STARS

University of Central Florida
STARS

Electronic Theses and Dissertations, 2020-

2021

Human Behavior in Domestic Environments: Prediction and Applications

Sharare Zehtabian
University of Central Florida

 Part of the Theory and Algorithms Commons
Find similar works at: https://stars.library.ucf.edu/etd2020
University of Central Florida Libraries http://library.ucf.edu

This Doctoral Dissertation (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations, 2020- by an authorized administrator of STARS. For more information, please contact STARS@ucf.edu.

STARS Citation

Zehtabian, Sharare, "Human Behavior in Domestic Environments: Prediction and Applications" (2021). *Electronic Theses and Dissertations, 2020-*. 945.
https://stars.library.ucf.edu/etd2020/945

HUMAN BEHAVIOR IN DOMESTIC ENVIRONMENTS: PREDICTION AND APPLICATIONS

by

SHARARE ZEHTABIAN
M.S. University of Tehran, 2016
B.S. Sharif University of Technology, 2012

A dissertation submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
in the Department of Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida, Orlando, Florida

Fall Term
2021

Major Professor: Ladislau Bölöni and Damla Turgut

© 2021 Sharare Zehtabian

ABSTRACT

A longstanding goal of human behavior science is to model and predict how humans interact with each other or with other systems. Such models are beneficial and have many applications, including designing and implementing assistive technologies, improving users' experiences and quality of life and making better decisions to create public policies. Behavior is highly complex due to uncertainties and a lack of scientific tools to measure it. Hence prediction of human behavior cannot be 100% accurate. However, prediction is also not hopeless because the biological needs, as well as cultural conventions (for instance, regarding meal times) set the general patterns of the humans' daily behavior. Furthermore, while individual humans might adjust these patterns according to their own preferences, they also show some degree of consistency in their daily routine.

In this dissertation, we focus on interrelated challenges of improving the prediction models for human daily activities and developing techniques through which intelligent applications can benefit from this improved prediction. We describe techniques for creating predictive models that can help humans in their daily life using deep learning-based models. One of the challenges of learning based approaches in this setting is the scarcity of data. If we are collecting information about a given human in a home, our database will increase with exactly one sample a day – this is insufficient for deep learning algorithms that are often trained on datasets with millions of samples. We investigate three directions through which the paucity of samples can be overcome.

First, we discuss techniques through which, starting from a small number of representative samples, we can generate much larger synthetic datasets that capture the statistical properties of the real world data, and can be used in training. We consider an application where we apply human behavior prediction to the practical problem of improving the quality of experience. By learning to predict the experience requested by the user, we are able to perform intelligent pre-caching, and

achieve higher average quality of experience for a given available network bandwidth.

Another direction we investigate is the collection of data from multiple users. This creates multiple challenges. First, users would prefer to minimize the shared personal data. This requires us to investigate techniques that learn predictive models from multiple user experiences without requiring the users to upload their data to a common repository. We adapt the technique of federated learning, which requires the users to only share the training gradients on a model that had been sent by a central server, but not raw data. We investigate procedures that allow the user to obtain the best possible model for her own prediction while minimizing the amount of data disclosed. The second challenge is that not all the users benefit to the same degree from creating a central learning model; by investigating how much the user can benefit, we can stop the learning process and implicit privacy loss earlier.

Finally, we developed predictive models for the spread of pandemics and techniques that use these predictions to recommend Non-Pharmaceutical Interventions (NPIs) to local stakeholders. We find that the prediction of pandemics is also conditioned on the behavior of individual humans and the actions taken by the governments and, especially in the early phases of the pandemic, suffers from a lack of data. We used a combination of a deep learning-based predictive model with a compartmental model, which is trained on the months elapsed from the pandemic and predicts infection rates for the next months. We used cultural and geographical attributes as constant features along with the history of cases and deaths as context features and NPIs as action features to train a single predictive model that can predict both the infection rate and the stringency of the NPIs deployed by policymakers for all countries / regions. We found that the stringency is not always aligned with the number of cases but also depends on political, economic and cultural factors.

ACKNOWLEDGMENTS

First and foremost, I am grateful for my advisors Ladislau Bölöni and Damla Turgut for their continuous guidance and support during my Ph.D. and giving me the opportunity to work on a variety of problems. I would like to thank my committee members Liqiang Wang and Yue Zhao for their time, support and valuable comments.

A special thank you and feeling of gratitude towards my parents for all they have done for me through the years of my studies. I am grateful for my sister, Shohre, who has always been a great friend and mentor to me and has helped me a lot throughout my studies. Finally, this dissertation was not possible without my loving husband's support, Siavash. I am beyond grateful to have him in my life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

Human behavior is defined as the way humans act and interact with each other or with other systems. Human behavior is influenced by several factors, such as genetic, cultural, and individual values and attitudes. Modeling human behaviors opens possibilities that can help us in different aspects of the daily living [82]. In the last two decades, a large number of research and commercial projects developed assistive technologies that aim to improve the quality of life for individuals by choosing the appropriate assistive actions based on the log of the user's activities of daily living [77, 1]. The main aspects of these systems are that they are easy to understand and can automatically adapt to behavioral patterns of the users in a specific context [71]. For example, we can give the user real-time feedback to improve their task deployment, automate the use of tools and devices to decrease costs, monitor users remotely, and provide e-health services according to the characteristics and needs of each particular citizen or residents [5].

Besides, we can design systems that can predict users' needs and provide information regarding various simple daily in-home scenarios. Examples include checking the news for major sport events, weather or traffic report, parking status, and checking online information for daily tasks such as appropriate cooking recipes. In these scenarios, we need a higher rate for content delivery, large bandwidth, increased capacity, low latency, and high throughput [80]. An intelligent strategy would be to predict what experience the user will need and in which time frame of the day this request will happen, so the system can cache the necessary content in advance to deliver that experience to the user. Therefore, an ability to model routine behaviors and their variations could help researchers improve such assistive technologies.

In addition to using predictive models for assistive technologies or improving user experiences, decision makers need to know about the prediction models, why they work and why they might not

work for decision making. Usually, a good predictive model provides one or more prescriptions for potential future actions that enable decision-makers to make better decisions. A practical example is predicting Covid-19 pandemic.

As manually creating a model of user behavior is prohibitively expensive, we propose to learn the predictive model from collected data. Machine learning, in particular deep learning models, made significant progress in the last decade. However, many deep learning algorithms work best under big data regimes, where the number of data samples is counted from the tens of thousands (e.g. the MNIST dataset) to 500 billion (the Common Crawl dataset used to train the GPT-3 model). The activity logs of assistive environments, in contrast, are an example of small data: the number of individual activities performed by a user each day is counted in dozens, and we expect the model to yield actionable predictions in matter of weeks after the system deployment. As another example, the prediction of pandemics such as Covid-19 pandemic is also conditioned on the behavior of individual humans and the actions taken by the governments and, especially in the early phases of the pandemic, suffers from a lack of data.

One potential solution for this challenge is generating synthetic data. Creating synthetic datasets has been applied to different problems in machine learning, such as Question-Answering challenges [86] and visual reasoning [106]. Synthetic human behavior data should capture the statistical properties of real-world behavior, but provide additional variation that helps with the training of machine learning models. Examples of technologies that can be used to generate such models include sampling from Markov chains, Poisson distributions [38] as well as generative models such as generative adversarial networks (GANs) [33].

Another solution to the problem of data scarcity is the use of collaborative learning which, by building a common model M_{shared} from the data of a pool of users, operates closer to the big data regimes favored by deep learning algorithms. The simplest choice of collaborative learning is

centralized learning: the environments transfer their logs to a cloud-based central authority that combines these logs into a common training set. A different variant of collaborative learning, federated learning [50], also relies on a cloud-based central authority but requires the environments to perform learning locally and transfer only parameters of the learned model to the central system. Having access, directly or indirectly, to more data, collaborative learning promises faster convergence.

A very important aspect of learning human behavior predictive models is the consideration of privacy. One of the fundamental principles of privacy is that of data minimization. In the context of machine learning, this principle means that the minimum amount of training data must be collected from users in order to acquire the specific benefits of the application. This principle was stated, among others, in the consumer privacy report of the US White House in 2012 [15], by the UK Information Commissioner's office [97] and it is also embedded in the European Union's General Data Protection Regulation (GDPR) [98].

The principle of data minimization, applied to a smart environment means that the environment should not disclose information unless it provides a quantifiable benefit. When this is not feasible, and everything else being equal, the system should prefer techniques such as federated learning, which can be used in ways to achieve differential privacy [31] to techniques such as centralized learning where privacy depends on assumptions about the central authority. However, the choice is not clear cut: attacks against federated learning systems had been recently demonstrated [6] and, even in the absence of attack, information can still leak through, sometimes simply by the participation of a home in a given federated learning pool.

As an application area, our topic is part of the field of human activity modeling and prediction that can be used in a decision making problem in a larger environment such as a city or a country. As the final step in this dissertation, we study the applications in predicting Covid-19 infection

rate during the pandemic that can help local stakeholders in decision making and recommending intervention plans.

The contributions of this dissertation have three distinct intellectual lineages.

- In one research direction, we investigate several strategies for predicting the information needs of a user in a smart-home [101, 103]. The paucity of datasets is a major challenge in such studies. We synthetically generate realistic content requests starting from real-world databases of user activities in smart homes. Using these synthetic datasets, we develop techniques for demand prediction and content caching that aim to optimize the quality of user satisfaction while minimizing the cost of caching and storing the data.

- In the second research direction, we focus on learning a predictive model of human activities in smart environments using collaborative learning [104]. We use state-of-the-art deep neural network-based techniques to learn predictive human activity models in the local, centralized, and federated settings. We track the temporal evolution of the data available to the learner and the data shared by the user and considered users that aim to preserve their privacy.

- Finally, we investigate human behavior modeling and prediction problem in a larger environment such as a city or a country. More specifically, we study the modeling and predicting of the infections in the Covid-19 pandemic, which was also found to be highly dependant on humans' behavior and interactions [102]. Such system could help regional governments, communities, and organizations to minimize harm when reopening during a pandemic. We take into account features dependent on social, cultural and geographical aspects alongside features created by using compartmental models in order to improve the prediction model.

CHAPTER 2: RELATED WORK

Human behavior definition. According to Nature's website, human behavior is defined as the way humans act and interact with each other or with other systems. Human behavior is influenced by several factors, such as genetic, cultural, and individual values and attitudes. When we study other topics in science such as physics, chemistry, or biology, we study organized information that has been collected during years of experiments. We might think of a scientist working in his or her laboratory with equipment such as telescopes, microscopes, and cyclotrons. The science of behavior is not one of those subjects which we can access it by using an instrument such as a telescope or microscope. We all are already very familiar with this subject since we have observed various behaviors for many years and as a result, we know so many facts about behaviors. However, this familiarity might be a disadvantage for the science of behavior since it might make us biased and might lead us to jump to conclusions [83]. Therefore, behavior is a difficult subject to study because not only cannot we access it by a specific instrument, but it contains uncertainties and is very complicated to be analyzed.

In literature, there are other descriptions of human behaviors. Pentland and Liu [74] described human behaviors as a set of dynamic models. They considered a human as a device with a large number of internal mental states. In this definition, each mental state has its own control behavior and interstate transition probabilities. Such a model of human behavior could be used to create better systems in which humans and machines can easily and naturally interact with each other. If the machine could recognize the human's behavior or, even better, if it could anticipate the human's behavior, it could adjust itself to serve the humans' needs better.

One of the best tools for understanding and managing complex systems such as human behaviors is computational modeling and simulation. In the following paragraphs, we describe the modeling

strategies for humans and the users of technologies in different contexts.

Modeling of human behavior/users. Human-centered technologies are systems that are built for humans based on human behavior models. The most important aspects of these systems is that they are simple to use and understand and they adapt automatically to behavioral patterns of the users and the context in which they are being used [71]. Behavioral modeling can help to enhance the user experience of these systems. For example, in the concept of online social networks, a social graph can be used as a classic and effective mathematical model to represent the connectivity and interaction between users [44].

Another crucial aspect of these systems is that they promote independent and convenient living for the people such as the elderly population. The aging populations all around the world are increasing and as a result of that, the number of people who are dependent on others for their daily tasks and suffer from a reduced level of autonomy is elevating [70, 29]. In addition to the elderly population, human-centered and goal-oriented technologies can improve the quality of life for people with disabilities, people with hearing or speech difficulty, visually impaired people, and people on the spectrum. Routine behaviors are specific behaviors that are defined as frequent and goal-oriented actions that people perform in different situations [40]. An ability to model routine behaviors and their variations could help researchers improve technologies that influence routine tasks to help people improve the quality of their lives [7].

To study human behaviors, we need large activity datasets. Data mining algorithms can be used to extract patterns from such datasets [58]. Also, visualization of the data could help researchers to extract information on human behaviors [93]. Also, machine learning provides new tools for researchers to better understand human behavior. With the recent advances in artificial intelligence, we can assist humans through different aspects of engineering such as biomedical, industrial, and robotics. Therefore, conceptualizing the behavior of humans becomes more important to enhance

the interaction between humans and machine [73]. The concept of user modeling has been mainly discussed in the context of human–computer interaction. For example, we model the users' behavior when we want to design an intelligent system for a group of users to improve the user experience with the system. In this model, the gathered data from the history of the interaction between the user and the systems are analyzed to estimate the intended actions of the users in the future [24].

Predictive statistical models enable the anticipation of different aspects of human behavior, such as intents, actions, and preferences [110]. Generally, user modeling can be classified into three classes: behavioral modeling, interest modeling, and intent modeling. User modeling can be defined as learning a latent representation for each user and extracting meaningful latent features from users' data. In order to learn this representation, we can use features of the system or characteristics of the user, and a user-system response matrix that can be applied to predict the response, recommendations, and so on [54]. The data could be either static data (e.g., tabular data) or sequential data (e.g., time-series data). In user modeling applications, user data are often organized in structured static datasets (e.g., user-movie rating matrix) or unstructured sequences (e.g., the purchase history of customers). To achieve representation learning, we could use either shallow models or deep models.

