaluation of deep reinforcement learning control for a radiant heating system," in *Proceedings of the 5th ACM Conference on Systems for Energy-Efficient Built Environments (BuildSys)*, 2018.

[56] H.-W. Kang and H.-B. Kang, "Prediction of crime occurrence from multimodal data using deep learning," *PloS one*, vol. 12, no. 4, e0176244, 2017.

[57] G. Mittal, K. B. Yagnik, M. Garg, and N. C. Krishnan, "Spotgarbage: Smartphone app to detect garbage using deep learning," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, 2016.

[58] R. Bhandari, A. Nambi, V. Padmanabhan, and B. Raman, "Deeplane: Camera-assisted gps for driving lane detection," in *Proceedings of the 5th ACM Conference on Systems for Energy-Efficient Built Environments (BuildSys)*, 2018.

[59] T. Yabe, K. Tsubouchi, and Y. Sekimoto, "Cityflowfragility: Measuring the fragility of people flow in cities to disasters using gps data collected from smartphones," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, no. 3, p. 117, 2017.

[60] R. Bhardwaj, G. K. Tummala, G. Ramalingam, R. Ramjee, and P. Sinha, "Autocalib: Automatic traffic camera calibration at scale," in *Proceedings of the 4th ACM International Conference on Systems for Energy-Efficient Built Environments (BuildSys)*, 2017.

[61] D. Chen, J. Breda, and D. Irwin, "Staring at the sun: A physical black-box solar performance model," in *Proceedings of the 5th ACM Conference on Systems for Energy-Efficient Built Environments (BuildSys)*, 2018.

[62] F. N. Melzi, T. Touati, A. Same, and L. Oukhellou, "Hourly solar irradiance forecasting based on machine learning models," in *15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2016.

[63] *Paper: Code and data*, 2019. [Online]. Available: https://bitbucket.org/gissemari/water-consumption-prediction.

[64] *Nyc open data*, https://data.cityofnewyork.us/Public-Safety/Emergency-Response-Incidents/pasr-j7fb.

[65] I. Fox, L. Ang, M. Jaiswal, R. Pop-Busui, and J. Wiens, "Deep multi-output forecasting: Learning to accurately predict blood glucose trajectories," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '18, ACM, 2018, pp. 1387–1395.

[66] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.

[67] L. Peng, L. Chen, Z. Ye, and Y. Zhang, "Aroma: A deep multi-task learning based simple and complex human activity recognition method using wearable sensors," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 2, pp. 1–16, Jul. 2018.

[68] Y. Guan and T. Plötz, "Ensembles of deep lstm learners for activity recognition using wearables," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 1, no. 2, 11:1–11:28, Jun. 2017.

[69] W. Cheng, Y. Shen, Y. Zhu, and L. Huang, "A neural attention model for urban air quality inference: Learning the weights of monitoring stations," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[70] A. A. Varamin, E. Abbasnejad, Q. Shi, D. C. Ranasinghe, and H. Rezatofighi, "Deep auto-set: A deep auto-encoder-set network for activity recognition using wearables," in *Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, ser. MobiQuitous '18, ACM, 2018, pp. 246–253.

[71] A. Chohlas-Wood, A. Merali, W. Reed, and T. Damoulas, "Mining 911 calls in new york city: Temporal patterns, detection, and forecasting," in *Workshops at the Twenty-Nineth AAAI Conference on Artificial Intelligence*, 2015.

[72] Y. Zha and M. M. Veloso, "Profiling and prediction of non-emergency calls in nyc," in *Workshops at the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[73] X. Zhao and J. Tang, "Modeling temporal-spatial correlations for crime prediction," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, ser. CIKM '17, Association for Computing Machinery, 2017.

[74] F. Yi, Z. Yu, F. Zhuang, and B. Guo, "Neural network based continuous conditional random field for fine-grained crime prediction," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI)*, 2019.

[75] B. Cortez, B. Carrera, Y.-J. Kim, and J.-Y. Jung, "An architecture for emergency event prediction using lstm recurrent neural networks," *Expert Systems with Applications*, vol. 97, pp. 315–324, 2018.

[76] N. Pathak, A. Ba, J. Ploennigs, and N. Roy, "Forecasting gas usage for big buildings using generalized additive models and deep learning," in *The 4th IEEE International Conference on Smart Computing (SMARTCOMP)*, 2018.

[77] B. Qolomany, A. Al-Fuqaha, D. Benhaddou, and A. Gupta, "Role of deep lstm neural networks and wi-fi networks in support of occupancy prediction in smart buildings," in *2017 IEEE 19th International Conference on High Performance Computing and Communications; IEEE 15th International Conference on Smart City; IEEE 3rd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, 2017.