To inspire and improve user loyalty and developing a meaningful one-to-one relationship, personalization approaches are being considered in building technologies. To this end, we need to understand the needs of each individual and help them to efficiently achieve their goal and accordingly address each individual's need in a given context [78]. If we can gather more information to model the users, we can easily build personalized services for the users [30].

Predicting human behavior. Advances in machine learning can help to understand offline and online human behavior. Accuracy is an important metric for evaluating machine learning algorithms;

however, no single machine learning algorithm can appropriately solve everything. This means that an acceptable accuracy would be enough for certain problems. AI researchers usually explain why their prediction models are working but they say less about why their prediction models might not work. Decision-makers need to know about both of these aspects to make better decisions. For example, in high-risk situations, we need to provide a deeper understanding of the understudy situation [85]. In addition, usually, a good predictive model provides one or more prescriptions for potential future actions that enable decision-makers to make better decisions.

Predicting human behavior relies on both psychological studies and data science [75]. This means that social scientists and data scientists can learn from each other. Many social scientists focus on understanding and explaining the behavior without considering quantitative predictions. On the other hand, most data scientists avoid careful examination of human behavior, and they mainly concentrate on big data.

Several techniques have been generated by machine learning and reasoning under uncertainty for predictive statistical modelings such as decision trees, neural networks, classification and rule-induction methods, and Bayesian networks [110]. Chen et al. [13] proposed a data mining approach to model user intention in which proper concepts of linguistic features are extracted using rule association and classified with a Naive Bayes classifier. Guha et al. [35] deployed a user modeling system for Google Now personal assistant based on long-term user history with thousands of queries and clicks. They showed that identifying contexts such as user's interests and habits is critical to building a useful personal assistant.

In the context of smart environments, predicting future events is very important to improve daily living and help individuals to live independently. Also, human activity learning is critical for designing human-centric technologies and understanding human behavior [17]. Most of the research in prediction in smart environments can be grouped into two categories: predicting the activities

of daily living and predicting the location of events [94]. We can monitor residents' activities in smart environments and/or when they are using smart devices to evaluate the current state of health conditions and for detecting abnormal behaviors on an ongoing basis. For example, such smart environments and devices are used to analyze eating behavior and whether people are taking prescribed medication, or to detect periods of depression and anxiety and suggest interventions using computer-based therapy [88, 68]. Also, it is important to monitor activities of people with visual or mobility impairment to predict falls, detect unsafe activities and provide real-time responses for them [105, 49, 27, 67, 66].

To build such smart environments and devices, it is important to store and utilize long-term data to predict future health conditions in advance and to make the necessary arrangements proactively [69].

In many domains, we can collect enormous datasets by scraping the Internet and hiring people to label and clean those datasets by tools such as Amazon Mechanical Turk. However, it is significantly harder to gather such a dataset for a human-inhabited system in which humans physically interact with the environment such as smart-home. This is not just an issue of cost, but also many people do not allow collecting data of their daily living due to obvious privacy concerns. One of the most complete datasets for smart-home research is CASAS (Center for Advanced Studies in Adaptive Systems) dataset that encouraged many research ideas. The dataset is built from smart homes' data and is maintained publicly. Minor, Doppa, and Cook [64] trained activity predictors by CASAS dataset. Some studies focus on a current activity recognition task from sensory data in CASAS dataset [56]. Furthermore, Choi et al. [14] proposed two deep learning algorithms based on deep belief networks [53] and restricted Boltzman machines [52] to predict the behaviors of residents using MIT home dataset [87]. We can loosely divide the studies on these datasets into two categories. The first set of algorithms are classification problems in which the predictor model predicts the probability of the next event given the current state. The second category includes the

regression models that predict the particular time delays when the next event or activity occurs.

The intelligent process and prediction for IoT-based smart-homes has significant benefits; however, there are some challenges and concerns that are required to be considered [100]:

- Hardware limitation such as positions of sensors, cameras or other devices, lack of memory.
- Device connectivity such as standardize the communication protocols, different devices from manufacturers and companies.
- Data management to transfer sensor data from raw form into higher abstraction representations, trigger-responding actions, acquire daily life activities and human behavior, accessible and understandable to humans and interpretable by machines.
- Methods from machine learning and accuracy to provide excellent reasoning, enable decision making, understand the time of the day and seasonal changes, and analysis and prediction based on historical data.
- User behavior, lifestyle, and habit modeling to make sure to improve comfort and satisfaction.
- Security issues such as privacy of data, cyber-attacks.

Data collection and data management for user modeling. To predict user behavior or activities, we need to gather information from actuators and sensors while the user is interacting with them. Collecting this data allows us to create intelligent and automated systems. Many research groups collect data in real-world environments [26]. During data collection, a good indicator of cognitive and physical capabilities are activities of daily living such as sleeping, bathing, cooking, and so on. To create intelligent models from sensor data, we can use machine learning algorithms and train them on train and validation datasets to learn patterns and distributions. One challenge is that

real-world sensor-driven data collection is limited both in terms of availability and variety. Thus, we can use simulated or scaled models of smart environments to collect data [57, 25, 11, 62, 10]. These datasets can be used to train machine learning based predictive models that predict the next state of the environments based on the previous actions.

Also, synthetic data that can reflect the same patterns and human behavior can be generated [19]. For example, Dahmen et al. [20] generated synthetic data that is reflective of real-world data for activity learning and anomaly detection tasks in smart environments as they stated that anomalies are rare and are not well documented in real-world smart environment data. To create synthetic data, they used the structure of a hidden Markov model (HMM) [79] learned from the real smart home data.

Privacy for human behavior modeling and prediction. In this era of information, the privacy of users is the utmost important aspect. The protection of user privacy is one of the most worrying issues in IoT. Nowadays, devices are off-loading tasks to the cloud and transfer personal information as habits, media, or preferences [42]. Birchley et al. study the very important topic of ethics in smart environments [9]. They raise concerns about privacy, consent, social isolation, and equity of access. Most importantly, they are concerned about privacy. Even though significant endeavors had made to reduce the probability of unapproved data sharing, still privacy and data breaches happen all the time. Studies show that in the context of smart homes, privacy is the crucial obstacle to the adoption of technologies such as healthcare, activity automation, energy conservation, and remote access services [72]. There are many diverse methods to violate the privacy of the users. For example, Jiang et al. show that we can mine sensory data from motion sensors to infer the daily habits and health information of a related user [43].

Federated learning had initially been proposed as a technique to improve communication efficiency in distributed learning [50]. However, it had been pointed out that the technique also allows the

learning system to ensure differential privacy [31]. One of the early, high profile applications was Google's Gboard [37] which used federated averaging (FedAvg) [60] to improve next word prediction. In recent years, several research projects improved the performance and privacy characteristics of federated learning. Zhao et al. [108] suggested a data-sharing approach to improve the performance of the FedAvg algorithm in case the training data is non-IID. Wang et al. [90] aim to optimize learning of a gradient-descent based federated learning algorithm at the edge. In federated learning algorithms, local training happens at the edge and global aggregation is performed in a central place. They proposed a control algorithm that determines the best frequency of global aggregation with which computation and communication resources at the edge can be used efficiently in federated learning. Zhang et al. [107] proposed building trustworthy federated learning systems using trusted execution environments (TEEs). Their main focus was to assure that the local training on the clients' side is being done correctly.

Human cultural behavior and applications in pandemic prediction. During a pandemic, complete and accurate data is required to predict the changes in human behavior after applying different intervention plans and building mathematical models for simulating different scenarios. However, in case of epidemics, there is a scarcity of such data. The epidemic modeling community is much less mature than other modelling communities. Moran et al. studied the difference between epidemic forecasting and weather forecasting [65]. They suggested that in order to accurately predict a pandemic, it is important to model human behavior and the potential changes during a course of the pandemic, while these factors are generally not considered in short- and medium-range weather forecasts. Flaxman et al. [28] studied the effects of major non pharmaceutical interventions (NPIs) across 11 European countries and introduced a Bayesian model to estimate the epidemic. Their model calculates backwards from observed deaths to estimate transmission that occurred several weeks previously, allowing for the time lag between infection and death. More specifically, they fit their model to the observed deaths from the data. Also, the number of infections is estimated as

the product of R_t (reproduction rate) with a discrete convolution of the previous infections. R_t is a function of both initial R_0 (R_t before applying any NPI) and the effect from NPIs. Dehning et al. [22] focused on short-term infection forecasting based on NPIs and studied how the interventions affect the epidemiological parameters. They combined a SIR model with Bayesian parameter inference to analyse the time-dependent spreading rate. They detected the change points in the spreading rate that have correlations with the times of announced interventions. They specifically focused on Covid-19 spread in Germany. They showed that changes in the spreading rate affect the confirmed cases with a delay of about two weeks, with a median reporting delay of 11.4 days plus a median change-point duration of 3 days. Arık et al. [4] proposed an approach for modeling Covid-19 forecasts by integrating covariate encoding to compartmental models. They used the standard SEIR model but modeled more compartments such as: Undocumented infected and recovered compartments, Hospitalized, ICU and ventilator compartments, and Partial immunity (we do not what fraction of recovered population are immuned to future infection). Then, they incorporated a time-varying encoding of the covariates and trained an end-to-end model. Finally, they used multiple learning approaches to improve the generalization of their model such as: masked supervision from partial observations, partial teacher-forcing to minimize error propagation, regularization and cross-location information-sharing. Jin et al. [45] focused on a direct data-driven prediction model for predicting Covid-19 without using compartmental models.They developed a neural forecasting model called Attention Crossing Time Series (ACTS) that predicts cases by comparing patterns of time series from different regions. They addressed the scarcity of time series historical data for each region by investigating other regions' time series in the dataset with similar long term or short term patterns. Specifically, they considered certain time periods with similar dynamics rather than the entire time series. Therefore, the task here is to find small segments in reference time series that show similarity with the target time series. They suggested attention models to find these segments by adding trend filtering to model long-term trends that are difficult for the attention model to capture. Xiao et al. [95] proposed C-Watcher which is a data-driven framework to screen all

the neighborhoods in a city and detect the neighborhoods with the highest risk of Covid-19 spread before they contaminate other neighborhoods. They used long-term human mobility data from Baidu Maps and characterized each neighborhood by using urban mobility patterns. Their model is based on cross-city transfer learning with four components: 1. neural network for learning a representation of a neighborhood based on POI Radius, Demographic, and Transportation-Related features, 2. a discriminator component to identify whether the output of the encoder belongs to the target city or not, 3. the reconstruction component consists of two decoders for the epicenter city and the target city, and 4. a classifier. Liao et al. [55] proposed a time-window based SIR prediction model and used a machine learning method to predict the basic reproduction number and the exponential growth rate of the epidemic. For their time-window based sir model, they specifically split historical data into a time window segment in order to capture the real-time changes in R_0 and the exponential growth rate. They used Covid-19 historical data in China, South Korea, Italy, Spain, Brazil, Germany and France. Mehta et al. [61] focused on country level prediction of Covid-19 for the near future a combination of health statistics, demographics, and geographical features of counties. They used US Census data to obtain county-level population statistics for age, gender, and density. Their predictive model has three different outcomes: 1. the probability that a county has at least one confirmed case of Covid-19, which is defined as a positive instance, 2. the number of confirmed Covid-19 cases within a county which is defined as occurrences, 3. vulnerability of the country. They used a XGBoost classifier to classify each county either as a positive or negative instance. To predict the number of occurrences, they used a XGBoost regression model. Finally, they combined results from the first two stages and calculated the expected occurrences for counties as a measure of county vulnerability. Watson et al. [92] proposed a Bayesian time series model that fits a location-specific curve to the velocity (first derivative) of the log transformed cumulative case count. Then they use a random forest algorithm that learns the relationship between Covid-19 cases and population characteristics to predict deaths. Finally, they embed these models to a compartmental model which can provide projections for active cases and confirmed recoveries. They

obtained the data from Covid-19 Tracking Project which is a combination of information from state health departments and other sources. Zou et al. [109] introduced the SuEIR model, a variant of the SEIR model, to predict confirmed and fatality cases, the peak date of active cases, and estimate the basic reproduction number (R_0) in the United States. Their model considers additional information such as the untested/unreported cases of Covid-19 and is trained by machine learning based algorithms by using historical data. They fit an ordinary differential equation (ODE) based model on the data. Their model could provide accurate short-term (daily ahead) projections for both confirmed cases and fatality cases at national and state levels. In the long term, they showed that the numbers of confirmed cases and deaths will keep increasing rapidly within one month.

Qian and Alaa [76] focused on developing a model to learn the policies that affect the fatality rate of the Covid-19 in a global context. They used a Bayesian model with a two-layer Gaussian process (GP) prior. The lower layer models the Covid-19 fatality curve over time within each country with a compartmental SEIR (Susceptible, Exposed, Infectious, Recovered) model. The upper layer is shared with all countries and it is another GP model that learns the R_0 as a function of country features and policy indicators. They used data that contains all economic, social, demographic, environmental and public health features. Sharma, Mindermann, Brauner et al. [81] investigated the robustness of the estimated effects of NPIs against Covid-19. In particular, they studied if NPI effectiveness estimates generalize to unseen countries without access to the ground truth NPI effectiveness estimates. They also studied numerous assumptions and limitations for the data-driven estimate models and suggested that the policy-makers should make decisions based on diverse sources of evidence, including other historical studies, experimental methods, and clinical experience. Mastakouri and Schölkopf [59] studied a causal time series analysis of the Covid-19 spread in Germany to understand the causal role of the applied non pharmaceutical interventions (NPIs) in containing the spread among German regions. They used a causal feature selection method for time series with latent confounders called SyPI to analyse and detect the restriction policies that

have a causal impact on the daily number of Covid-19 cases. They performed the analysis on a state and on a district level. Yeung et al. [99] combined NPIs and Hofstede cultural dimensions in predicting the infection rate for 114 countries. Particularly, they predict confirmed infection growth (CIG), which is defined as the 14-day growth in the cumulative number of reported infection cases. They used OxCGRT data of the NPIs and trained different non-time series models such as ridge regression, decision tree, random forest, ada boost, and support vector regression using mean squared error (MSE), and performed a grid search on the combination of these models. Their results showed that random forest regression and AdaBoost regression were the best performing predictors out of the five evaluated machine learning models. Johndrow et al. [46] built a model for Covid-19 transmission only by using the number of daily deaths, timing of containment measures, and information on the clinical progression of the disease. The authors suggested that using less precise information such as the number of confirmed cases will result in unreliable analysis. Their modeling approach is a SIR model of disease spread via a likelihood that accounts for the lag in time from infection to death and the infection fatality rate. They specifically fit a Markov Chain Monte Carlo (MCMC) algorithm to their data. Bengio et al. [8] proposed a proactive contact tracing method. They embedded two neural networks namely Deep Sets and Set Transformers and evaluated the resulting models via the COVIsim testbed. Their methods are able to leverage weak signals and patterns in noisy, heterogeneous data to better estimate infectiousness compared to binary contact tracing and rule-based methods. Their method provides a good trade-off between restrictions on mobility and reducing the spread of disease. Davis et al. [21] designed a stochastic age-structured transmission model to study different intervention scenarios and their impacts on the transmission of Covid-19 cases in the UK. They specifically explored four base interventions scenarios of school closures, physical distancing, shielding of people aged 70 years or older, self-isolation of symptomatic cases, and modelled the combination of these interventions.

CHAPTER 3: HUMAN BEHAVIOR PREDICTION FOR CONTENT CACHING

In a possible future, high-quality and visually-compelling contents such as pervasive augmented and/or virtual reality or 3D immersive experiences might become the primary type of information to deliver to the users. Max reality, for example, is a product of the weather company, an IBM business aiming to deliver personalized and more interactive visualization of weather or traffic reports to the users to better understand travel conditions and plan for their commute. Their studies showed that max reality appealed to 62% of participants, and 64% will stay tuned in longer if max reality is coming up in the next segment. To achieve a high level of user satisfaction, such a system must answer user requests with the best quality and with minimal lag. As bandwidth and latency limitations will still apply, the system must perform predictive caching of the content. In this paper, we investigate several strategies for predicting the information needs of a user in a smart home. The paucity of datasets is a major challenge in such studies. We are starting from the hypothesis that the user's patterns of daily life guide the content consumption regardless of the delivery medium. This allows us to synthetically generate realistic content requests starting from real-world databases of user activities in smart homes.Using these datasets, we develop techniques for demand prediction and content caching that aim to optimize the quality of user satisfaction while minimizing the cost of caching.We propose three algorithms: one based on probabilistic modeling, one based on long short term memory (LSTM) networks, and one based on majority voting. Through a set of experimental studies, we show that our techniques outperform baseline caching techniques both in terms of user satisfaction and caching cost.

User modeling

Problem statement

In this paper, we consider a set of household scenarios in which residents request experiences such as (a) summary of the news, (b) weather report, (c) parking availability report, (d) traffic report, and (e) food recipe. Statistically, a given type of experience is more likely to be accessed at a specific time of the day. For instance, a user might more likely need a recipe during dinner preparation time, or a weather forecast can be most helpful before leaving home. The users go through a series of interactions split into experience units, a particular short interaction with the system. In our scenarios, we consider experiences that each "unit of experience" is approximately 15 to 20 seconds long and can be delivered at different "level of quality" from high such as 4K videos or 3D animation to low such as 3GP low-resolution QCIF. For instance, a weather forecast can be delivered as a short text message, a dynamic 3D image on a big screen, or a dual-4K immersive visualization. Providing a high-quality experience requires both networking and computing power. Therefore, it is limited by the (1) capabilities of the devices through which it is delivered and by the (2) signal limitations such as network delay and bandwidth. Table 3.1 shows the quality levels we consider in this paper and the size of the data chunk necessary to deliver the experience unit.

Considering a particular experience e, we define the delay by dividing the size of content in MB by external bandwidth:

$$delay_e = \frac{size(e)}{external\_bandwidth}. \tag{3.1}$$

One customary approach to rate the user satisfaction would be by evaluating objective metrics such as video definition (video quality), fluency (interruptions), response speed (initial delay) [47].

Table 3.1: Relative quality and caching cost levels of experience units and data chunk size for a 15 to 20 second experience unit. To compute the relative cost, we consider the worst case for each type of format.

Type	**Size of exp. unit**	**Relative quality**	**Relative cost**
4K video	32.7 MB	1.00	205.66
HD video (1080p)	10.6 MB	0.90	66.67
low-res video MKV	483 KB	0.81	3.04
3GP low-res QCIF	159 KB	0.73	1.00
sound-only	-	0.66	-
text-only	-	0.59	-
3D animation	2 MB - 20 MB	1.00	125.79

According to this, we estimate the "user satisfaction" by the score as follows:

$$score_e = d_d^{delay_e} \cdot d_f(e) \cdot max\_score \tag{3.2}$$

in which d_d is the discount factor for delay, $d_f(.)$ is the discount factor of quality of each experience shown in Table 3.1 (column of relative quality), and max_score is the value for the maximum quality. Notice that the value of d_d and $d_f(e)$ are between 0 and 1. In other words, greater delay or lower quality both result in smaller score value.

Other than user satisfaction, we have to consider the cost of caching. We calculate the cost of caching as $cost = n_c \cdot r_c$, where n_c is the number of cached items and r_c is the relative cost of each type of content which are available in Table 3.1 (column of relative cost). Since the size of sound-only and text-only formats are very small, we do not consider cache cost for these types of content. We obtain relative cost by $\frac{size}{min\_size}$ in which $size$ is the content size for a 15 to 20 second experience unit based on type, and min_size is the minimum content size which is for 3GP low-res QCIF type and is equal to 159KB. Eventually, the final score is computed as

$$final\_score = \alpha \cdot score - \beta \cdot cost \tag{3.3}$$

where α and β are coefficients for score and cost value, respectively.

Modeling the users' interaction with devices

As we discussed earlier, our objective is to maximize the quality of the experiences for the user, by predicting the time when specific experiences will be requested and by using this information for efficient predictive caching. The prediction of the requests is ultimately rooted in the regularities of everyday life. For instance, a user typically asks for a weather report in the morning before leaving the house. However, we need to be aware that such predictions are inevitably probabilistic. In a given day, the same user might ask for the weather report in the evening, or not ask for it for several days. As user preferences are highly specific to the given user and household, we propose techniques through which they can be learned from actual user data, as opposed to engineering a user model from first principles.

One of the challenges of such an approach is the lack of existing datasets for this kind of requests. To the best of our knowledge, no extensive data of such requests is yet available. However, the design of the system would need exactly such data to learn the user model. To solve this chicken-and-egg problem, we propose to generate training data starting from daily user behavior datasets already acquired in smart homes and augmenting/extending these datasets with logical assumptions about when the user's would have requested experiences, should they have been available.

Real world and simulated datasets of user activities in homes

The emergence of sensor-augmented smart homes made it possible to acquire datasets that track certain aspects of the inhabitants' behavior in the last several years. In general, tracking the personal life of users opens serious privacy issues. However, several projects captured and made

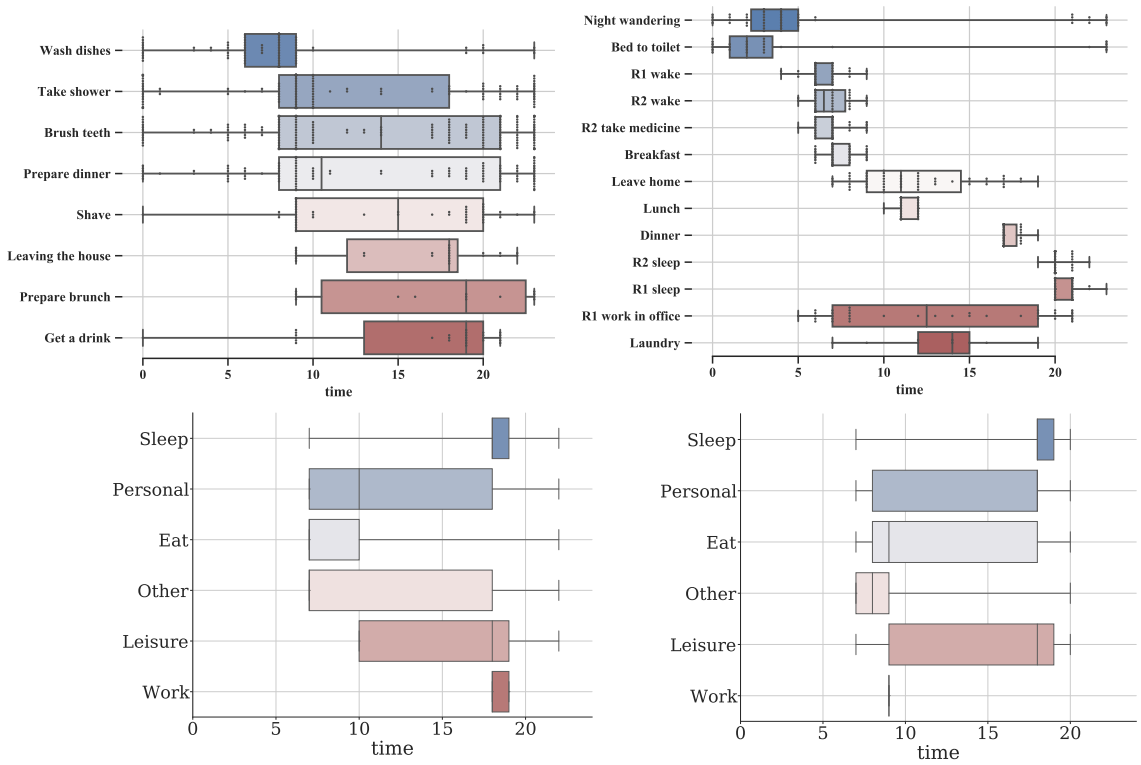

Figure 3.1: The action distributions for each day. The x-axis shows time in hours during a day of Real-world Dataset 1 (top-left), Real-world Dataset 2 (top-right), Simulation Dataset 1 (bottom-left), Simulation Dataset 2 (bottom-right).

publicly anonymized datasets of human behavior in the home, tracking project-specific collection of actions. In addition to real-world dataset of daily activities in a smart home, in order to facilitate the dataset building, simulation environments are designed as a smart home by which the users can simulate their daily activities and collect these actions. While these actions (in both real or simulated environments) might not directly map to experience requests, they can anchor the generation of training data.

Real-world Datasets

In this paper, we started from two publicly available real-world datasets of Activities of Daily Livings (ADLs) in two different homes:

Dataset 1: WSNs One of the most promising technologies in this domain are wireless sensor networks (WSNs), due to their low cost, flexibility, ability to supply constant supervision, and inherent non-intrusive characteristics (as compared to video-based supervision). However, WSNs cannot gather as much information about user contexts as other sensing systems, such as video cameras. Obtaining a good body of labeled data is difficult. Users are reluctant to write down their activities because it is time-consuming and can compromise their privacy. The dataset [48] describes the activities of a 26-year-old man in a smart home with 14 state-change sensors installed at doors, cupboards, the refrigerator, and the toilet flush. Sensors were left unattended, collecting data for 823 data points of 28 days activities in the apartment. Eight annotated ADLs included shave, brush teeth, get a drink, get dressed, prepare for leaving, prepare brunch, and prepare dinner. We can see the list of performed actions and their distribution during the day in Figure 3.1 (top-left). The x-axis describes time intervals from hour 0 to 24 during a day. We also show the names of the eight tasks on the y-axis.

Dataset 2: CASAS This dataset [18], collected by CASAS (Center for Advanced Studies in Adaptive) research group, describes the activities of two residents in an apartment for 1199 data points of 57 days activities. This dataset contains sensor data that was collected in the home of a volunteer adult couple. Residents R1 and R2 can do different tasks in the house. Annotated actions in this dataset include night wandering, bed to toilet, R1 wake, R2 wake, R2 take medicine, breakfast, leave home, lunch, dinner, R1 sleep, R2 sleep, R1 work in office, and laundry. Figure 3.1 (top-right) visualizes dataset 2 by showing each task's frequency in specific time slots. The x-axis describes time intervals from hour 0 to 24 during a day and the y-axis shows the names of the 13

tasks (R1 and R2 are the residents).

Simulated Dataset: Open Smart-Home simulated (OpenSHS)

Since the number of data points in real-world daily activity datasets is limited, we also used simulated datasets of everyday activities collected from the OpenSHS environment (Open Smart Home Simulator) [2]. OpneSHS is an open-source, cross-platform 3D smart home simulator for dataset generation. Data collection in this simulation environment requires two entities: the researcher and the participants. The researcher designs the environment, import the devices and sensors and assigns activities' labels. The researcher is also responsible for designing contexts such as morning, evening, weekday or weekends. Participants performed the Activities of their Daily Livings (ADLs) for different contexts in a week, e.g., weekdays, weekends, and in a day e.g., mornings and evenings in this simulation environment [3]. The actions include work, eat, sleep, leisure, personal, and other. We chose two out of seven simulated datasets for our experiments.

Dataset 1 Two months of activities simulated and collected in this dataset with time-margin equal to 0. This dataset contains 77 328 data points. See Figure 3.1 (bottom-left).

Dataset 2 Similar to dataset 1, dataset 2 contains two months of activities simulated and collected in this dataset with time-margin equal to 0. This dataset contains 100 544 data points. See Figure 3.1 (bottom-right).