[78] A. Kulkarni, A. Seetharam, and A. Ramesh, "Deepfit: Deep learning based fitness center equipment use modeling and prediction," in *Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, ser. MobiQuitous '19, Association for Computing Machinery, 2019.

[79] A. Kulkarni, A. Seetharam, A. Ramesh, and J. D. Herath, "Deepchannel: Wireless channel quality prediction using deep learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 443–456, 2020.

[80] V. Reddy, P. Yedavalli, S. Mohanty, and U. Nakhat, *Deep air: Forecasting air pollution in beijing, china*, 2018.

[81] S. Tomkins, J. Pujara, and L. Getoor, "Disambiguating energy disaggregation: A collective probabilistic approach," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*, 2017, pp. 2857–2863.

[82] K. Shaloudegi, A. György, C. Szepesvári, and W. Xu, "SDP relaxation with randomized rounding for energy disaggregation," in *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, 2016.

[83] G. W. Hart, "Nonintrusive appliance load monitoring," *Proceedings of the IEEE*, vol. 80, no. 12, pp. 1870–1891, 1992.

[84] Z. Ghahramani and M. I. Jordan, "Factorial hidden markov models," *Machine Learning*, vol. 29, no. 2, pp. 245–273, 1997.

[85] H. Kim, M. Marwah, M. Arlitt, G. Lyon, and J. Han, "Unsupervised disaggregation of low frequency power measurements," in *Proceedings of the SIAM International Conference on Data Mining*, 2011.

[86] J. Z. Kolter and T. Jaakkola, "Approximate inference in additive factorial hmms with application to energy disaggregation," in *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, vol. 22, La Palma, Canary Islands: PMLR, Apr. 2012, pp. 1472–1482.

[87] O. Parson, S. Ghosh, M. Weal, and A. Rogers, "Non-intrusive load monitoring using prior models of general appliance types," in *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[88] M. J. Johnson and A. S. Willsky, "Bayesian nonparametric hidden semi-markov models," *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 673–701, Feb. 2013.

[89] H. Altrabalsi, V. Stankovic, J. Liao, and L. Stankovic, "Low-complexity energy disaggregation using appliance load modelling," *AIMS Energy*, vol. 4, no. 1, pp. 884–905, 2016.

[90] S. K. Barker, "Model-driven analytics of energy meter data in smart homes," PhD thesis, University of Massachusetts - Amherst, 2014.

[91] A. Faustine, N. H. Mvungi, S. Kaijage, and K. Michael, "A survey on non-intrusive load monitoring methodies and techniques for energy disaggregation problem," *Arxiv*, 2017.

[92] S. Makonin, F. Popowich, I. V. Bajić, B. Gill, and L. Bartram, "Exploiting hmm sparsity to perform online real-time nonintrusive load monitoring," *IEEE Transactions on Smart Grid*, vol. 7, no. 6, pp. 2575–2585, 2016.

[93] L. Li and H. Zha, "Household structure analysis via hawkes processes for enhancing energy disaggregation," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.

[94] M. Zhong, N. Goddard, and C. Sutton, "Latent bayesian melding for integrating individual and population models," in *Advances in neural information processing systems*, 2015.

[95] J. Kelly and W. Knottenbelt, "Neural nilm: Deep neural networks applied to energy disaggregation," in *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments (BuildSys)*, 2015.

[96] H. Lange and M. Bergés, "The neural energy decoder: Energy disaggregation by combining binary subcomponents," in *Proceeding of the 3rd International Workshop on Non-Intrusive Load Monitoring*, 2016.

[97] K. S. Barsim, L. Mauch, and B. Yang, "Neural network ensembles to real-time identification of plug-level appliance measurements," *CoRR*, 2018.

[98] P. P. M. do Nascimento, "Applications of deep learning techniques on nilm," PhD thesis, Universidade Federal do Rio de Janeiro, 2016.

[99] C. Zhang, M. Zhong, Z. Wang, N. H. Goddard, and C. A. Sutton, "Sequence-to-point learning with neural networks for non-intrusive load monitoring," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[100] A. Huss, "Hybrid model approach to appliance load disaggregation: Expressive appliance modelling by combining convolutional neural networks and hidden semi markov models.," Master's thesis, 2015.

[101] N. Batra, Y. Jia, H. Wang, K. Whitehouse, *et al.*, "Transferring decomposed tensors for scalable energy breakdown across regions.," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[102] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proceedings of the 31st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 32, PMLR, 2014, pp. 1278–1286.

[103] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, 2015.

[104] *Pecan street inc., dataport*, 2016.

[105] J. Z. Kolter and M. J. Johnson, "REDD: A public data set for energy disaggregation research," in *Proceedings of the KDD Workshop on Sustainability (SustKDD)*, 2011.