Creating synthetic datasets using common-sense association

We create realistic synthetic datasets of the user's requests from the system using datasets of the users' daily activities in their home. We generate our synthetic scenarios by matching the statistical properties of the real-world and simulated datasets. We probabilistically associate certain

Table 3.2: Mapping approach from daily task to daily request of the users for real-world dataset 1 (top), real-world dataset 2 (middle), and simulated dataset 1 and 2 (bottom)

task (Real-world Dataset 1)	**corresponding request**
Shave, Brush teeth, Get a drink	Summary of news
Get dressed, Prepare for leaving (30% of the times)	Weather report
Prepare for leaving (50% of the times)	Traffic report
Prepare for leaving (20% of the times)	Parking status
Prepare brunch, Prepare dinner	Recipe

task (Real-world Dataset 2)	**corresponding request**
R1 wake, R2 wake	Summary of news
Breakfast (70% of the times), Leave home (30% of the times)	Weather report
Leave home (50% of the times)	Traffic report
Leave home (20% of the times)	Parking status
Breakfast (30% of the times), Lunch, Dinner	Recipe

task (Simulated Datasets 1 and 2)	**corresponding request**
Other (50% of the times), Leisure(60% of the times)	Summary of news
Other (15% of the times), Work (30% of the times)	Weather report
Other (10% of the times), Work (50% of the times)	Traffic report
Other (5% of the times), Work (20% of the times)	Parking status
Other (20% of the times), Leisure(40% of the times), Eat	Recipe

experiences with activities that are present in the dataset using common-sense associations. Creating synthetic datasets using common-sense has been applied to different problems in machine learning, such as Question-Answering challenges [86] and visual reasoning [106].

The mappings that we used to create the synthetic data are shown in Table 3.2. In these mappings, we assumed that, for example, the user is likely to request a recipe while preparing food. Also, it is more probable to check the weather news, traffic report, or parking status before leaving home or while the user is getting dressed. We assign a probability of occurrence for each mapping from activity or task in the real and simulated datasets to the corresponding request. These probabilities are also set based on common-sense. For example, people usually leave their homes to work in the

morning. So, it is more likely that they want to see the weather report while eating or preparing breakfast rather than watching a breakfast recipe.

Methods

In this section, we describe our design for a predictive controller, which, knowing the preferences and habits of the user, makes intelligent decisions about what to cache. Also, we describe three caching algorithms as the baselines caching methods.

Predictive Caching Algorithms

Implemented caching strategies for the proposed controller are as follows[1]:

Probability-based caching

We define 24 intervals with the length of 1-hour for each day in the datasets. The proposed algorithm is based on calculating the probability of a specific request in a specific time interval by counting the data occurrences in the training set. Accordingly, requests with a probability higher than a threshold are cached for each interval in each day. We validate this approach on the different threshold for the number of request occurrence in a specific time interval, then the best result on the validation dataset is applied to the test dataset.

[1]The code is available here: https://github.com/sharare90/AR-VR-Research

LSTM-based caching

We propose the LSTM-based caching algorithm, based on training a Long Short Term Memory (LSTM) [39] recurrent neural network on the training dataset. We tried two different approaches for implementing the LSTM model: (i) many-to-one and (ii)many-to-many. The LSTM models are shown in Figure 3.2 and 3.3.

Input data is a sequence of requests. We divide a day to 24 intervals, each with a fixed list of requests shown with 0s and 1s. We have N different type of requests: $\{r_1, r_2, ..., r_N\}$. The data for each interval is a vector x of length N, and N is five in our experiments since we assumed that we have five types of requests (summary of news, weather report, parking status, traffic report, food recipe). The value of element $x_i (i \in \{1, 2, ..., N\})$ is 1 if the request r_i has occurred in the interval, otherwise its value is 0. Furthermore, the number of classes equals the number of valid requests.

Many-to-one LSTM based prediction: In this LSTM based prediction, the input is T = 24 hours history of request actions, and the output is the request of the next interval (Figure 3.2). For example, if we need to predict the action of the user during the time interval of 3:00 PM - 4:00 PM, the sequence of input intervals will include 3:00 PM - 4:00 PM yesterday, 4:00 PM - 5:00 PM yesterday, and so on until 2:00 PM - 3:00 PM today. The LSTM takes these intervals sequentially as input and updates its hidden state based on this sequence. When all the intervals are processed, the LSTM layer outputs a vector, and that vector goes into the following fully connected and dropout layers. The dropout layers, in turn, help with generalization and are only active during training. In other words, they mask a part of input such that the network learns to predict the output from a partial input. As a result, the network remains impartial to just one particular element in the output vector of the LSTM. During testing, dropout layers are disabled. In other words, they act like identity function and pass their input without masking any element to the next layer. Finally, the classification layer with as many neurons as the number of activities predicts the probability of

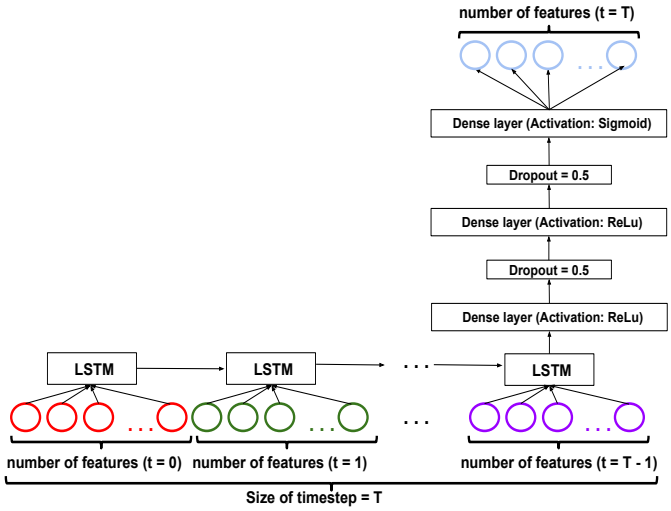

Figure 3.2: The many-to-one neural network used in the LSTM-based caching algorithm.

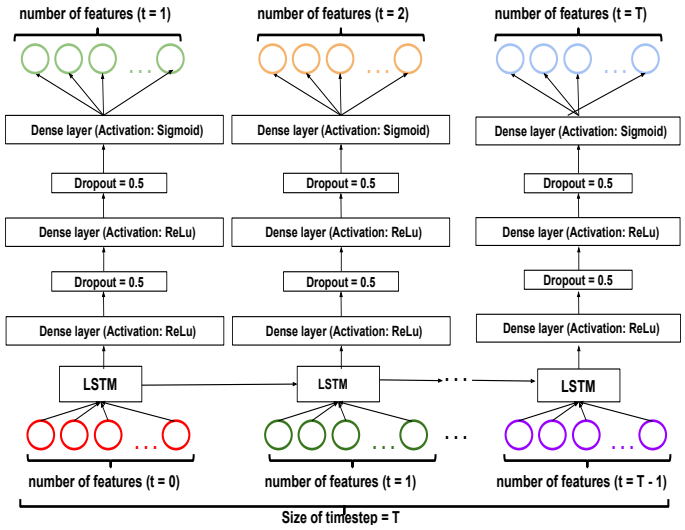

Figure 3.3: The many-to-many neural network used in the LSTM-based caching algorithm.

each activity by a number between 0 and 1.

Many-to-many LSTM based prediction: In this LSTM based prediction, network processes each interval vector one at a time and outputs the occurrence probabilities of requests for the next

Table 3.3: Selected values for hyperparameters of majority vote-based prediction.

Hyperparameters	**Values**
learning rate	0.001, 0.01
number of epochs	225, 300, 500, 1000
number of dense layers	2, 3
regularization method	dropout (0.0, 0.2, 0.5, 0.8), l1 and l2

interval (Figure 3.3). In the first interval of the day, all the input values are 0. We then concatenate this zeros vector with the lists of requests for interval 1 to 23, so we have 24 lists of requests for T = 24 intervals of the day as the input of the network. The difference with the many-to-one approach is that instead of processing all the intervals first and then output a vector, the LSTM layer outputs a vector as soon as it receives the first interval and outputs another vector once it receives the second input interval, and so on. These vectors then go into dense and dropout layers, and for each of them, we predict the activity of the next interval. For example, at the time interval of 12:00 AM - 1:00 AM, the LSTM predicts the action in time interval 1:00 - 2:00 AM and also updates its internal hidden states. After receiving the ground truth activities of what actually happened in time interval 1:00 - 2:00 AM and based on its updated hidden states, it predicts the activities in the next time interval, which is 2:00 AM - 3:00 AM, and so on.

Majority vote-based caching

Majority voting is one of the basic prediction/classification methods in which multiple classifiers are used to predict the label based on the majority vote of the classifiers [23, 34]. We create N different LSTM models by altering hyperparameters such as learning rate, number of epochs, number of layers, or changing regularization method (dropout or l1 or l2 regularization), initial weights, and so on. See Table 3.3 for majority voting hyperparameters. Subsequently, we predict the label value: $\hat{y} = mode\{\hat{y}_1, \hat{y}_2, ..., \hat{y}_N\}$.

Baseline Caching Algorithms

In this section, we explain the three baseline predictive caching algorithms.

Oracle

This baseline method is a caching algorithm where we assume that the algorithm can predict the requests with 100% accuracy. The final score that we can achieve with this algorithm is the highest score that we can have in each experiment.

Cache everything

This baseline algorithm suggests to cache every possible experience, ensuring that every experience can be delivered with a $delay = 0$. However, caching everything has a high redundancy, and since we have $cost = 1$ for each cached request, the caching cost increases.

Random caching

In this basic strategy, we assume that we do not have any prior knowledge about the request, and thus we cache a randomly chosen request from the pool of possible requests.

Experimental Approach

In this section, we describe a series of experiments that we have done on real and simulated datasets to predict the user's next request from audio-visual devices. We compared the three predictive

caching algorithms that we propose to three baseline algorithms in this series of experiments. Thus in the remainder of this section, we refer to the following six caching algorithms:

- Probability-based caching.
- LSTM-based caching.
- Majority vote-based caching.
- Oracle.
- Cache everything.
- Random caching.

Furthermore, we are interested in two performance metrics:

- Prediction accuracy as measured by the F1-score for the training phase on different datasets.
- The *final_score*, that combines the cost and user's satisfaction (Equation 3.3). The less latency that the agent has, the more satisfied the user would be. Also, the more optimized caching translates to the less costly predictive model. There is a cost for caching whether the user has a request or not, but caching before the user's request, will give the user more satisfaction, which should avoid the extra cost for missing requests.

Prediction accuracy

We described in Section 3 how the input matrix to LSTM is constructed. It is important to notice that the data is unbalanced, which means the number of 0s is much more than the number of 1s

in constructed matrices. Therefore, accuracy is not a good metric for evaluation since it could be very high even if the network predicts only 0s. Thus, we report our results by F1-score alongside with precision and recall to handle datasets unbalances in terms of requests. Precision shows how many predicted requests are relevant to the user's real requests in a specific time interval. On the other hand, recall demonstrates that our predictor could predict how many users' real requests are in a specific time interval.

Long-short term memory network

The experimental results for trained many-to-one LSTM based predictive agent are shown in Table 3.4 for real-world and simulated datasets 1 and 2.

As expected, there is a gap between test and train accuracy in real data. We see that the gap is minimal for simulated data, and sometimes test accuracy is even better. The reason is that usually in simulated datasets, the distribution of test data is very close to the distribution of data during training, and that is because simulated datasets are not as complicated as real datasets in nature. Nevertheless, our method is effective in both scenarios. We show that this is due to data scarcity and having bigger real datasets can help decrease the gap between train and test accuracy. Figure 3.4 shows that by increasing the number of days of data collection in a real situation, the F1-score on validation is approaching the train F1-score. Therefore, the current gap between train F1-score and validation F1-score is a variance error which can decrease by training over a bigger dataset.

Figure 3.5 shows how train and validation F1-score grows by increasing the number of epochs on real-world datasets 1 (top) and 2 (bottom) for many-to-one architecture. Figure 3.6 shows the many-to-one LSTM F1-score on the training and validation data for simulated dataset 1 (top), and simulated dataset 2 (bottom) based on history of activities with size of 24 hours using patience 70 and time interval length = 1 hour. The graphs of the evolution of the F1-score correspond to

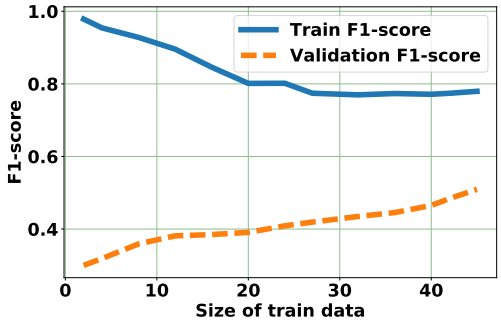

Figure 3.4: Train and validation F1-score by increasing the size of data.

the prediction accuracy based on LSTM method. In training an LSTM network, the precision, recall, and F1-score on train and validation data are calculated after each epoch. By plotting these figures, we can figure out the best time to stop training. For example, if the accuracy of validation is not changing more than a threshold after 70 epochs (patience=70), we can stop training to avoid overfitting.

We also monitor the evaluation metrics during the training and report them for trained many-to-many LSTM based predictive agent. Table 3.5 shows the results for real-world datasets 1 and 2 and simulated datasets 1 and 2 for many-to-many architecture. As expected, the gap between validation and training is more significant within real-world datasets than simulated ones.

Figure 3.7 shows the many-to-many LSTM F1-score on the training and validation data for real-world 1 (top), real-world 2 (bottom), and Figure 3.8 shows the many-to-many LSTM F1-score on the training and validation data for simulated dataset 2 (top), and simulated dataset 2 (bottom) after 225 epochs and time interval length = 1 hour.

We see that validation accuracy has plateaued, indicating the need to stop training to avoid overfitting.

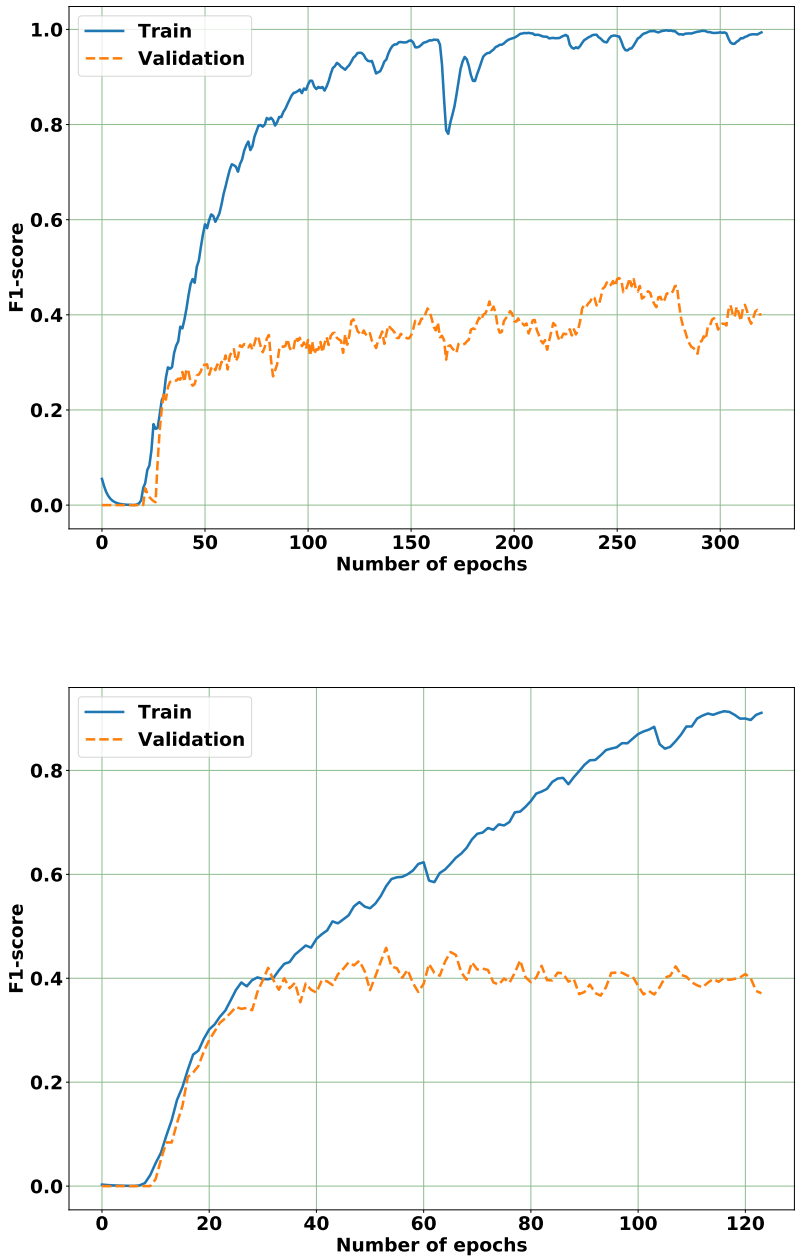

Figure 3.5: The F1-score of the prediction, using many-to-one LSTM model for the train (blue) and validation (orange) of real-world 1 (top) and real-world 2 (bottom) based on history of 24 hours by using patience 70 and 1 hour time interval.

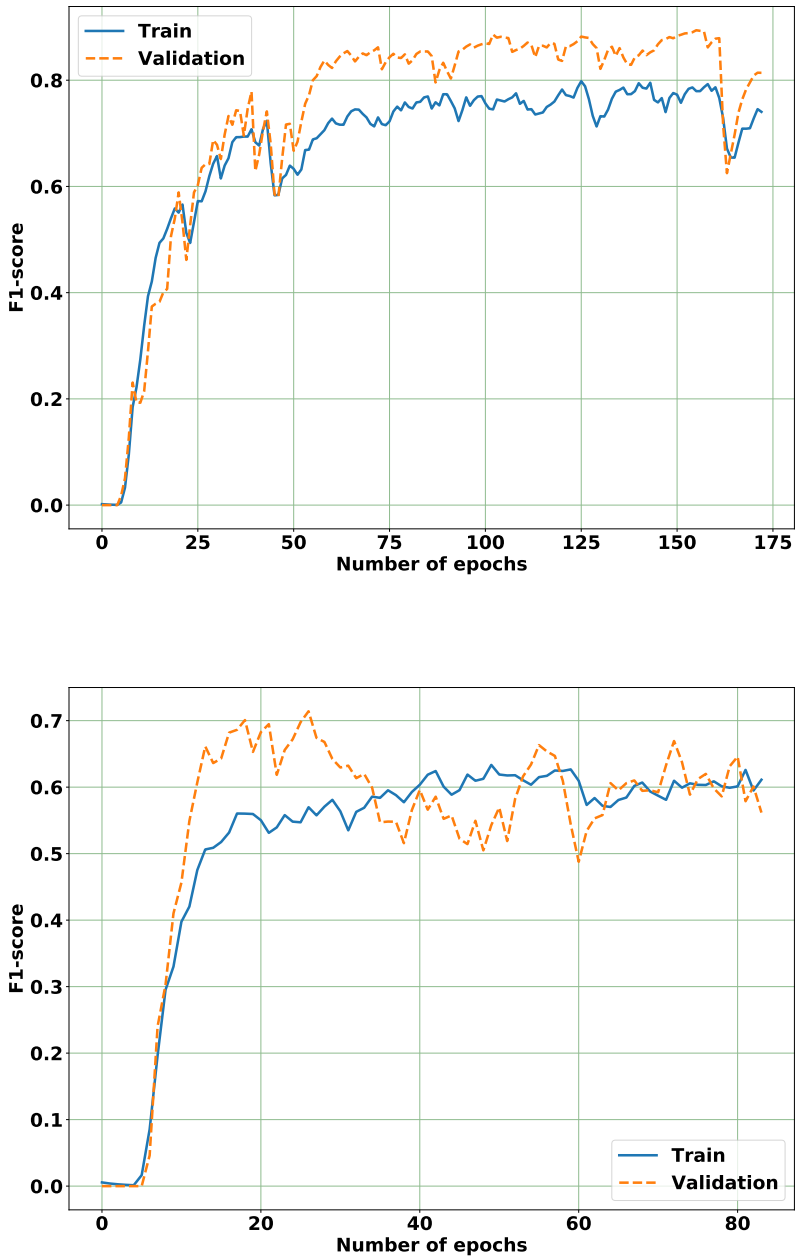

Figure 3.6: The F1-score of the prediction, using many-to-one LSTM model for the train (blue) and validation (orange) of simulated dataset 1 and (top), and simulated dataset 2 (bottom) based on history of 24 hours by using patience 70 and 1 hour time interval.

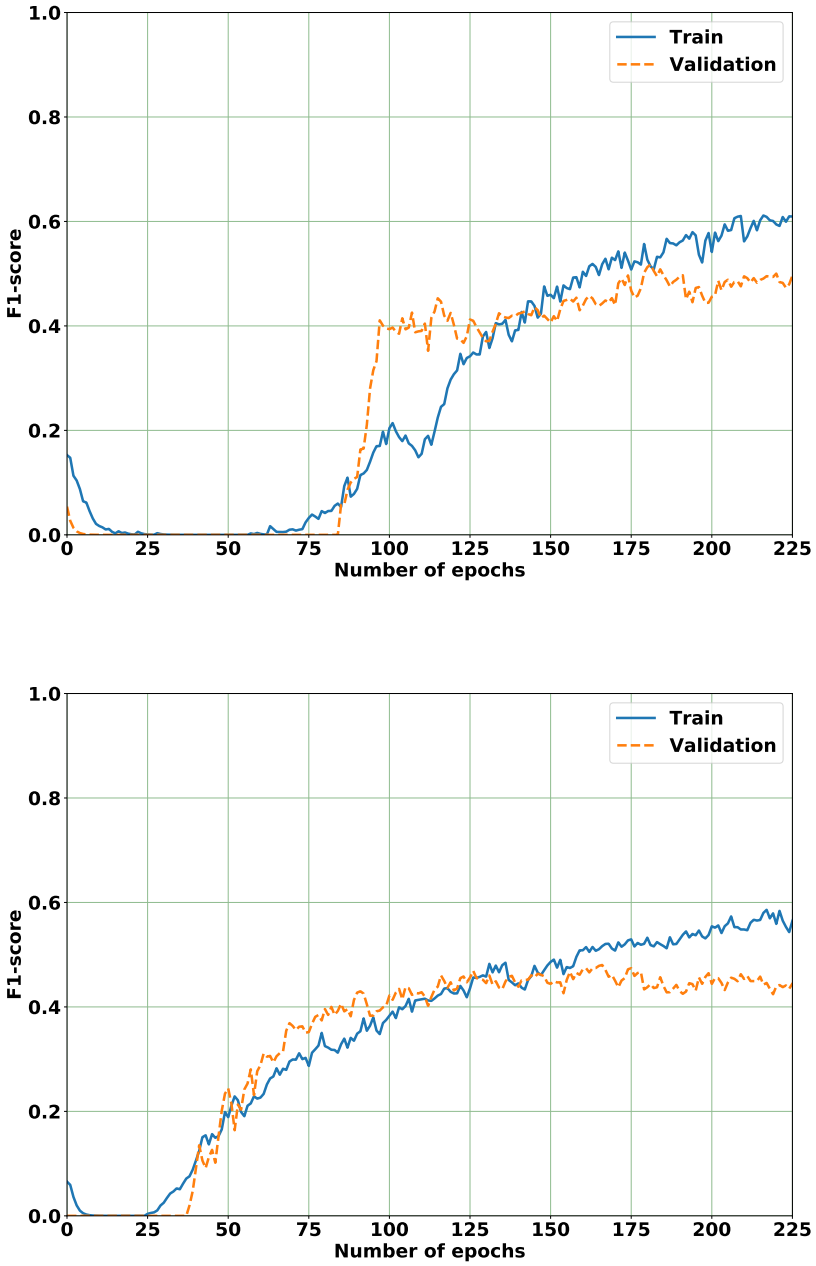

Figure 3.7: The F1-score of the prediction, using the proposed many-to-many LSTM model for the train (blue) and validation (orange) of real-world dataset 1 (top), real-world dataset 2 (bottom) after 225 epochs and time interval length = 1 hour.

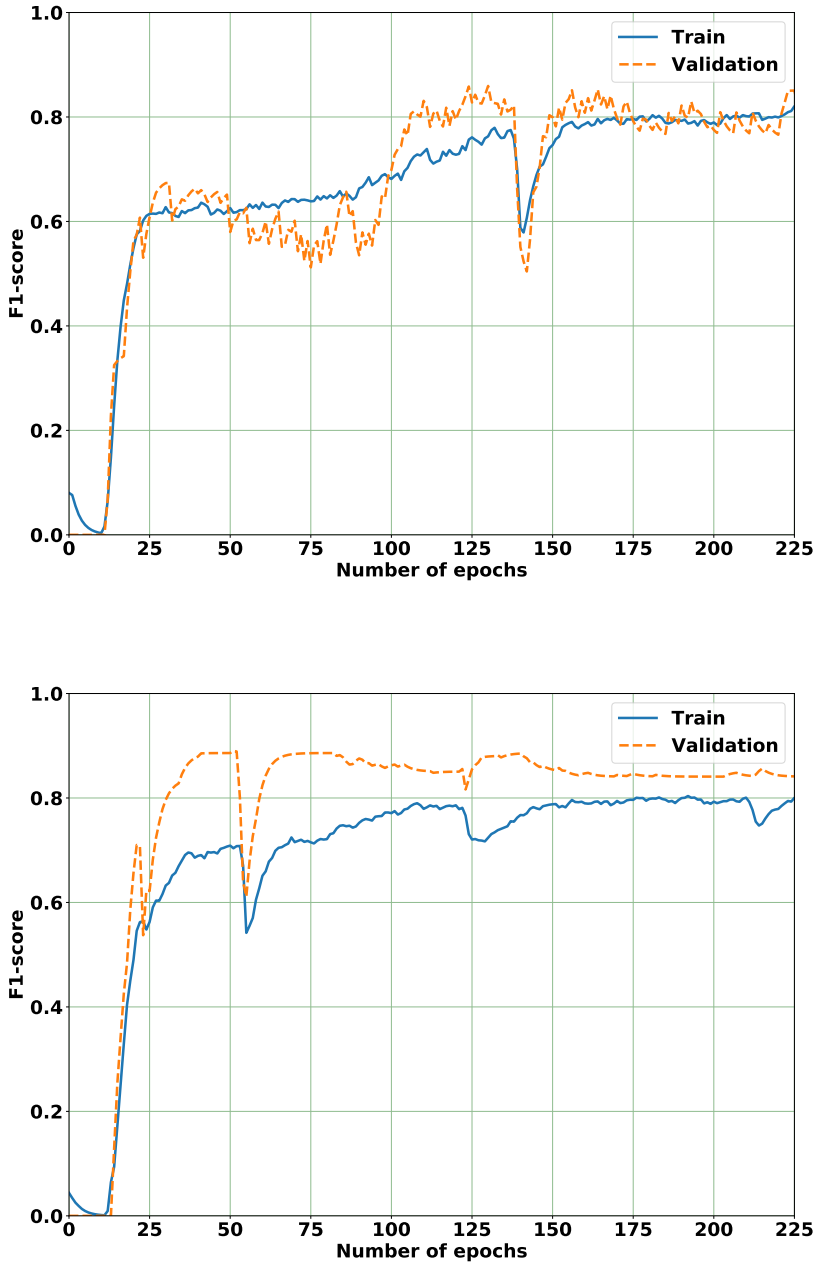

Figure 3.8: The F1-score of the prediction, using the proposed many-to-many LSTM model for the train (blue) and validation (orange) of simulated dataset 1 (top), and simulated dataset 2 (bottom) after 225 epochs and time interval length = 1 hour.

Table 3.4: F1-score of the prediction, using the many-to-one LSTM model on real-world dataset 1, real-world dataset 2, simulated dataset 1, and simulated dataset 2 based on history of activities with size of 24 hours by using early stopping with patience 70.