[106] N. Batra, J. Kelly, O. Parson, H. Dutta, W. Knottenbelt, A. Rogers, A. Singh, and M. Srivastava, "NILMTK: An open source toolkit for non-intrusive load monitoring," in *Proceedings of the 5th International Conference on Future Energy Systems*, ACM, 2014, pp. 265–276.

[107] M. Pal, R. Roy, J. Basu, and M. S. Bepari, "Blind source separation: A review and analysis," in *International Conference Oriental COCOSDA held jointly with Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE)*, 2013.

[108] I.-F. Kao, Y. Zhou, L.-C. Chang, and F.-J. Chang, "Exploring a long short-term memory based encoder-decoder framework for multi-step-ahead flood forecasting," *Journal of Hydrology*, vol. 583, p. 124 631, 2020.

[109] L. Xiang, J. Li, A. Hu, and Y. Zhang, "Deterministic and probabilistic multi-step forecasting for short-term wind speed based on secondary decomposition and a deep learning method," *Energy Conversion and Management*, vol. 220, p. 113 098, 2020.

[110] Y. Xin, D. Miao, M. Zhu, C. Jin, and X. Lu, "Internet: Multistep traffic forecasting by interacting spatial and temporal features," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 3477–3480.

[111] C. Chen, H. Wei, N. Xu, G. Zheng, M. Yang, Y. Xiong, K. Xu, and Z. Li, "Toward a thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control," *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, vol. 34, pp. 3414–3421, 2020.

[112] S. Du, T. Li, Y. Yang, and S.-J. Horng, "Multivariate time series forecasting via attention-based encoder–decoder framework," *Neurocomputing*, vol. 388, pp. 269–279, 2020.

[113] E. Tzinis, S. Venkataramani, Z. Wang, C. Subakan, and P. Smaragdis, "Two-step sound source separation: Training on learned latent targets," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[114] A. K. Clarke, S. F. Atashzar, A. D. Vecchio, D. Barsakcioglu, S. Muceli, P. Bentley, F. Urh, A. Holobar, and D. Farina, "Deep learning for robust decomposition of high-density surface emg signals," *IEEE Transactions on Biomedical Engineering*, vol. 68, no. 2, pp. 526–534, 2021.

[115] W. Yang, J. Wang, T. Niu, and P. Du, "A novel system for multi-step electricity price forecasting for electricity market management," *Applied Soft Computing*, vol. 88, p. 106 029, 2020.

[116] S. Gomez-Rosero, M. A. M. Capretz, and S. Mir, "Transfer learning by similarity centred architecture evolution for multiple residential load forecasting," *Smart Cities*, vol. 4, no. 1, pp. 217–240, 2021.

[117] V.-N. Tuyet-Doan, T.-D. Do, N.-D. Tran-Thi, Y.-W. Youn, and Y.-H. Kim, "One-shot learning for partial discharge diagnosis using ultra-high-frequency sensor in gas-insulated switchgear," *Sensors*, vol. 20, no. 19, 2020.

[118] Z. Jiao, Y. Zhang, J. Xin, L. Mu, Y. Yi, H. Liu, and D. Liu, "A deep learning based forest fire detection approach using uav and yolov3," in *2019 1st International Conference on Industrial Artificial Intelligence (IAI)*, 2019, pp. 1–5. DOI: 10.1109/ICIAI.2019.8850815.

[119] X. Yan, H. Cheng, Y. Zhao, W. Yu, H. Huang, and X. Zheng, "Real-time identification of smoldering and flaming combustion phases in forest using a wireless sensor network-based multi-sensor system and artificial neural network," *Sensors*, vol. 16, no. 8, 2016, ISSN: 1424-8220. DOI: 10.3390/s16081228. [Online]. Available: https://www.mdpi.com/1424-8220/16/8/1228.

[120] Q. Zhang, M. Lin, L. T. Yang, Z. Chen, and P. Li, "Energy-efficient scheduling for real-time systems based on deep q-learning model," *IEEE Transactions on Sustainable Computing*, vol. 4, no. 1, pp. 132–141, 2019. DOI: 10.1109/TSUSC.2017.2743704.

[121] S. Liu and J. Du, "Poster: Mobiear-building an environment-independent acoustic sensing platform for the deaf using deep learning," ser. MobiSys '16 Companion, Singapore, Singapore: Association for Computing Machinery, 2016, p. 50, ISBN: 9781450344166. DOI: 10.1145/2938559.2948831. [Online]. Available: https://doi.org/10.1145/2938559.2948831.

[122] M. Abbasi, A. Shahraki, and A. Taherkordi, "Deep learning for network traffic monitoring and analysis (ntma): A survey," *Computer Communications*, vol. 170, pp. 19–41, 2021.