Results for real-world dataset 1	**Train**	**Validation**	**Test**
Precision	1.00	0.33	0.13
Recall	1.00	0.50	0.12
F1-Score	1.00	0.40	0.13

Results for real-world dataset 2	**Train**	**Validation**	**Test**
Precision	0.93	0.36	0.61
Recall	0.92	0.40	0.43
F1-Score	0.92	0.36	0.47

Results for simulated dataset 1	**Train**	**Validation**	**Test**
Precision	0.70	0.82	0.81
Recall	0.80	0.85	0.98
F1-Score	0.73	0.81	0.88

Results for simulated dataset 2	**Train**	**Validation**	**Test**
Precision	0.71	0.87	0.73
Recall	0.65	0.39	0.62
F1-Score	0.65	0.47	0.65

Experimental results of the overall predictive caching agent

It is common to rate the user satisfaction by objective metrics such as video definition (video quality), fluency (interruptions), response speed (initial delay) [47]. If the user sends a request and we have the requested content cached in the system, we increment the score by using Equation 3.2 and the $d(e) = 0$ since there is no delay in this case. However, if a user requests non-cached content, we need to load it with a delay that depends on the size of the content and external bandwidth. We consider $external\_bandwidth = 100Mbps$ for the experiments in this paper. We also normalize the results with a max-min normalization.

We decided to use many-to-many LSTM prediction for the LSTM-based caching experiments since

Table 3.5: F1-score of the prediction, using the many-to-many LSTM model on real-world 1, real-world 2, simulated dataset 1, and simulated dataset 2 after 225 epochs

Results for real-world dataset 1	**Train**	**Validation**	**Test**
Precision	0.69	0.62	0.52
Recall	0.54	0.52	0.31
F1-Score	0.61	0.54	0.39

Results for real-world dataset 2	**Train**	**Validation**	**Test**
Precision	0.72	0.54	0.54
Recall	0.55	0.43	0.46
F1-Score	0.62	0.47	0.50

Results for simulated dataset 1	**Train**	**Validation**	**Test**
Precision	0.84	0.87	0.90
Recall	0.81	0.72	0.82
F1-Score	0.82	0.79	0.83

Results for simulated dataset 2	**Train**	**Validation**	**Test**
Precision	0.79	0.77	0.89
Recall	0.80	0.93	0.88
F1-Score	0.79	0.84	0.86

the prediction results are more promising for this architecture.

Figures 3.9 and 3.10 show the scaled final score for each caching algorithm which are calculated based on Eq 3.3 on real and simulated datasets, respectively. The *final_score* for the majority voting and LSTM-based approach is almost better than other approaches. However, for the lowest quality of delivery, such as low-res video MKV, we do not see that much difference between approaches. The final results for 3GP low-res QCIF are mostly near 0 since both user satisfaction and caching costs are around 0. Furthermore, the prediction accuracy is higher when we have a bigger dataset with more data points (see Figure 3.10).

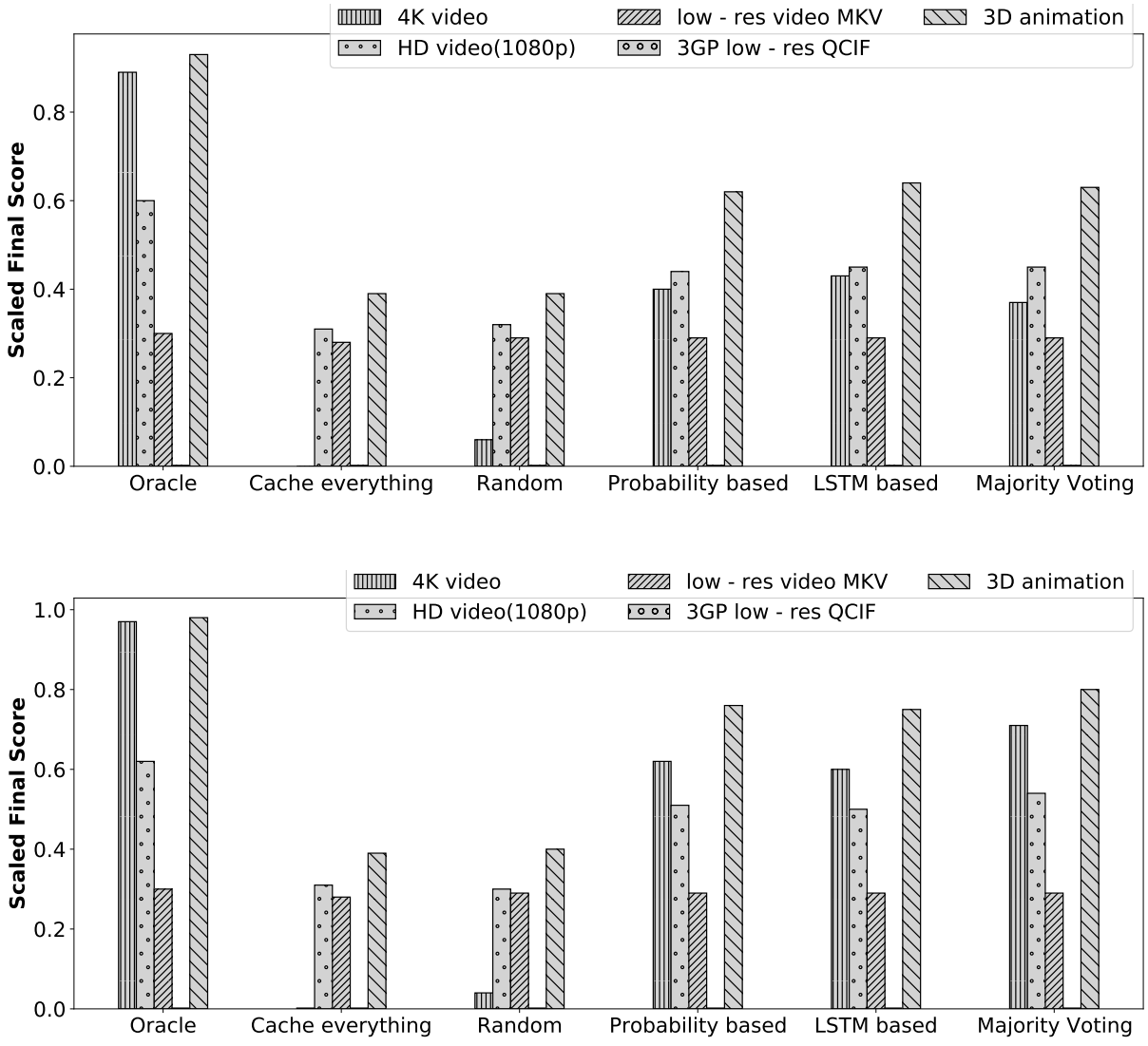

Figure 3.9: Caching approaches scaled final score (Eq 3.3) results on real-world dataset 1 (top) and real-world dataset 2(bottom) for each delivery format.

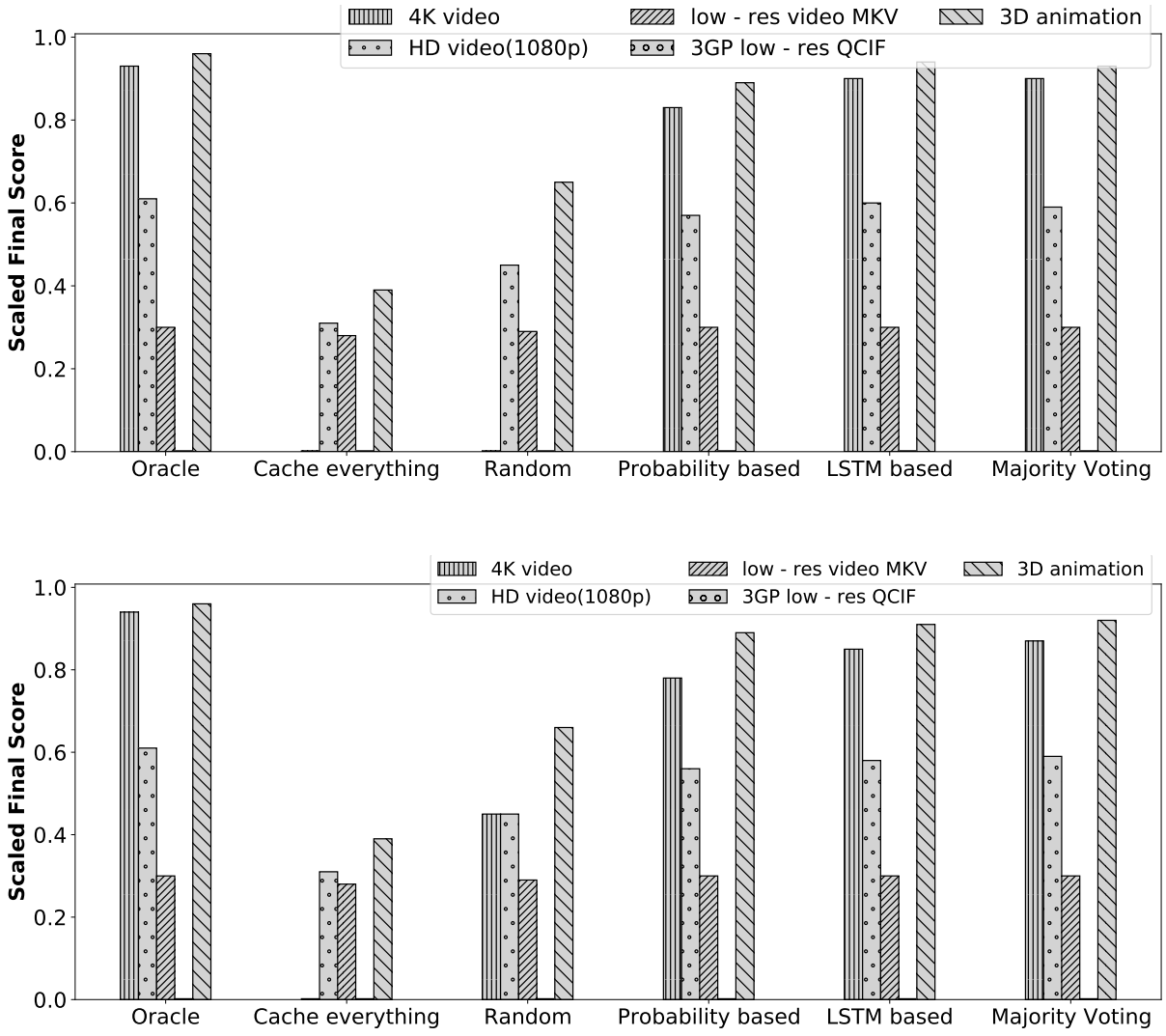

Figure 3.10: Caching approaches scaled final score (Eq 3.3) results on simulated dataset 1 (top) and simulated dataset 2 (bottom) for each delivery format.

CHAPTER 4: PRIVACY-PRESERVING LEARNING OF HUMAN BEHAVIOR PREDICTORS

The daily activities performed by a disabled or elderly person can be monitored by a smart environment, and the acquired data can be used to learn a predictive model of user behavior. To speed up the learning, several researchers designed collaborative learning systems that use data from multiple users. However, disclosing the daily activities of an elderly or disabled user raises privacy concerns.

In this chapter, we use state-of-the-art deep neural network-based techniques to learn predictive human activity models in the local, centralized, and federated learning settings. A novel aspect of our work is that we carefully track the temporal evolution of the data available to the learner and the data shared by the user. In contrast to previous work where users shared all their data with the centralized learner, we consider users that aim to preserve their privacy. Thus, they choose between approaches in order to achieve their goals of predictive accuracy while minimizing the shared data. To help users make decisions before disclosing any data, we use machine learning to predict the degree to which a user would benefit from collaborative learning. We validate our approaches on real-world data.

Training Data for Collaborative Learning in Smart Environments

The performance of machine learning models depends on the data used for training. Everything else being equal, more data is better, and highly expressive models, such as deep neural networks, require more training data to avoid overfitting. Many recent achievements of deep learning took place in a ”big data” regime; Google, Facebook and Amazon rely on a steady stream of data from

the users interacting with their services. For instance, a success story in federated learning is predicting the next word on a mobile device's keyboard [37], relying on a very large number of users receiving the collaborative learning client simultaneously.

However, in the case of a smart environment participating in a collaborative learning scheme, this model cannot be taken for granted. A privacy conscious user (or the environment acting on her behalf) would not share any data unless there is a strong likelihood that it would benefit from the transaction. At the same time, the user will stop sharing when no further benefit is likely. As we have seen in the introduction, this data minimization behavior approach had been recommended by government directives in the US, UK and EU. As the problem of privacy is particularly acute for the vulnerable elderly and disabled population, the regulatory pressure is likely to increase.

We need to emphasize that the data minimization principle does not preclude the use of collaborative learning and other cloud based techniques. It means however, that some of the simplifying assumptions are not applicable: we need a better understanding of the temporal aspect of data sharing: what training data is available, to whom and when.

The first simplifying assumption we need to discard is the synchronized start of data collection. The deployment of a smart environment for the disabled or elderly is not instantaneous. It requires physical installation of hardware, software configuration, user training and possibly legal and medical approval. Thus, the smart environment will be deployed for some users earlier than for others. As the deployment times are random but independent from each other, they follow a Poisson arrival process. Furthermore, the number of smart environments contributing to a single collaborative domain is significantly smaller than the countrywide domains used by web services. Under these conditions, modeling the deployment time is necessary, because it affects the amount of training data available to the predictor.

Let us discuss the problem of the data available for learning in a group of smart environments. We will consider a set of smart homes $H_1 \dots H_M$ that started to operate at times t_i^{start} distributed according to a Poisson process with an arrival rate λ. We take the perspective of the target smart home H_{tg} that had started to operate on day t_{tg}^{start}. Figure 4.1 illustrates an example with $M = 30$, $\lambda = 0.5$, starting time January 1st, and we are considering $H_{tg} = H_{14}$ with $t_{tg}^{start} =$ January 6.

Local learning involves the training of a model based on data collected from the same home. This approach has the highest level of privacy, as no personal information needs to leave the premises. The weakness of the local training model is the paucity of the data, especially early in the deployment. On day t_{tg}^{start}, the system has no training data whatsoever, on day $t_{tg}^{start} + 1$ it will only have one day of training data and so on. Thus we expect the accuracy to start from a very low level, but increase in time as training data accumulates. The red part of Home 14 data in Figure 4.1 shows the data available to this home on January 16 in the local training regime (ten days of recordings from January 6 to January 15).

Let us now consider the case of *centralized learning*. As the smart environment was deployed at different homes at different times, on the day the target home had started, a number of other homes are already operating and providing data. If home H_i started at time t_{tg}^{start}, the total amount of training data available at that point will be:

$$D_{tg}^{centralized} = \bigcup_{i; t_i^{start} < t_{tg}^{start}} D_i[t_i^{start} : t_{tg}^{start}] \tag{4.1}$$

For our example, the data available for training is the data from all homes where the system was deployed before January 16 - these are all the parts of bars shown in red in Figure 4.1.

Another simplifying assumption that is not applicable for privacy-aware users is that once a user joined a collaborative learning setup, it will provide data indefinitely into the future. To do this

might be in the interest of the central authority, but it is not compatible with the privacy principle of data minimization. A rational user will stop providing data to the central system as soon as the local learning yields better results than the predictive models received from the center. As shown by the termination of the bars in Figure 4.1, this *cross-over point* might happen sooner or later in time and it triggers the end of sharing data t_i^{shend}. We note that this time point only shows the end of data *sharing*; the smart environment will continue to operate and the local learning will continue to receive data past this time. Thus the data available for centralized learning will be:

$$D_{tg}^{centralized} = \bigcup_{i; t_i^{start} < t_{tg}^{start}} D_i[t_i^{start} : \min(t_i^{shend}, t_{tg}^{start})] \tag{4.2}$$

Federated learning operates on the same amount of data, with the difference that the data is never put together to a shared database.

Learning the Activity Prediction Model

In this section we describe the architecture and training process of a human activity predictor for a smart environment that predicts the future activities of the residents based on the history of activities and current environment. We represent the input as a sequence of tuples (h, d, a) containing one-hot encoded hour of the day h, day of the week d and activity label a. Our predictor f takes a sequence of l tuples and outputs the probability of occurrence of next activity:

$$f((h_t, d_t, a_t), ..., (h_{t+l-1}, d_{t+l-1}, a_{t+l-1})) \rightarrow p(a_{t+l}) \tag{4.3}$$

Our goal is to find the "best" predictor. One way to formalize this is by assuming that the function f is part of a parameterized family of sufficiently expressive functions $f(\cdot) = F(\cdot, \theta)$. In our case,

this family will be a particular type of deep neural network, and θ will map to the network weights - but many other choices exist. Thus finding the best function is mapped to finding the optimal $\theta = \theta^*$.

Naturally, we cannot exactly predict every activity due to the inherent randomness of the human behavior. We will define the accuracy of predictor in the form of a loss function expressed as the cross-entropy between the predicted probabilities and the actually occurring activity. The optimal θ^* will be the value that minimizes this loss over the available training data. In the remainder of this section, our focus will be on finding the appropriate form for the function F and the optimization process for finding θ^*.

A Long-Short Term Memory Based Activity Predictor

In recent years, time series predictors based on a specific type of recurrent neural networks, Long-Short Term Memory (LSTM) [39] had been successfully applied to problems ranging from natural language processing [91, 32] to robotics, computer vision and taxi demand prediction [12, 51, 96] and predictive caching [101]. Compared to other machine learning approaches where feature engineering is essential, deep neural networks, trained end-to-end using stochastic gradient descent, learn their own latent feature encoding. Within the field of deep neural networks, LSTMs have the advantage of having a learned memory state. This allows a prediction to be conditioned on events that happened many time steps in the past, while still handling one event at a time.

Fig. 4.2 shows the architecture of a deep neural network designed to learn the prediction function Eq 4.3. The input layer of shape $l \times n$ encodes the l tuples of history. The second layer is an LSTM of size 256 unrolled l times. The hidden state h and the cell state c (memory) in the previous time step alongside the input in the current time step is given to the current LSTM cell. This procedure runs repeatedly until all data in the given sequence is processed. At that point, the output of the

LSTM cell o will become the input of the next layer, which is a dense layer with a ReLU activation function. This layer is followed by a dropout layer [84] with a dropout rate of 0.5 to improve the generalization of the model. Finally, we have another dense layer with a softmax activation function that outputs a probability for each activity. For training purposes, we are using a cross-entropy loss between the output of the softmax and the actual next activity. When deployed and used as a predictor, the smart environment can take the activity with the highest probability to be the predicted activity for the next time step.

In the following, we describe three possible scenarios for training activity predictors for the smart environments: local training and two collaborative training scenarios - centralized and federated. As we predict activities in real-time since after training, we only need to do a feed forward pass to compute the predictions. Following the practice of deep learning literature, to make fair comparisons for all the training models we are using exactly the same predictor architecture from Fig. 4.2.

Local Training

For the case of local training, the smart environment uses only the data collected from the user to train a predictor specific for the environment. Overall the number of trained predictors is the number of deployed environments. The advantage of this model is that the predictor is tailored to the home. The disadvantage is that the network is training with less data for instance, for the first day there is only one day of training data. The training process will be repeated overnight using the full set of data available for the node. The local training process is outlined in Fig 4.3-Top.

The accuracy of the prediction is measured on the home's own data. In general, we expect the paucity of the training data to result in an initially weak predictor which, however, will improve in time as more training data becomes available.

Centralized Training

In the case of the centralized training model we assume a cloud-based central authority. The participating homes upload their daily logs to the central authority as training data. The central authority runs the learning algorithm daily, creating a single predictor which is transferred to the homes. The training data available to the centralized learning is described by Eq.4.2. This learning model is shown in Fig 4.3-Middle.

The accuracy of this predictor is then evaluated on the local home's test data. Thus, the same predictor will have different accuracy results in different homes. A positive aspect of the centralized predictions is that the learning happens with much more data especially compared to the local learning for smart environments recently deployed. A weakness of the model is that the centralized learner learns a shared model, and will not adapt to the preferences and idiosyncrasies of the individual users. Users that joined the centralized learning pool will provide more data and they had more opportunities to shape the predictors to their own routines. We thus expect that for each node the centralized learning model will start with a better accuracy for a newly joined node, but it will improve comparatively slower from there.

Federated Training

Federated training is a variant of collaborative learning which does not require the participating nodes to share their data. Each node implements a learner that has access to the locally generated data. As in the centralized training approach, there is a cloud based central authority that learns and distributes a shared model. However, in contrast to the centralized approach, the central authority does not receive training data from the environments, but parameters from the locally updated models.

There are several techniques through which the federated learner can update the shared model. The approach we use is the federated averaging model introduced by McMahan et al. [60], due to its robustness to imbalanced datasets like the ones found in smart environments with different deployment dates blue (see Fig 4.5), where some homes provide significantly more data as they started earlier. The updated model is then transmitted to the nodes. Fig 4.3-Bottom shows the organization of the federated training approach.

There are many similarities between the federated and centralized models. The total amount of data to which the system as a distributed learner has access is the same, as shown in Eq.4.2. The difference, however, is that the centralized model has access to this data directly, while the federated model only through the mediation of the parameters of the local learners.

The same considerations apply to the expected accuracy of the shared model. Everything else being equal, we expect the centralized approach to show a better accuracy than the federated one, because it has a better access to the training data. A way to illustrate this is that we can always emulate federated learning in a centralized fashion, but not the other way around. Naturally, due to the randomness inherent in human behavior, it is possible that for a given day the federated model to be better for a particular home than the centralized one.

The expected weaker performance of the federated learning model is compensated by the better privacy properties. As the federated learning only shares parameters on the local model, it is expected that less information is disclosed compared to the centralized learning approach.

The choice between federated and centralized learning for a privacy-aware agent boils down to the performance gap between them. If the performance gap is major, the smart environment is better off using a centralized approach (and, possibly, cutting off the data sharing when the local model catched up). If the centralized-federated performance gap is minor, the system is better off using federated learning.

Predicting If Smart Environments Benefit from Federated Training

Let us now summarize the expected accuracy profiles and the three learning approaches we consider:

- Local training: will start with a low accuracy due to lack of training data, the performance will increase in time, and in principle is limited only by natural variability of activities and by model capacity. Privacy is guaranteed as no data leaves the premises.
- Centralized training: will start with a higher accuracy due to existing training data from nodes that were deployed earlier. The accuracy will increase relatively slowly and, in addition, will be limited by the non-iid distribution of the training data between different environments. Significant privacy concerns due to data sharing.
- Federated training: accuracy profile expected to be similar to centralized training. Privacy concerns lower, but information leakage still possible.

Note that we are expecting that eventually the local training will overtake the collaborative learning approaches. At this point, a rationally behaving privacy aware smart environment will stop participating in the collaborative learning model, stop sharing data and continue improving its activity predictor using local learning.

One additional insight we must consider is that simply participating in the collaborative learning and sharing a single day's activities might be the largest privacy loss, as it might disclose the user's age, medical needs and disability condition. Disclosing further day's data of the same daily routine will disclose relatively few additional information. Thus, the smart environment must consider carefully whether it should participate in the collaborative learning even for a short time.

We are going to define a number of measurable quantities that would allow the environment to make these decisions. One such quantity is the *crossover point*: the day in the future from which the model acquired through local training will consistently overtake the one received from the collaborative learning (centralized or federated). Intuitively, the closer the crossover point is, the less justified is for the user to join the collaborative learning pool.

The second quantity of interest is the area between the local and collaborative learning models accuracy in time up to the crossover point. Using a term borrowed from the field of reinforcement learning, we will call this quantity *regret* - this is the overall accuracy performance lost if the user does not join the collaborative learning pool. The smaller the regret, the less justified is to join the collaborative learning pool.

Naturally, both the crossover point and the regret can be measured only after the fact. In this paper, we propose the hypothesis that while these values are difficult to predict, we can train a classifier for a good surrogate measure that can be used as a decision aid. We will create a classifier that, based on the histogram of the first k days of the node and the average over all nodes will predict whether the crossover point will happen before specific day d or not.

As a note: training such a classifier requires the collaboration of the central authority, and might result in the node not joining the collaborative learning pool. Thus, it would not be in interest of the centralized authority to provide this classifier if the authority has a business model that relies on data sharing. However, it *would* be in the interest of the authority to help make this decision if the privacy interests of the central authority and the nodes are aligned.

Experimental Study

In the previous section we made certain conjectures about the accuracy profiles of the activity predictors. *Qualitatively*, these predictions are supported by objective facts: we know that local training has less training data than collaborative ones, and we know that centralized training can emulate federated learning but not the vice-versa. However, any practical application would need to rely on the *quantitative* results. For instance, if the crossover point would take years to reach, collaborative learning would be the only reasonable alternative for a smart environment. If the difference between the centralized and federated learning results is large, the system will need to choose centralized learning even if privacy vulnerabilities exist.

These quantitative factors strongly depend on the actual data. We could be right about the overall patterns, but wrong about the scales at which these patterns happen. Performing experiments using real world data is the only way in which we can understand the decisions faced by smart environments.

Datasets and Pre-processing

For our experiments we used the datasets collected by the CASAS project [16] [1]. This collection contains 30 datasets collected in homes with volunteer residents performing their daily routines. There is a significant diversity in the datasets and the routines: some of the residents were younger adults, some were healthy older adults, some were older adults with dementia, and some were having pets.

In order to make the datasets suitable for our experiments, we performed several pre-processing

[1] https://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+from+Continuous+Ambient+Sensor+Data

steps.

Mapping the activity labels into a common ontology. The original activity labels are closely related, but not fully identical across the various datasets. Labels at various levels of granularity exist, such as *work*, *work at table*, *work on computer* and *work at desk*. The use of different labels would make any form of collaborative learning impossible, and local learning difficult to compare between datasets. We solved this problem by mapping the activity labels to a higher level, courser granularity categories, creating 10 new category labels without overlap between them. The mapping of the original activity labels to the new common ontology is shown in Table 4.1.

Converting to an event-based time series. The CASAS dataset uses a variable sampling rate from a resolution ranging from several seconds to sometimes hours. When the sampling rate is fast, it usually results in many repeated entries with the same activity label.

We have several choices to convert these entries into a format that is more suitable to machine learning. We could, for instance, map the entries into a shared, uniform time grid across all the datasets. However, this approach would create very large datasets, with redundant information.

Instead, we chose to use an *event-based* representation of the activity time series, by representing every contiguous activity with a single entry. One side effect of this is that the dataset for each day will be significantly shorter, but entries for the individual days might have a varying length. This, however, is naturally handled by the LSTM-based activity recognition engine. Note that this approach does not encode the length of an activity through the number of repeated entities. However, the temporal information is still present by the encoding of the time of the day as one of the features.

Feature representation We are using a representation where every entry in the time series has

Table 4.1: Mapping the dataset activities to a higher level category

Original Activities	**Category**
evening meds, morning meds, take medicine, exercise, toilet, groom, dress, r2.dress, bathe, personal hygiene, r2.personal hygiene	PERSONAL HEALTH AND HYGIENE
eat, eat breakfast, r2.eat breakfast, eat lunch, eat dinner	EAT
drink	DRINK
cook, r1.cook breakfast, cook lunch, cook dinner, cook, cook breakfast, wash dishes, wash breakfast dishes, wash lunch dishes, wash dinner dishes, laundry	CHORES
nap, sleep, r1.sleep, sleep out of bed, go to sleep / wake up (interval between them)	REST
relax, watch TV, read	RELAX
phone, entertain guests	SOCIAL
work, work at table, work on computer, work at desk	WORK
leave home	LEAVE HOME
enter home	ENTER HOME
other activity, step out, bed toilet transition	NOT TRACKED

three data fields: the hour of the day (as an integer 0-23), the day of the week (as an integer 0-6), and the activity label which is also encoded as an integer in the range 0-9. Each value was individually encoded with a one-hot representation, and the resulting values were concatenated. Thus the input data was organized in the form of 24+7+10=41 binary values. Correspondingly, the output is encoded as an array of 10 values which, being the output of a softmax layer, encode the probabilities of the next activity.

Modeling the deployment times. Our objective is to model the data available for a collaborative / local model at various moments in time. As we discussed in Section 4, for the collaborative learning models, this depends on the deployment schedule. The CASAS datasets were collected over many years, at time points separated by large intervals, sometimes from successive inhabitants from the same home. To model our scenario, we changed the starting times of the individual datasets to represent different smart environments deployed over the course of 2 months, with a Poisson arrival distribution. This is a realistic model of a small scale deployment by a local health care provider.

Training the Activity Predictor

Using the preprocessed datasets described in the previous section, we trained the LSTM-based activity predictor from Fig. 4.2. For the local training case, we trained 30 different predictors on their local data, for every day of operation. For the centralized predictor, we trained a single shared predictor on the data available, described by Eq. 4.2. For the local and centralized learning, we used the Keras library on top of Tensorflow 2.1.0. For the federated learning, we have trained local predictors with the appropriate local data and updated the shared model using Tensorflow-Federated 0.13.1.

The training configurations for the different models are shown in Table 4.2. The code is also available on https://github.com/sharare90/Privacy-Preserving-Learning

Results: Accuracy, Crossover Point and Regret

The approach we took in this paper is to focus on the individual user of the specific smart environment. The centralized and federated approaches are not a goal in themselves, they are useful only

Table 4.2: Local, centralized, and federated training hyperparameters.

Hyperparameters	Local	Centralized	Federated
batch size	64	64	64
number of epochs	500	500	-
lstm use bias	true	true	true
early stopping patience	50	50	-
early stopping minimum delta	0.01	0.01	-
number of rounds	-	-	20
client learning rate	0.001	-	0.001
server learning rate	-	0.001	0.5
client optimizer	Adam	-	Adam
server optimizer	-	Adam	SGD

inasmuch as they help the individual.

Thus, our performance evaluation is based on measuring the accuracy of the learned predictor (one per home for the local, a shared one for centralized and federated models) on the *individual user's data*. As a note, even when the predictor is shared, it will give different results for the individual users.

We found that the temporal evolution of the accuracy curves fall into several different patterns. Fig. 4.4 shows a selection of 12 out of the 30 homes in our dataset, chosen to be representative of the different patterns. Note that the starting day on the x axis varies reflecting the deployment day of the various smart environments.

We can make several observations:

Relatively good prediction results. In interpreting Fig. 4.4 we need to keep in mind that the accuracy of random prediction would be 0.1. Fully accurate prediction is not possible, as the users behavior can vary randomly from day to day. The ability to predict the next action with about 45% accuracy out of 10 possibilities is helpful for many applications for the smart environments.

The gap between the federated and centralized training is minor. As expected, we found that the centralized training gives better accuracy results than the federated. However, the differences are small and usually diminish in time. The practical conclusion, for a deployment is that a privacy-aware smart environment would participate in a federated training based collaborative model, as the privacy benefits are significant and the accuracy cost minor.

The crossover point is sometimes very early. The next question we need to investigate is the relationship between the local and the federated training models. Fig. 4.4 shows with a red triangle the crossover point when the local training overtakes the federated learning (if such a point exist in the time interval considered), and with a filled with yellow fill the area corresponding the regret - the accuracy lost if the environment would choose not to participate at all in the federated learning pool.

We found that the result validate our expectations about the shapes of the accuracy curves: the local learning starts out lower but eventually overtakes the federated learning in 10 out of 12 cases in the figure. In Home 3 the local training starts out better and stays as such, so the regret is zero. In Home 14, where the trends are as expected but the local learning did not yet overtake the accuracy of the federated at the end of the data collection.

Overall, the location of the crossover point varies. For homes 3, 25, 27 and 29, the crossover happens so early, and the regret area is so small that the deployment of the collaborative learning does not appear to be justified. For other homes, such as Home 2, the crossover happens almost a month after the deployment and the regret is significant.

Predicting the Benefits of Federated Training

The different profiles found in Fig 4.4 raise an interesting question: what factors make for certain homes local learning techniques with very low amount of training data overtake federated learning despite the much more data available to the latter? The main variable here is the way in which the shared model applies to the activities of the specific user. That is: it is not that the local learning is particularly efficient for these users, but that the shared model is not very good for them. The limiting factor of learning the shared model is that the user's behavior is not independent and identically distributed (iid).

Can we predict the future performance of the federated learning for a given node without joining the pool at all? Our hypothesis is that the proportion of the activities in the home for the first two days contain sufficient information to predict the performance of the predictor. Fig 4.5 shows the activity proportions for the same set of users as shown in Fig 4.4. A visual analysis shows that the users clearly spend different fractions of their time at different activities.

To predict the relative performance of federated and local training, we trained a classifier whose inputs are the values from Fig 4.5 encoded as floating point values in the $[0, 1]$ range, the deployment day, and a day $d \in \{1, 2, \ldots, 45\}$, and the output is a single binary value that answers the question: "is the crossover point happens on day d from the deployment"? We chose the values of d to be between 1 and 45 since if crossover point happens more than 45 days from deployment, we assume that the home will benefit from collaborative learning. As the number of training data points is small, this problem is better suitable for the traditional machine learning approaches. To find the best model, we trained four different classifiers based on decision trees, support vector machines, nearest neighbors and random forests.

The F1-score of the results are shown in Table 4.3. All models achieve a good predictive value,

Table 4.3: Comparison of classifiers for predicting the benefit of collaborative learning

Classifiers	**F1-score mean**
Decision Tree	0.70
SVM	0.72
Nearest Neighbors (k=2)	**0.77**
Random Forest (# estimators=10)	0.72

considering the very small amount of training data. The best performing model was the k-nearest neighbor with $k = 2$, possibly due to the fact that the best predictor of high federated learning performance is the similarity in profile to nodes already in the pool.

Overall, the performance of the classifier is sufficient to serve as a decision making aid in helping the user join the federated learning pool or not. One drawback of the approach is that the training of the classifier requires information from all nodes, and thus it can only be done by a centralized authority.

Figure 4.1: The data available for collaborative training for a group of users. The deployment time of the system is modeled through a Poisson arrival starting from January 1st. The users stop sharing data when further data sharing is not justified by the advantage of collaborative learning. The red part of the bars illustrates the data available for collaborative training on January 16th.

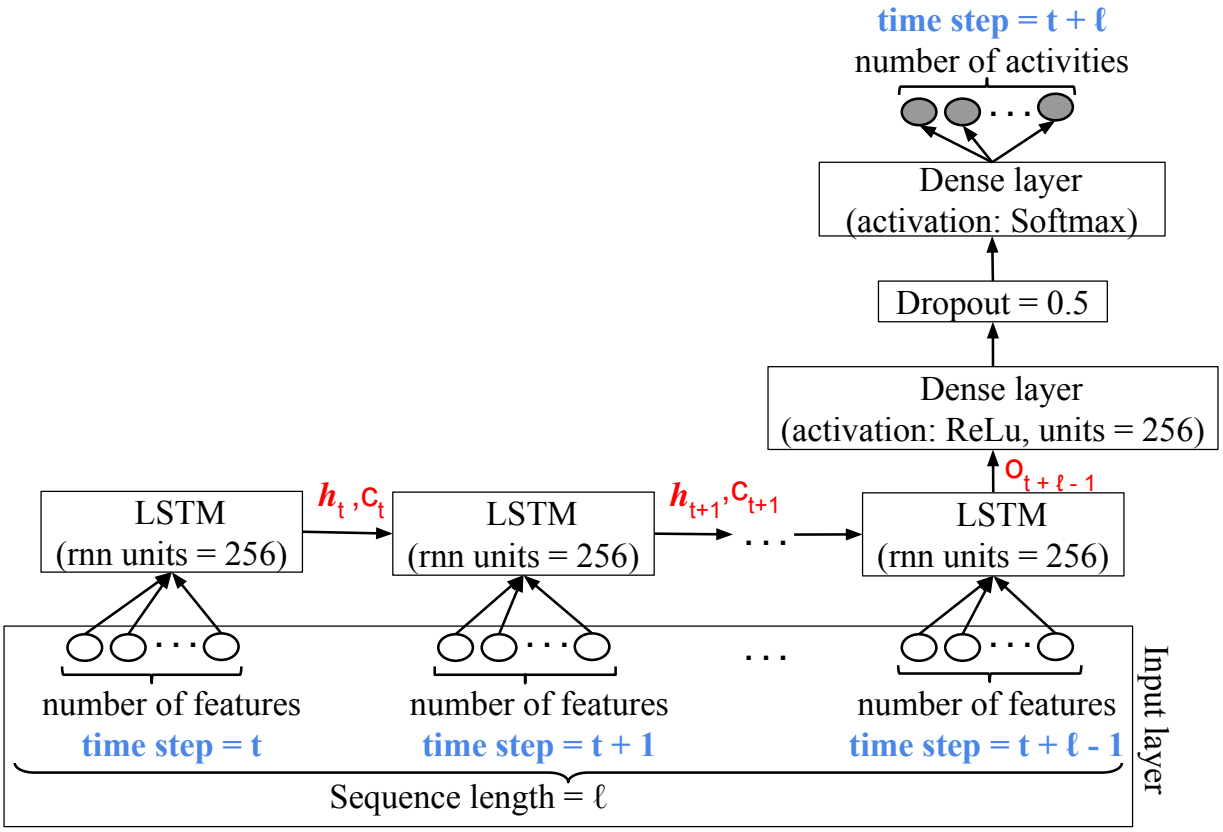

Figure 4.2: Architecture of LSTM based prediction model. Each circle in the input layer shows a single feature. A set of features is considered as the inputs in each time step. Gray circles in the output of the model correspond to the activities. h is the hidden state and c is the cell state.

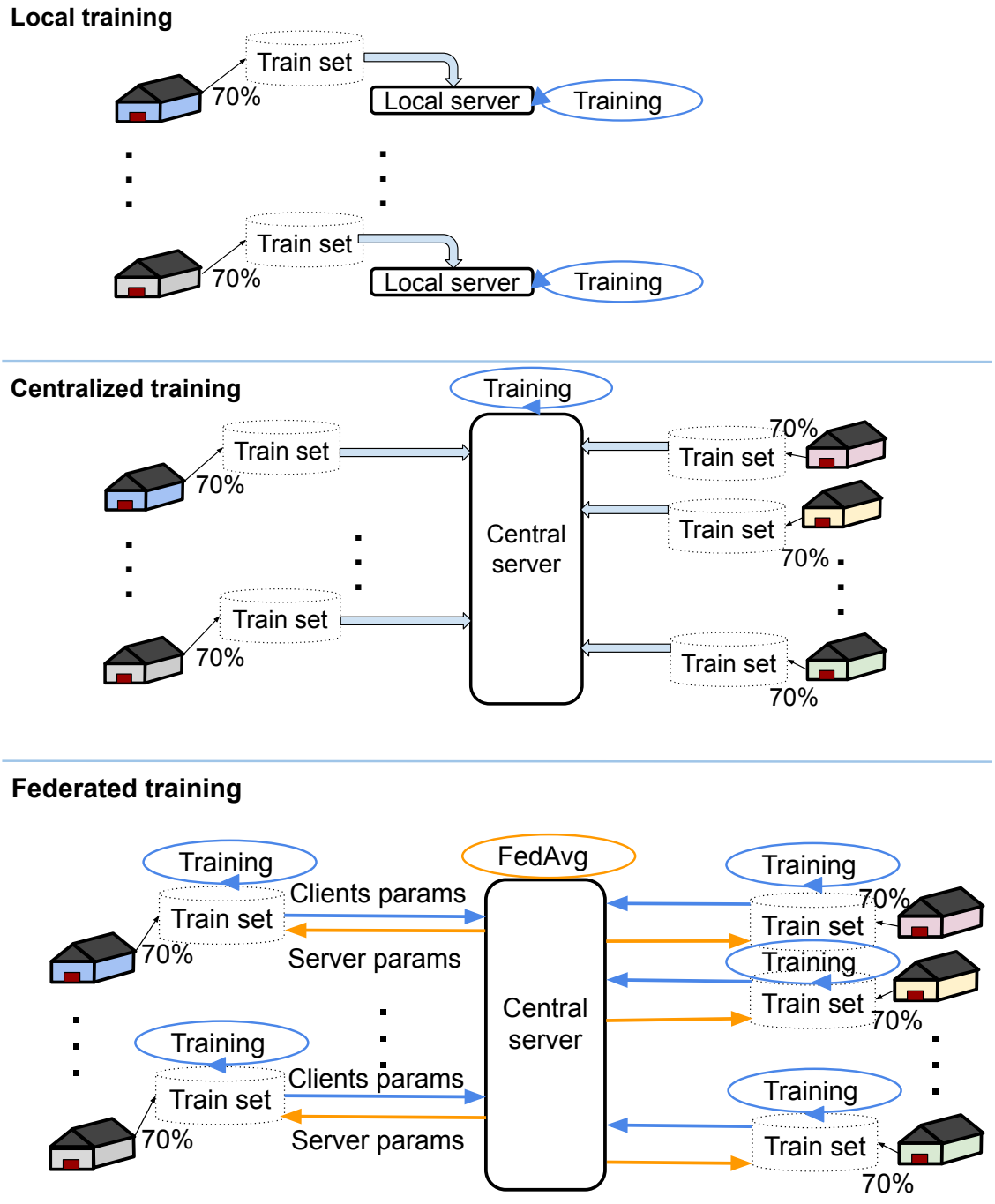

Figure 4.3: Activity prediction approaches 1. local (in-home) training (top), 2. centralized training (middle), and 3. federated training (bottom). We used 70% of each home's data for training and 30% for testing.

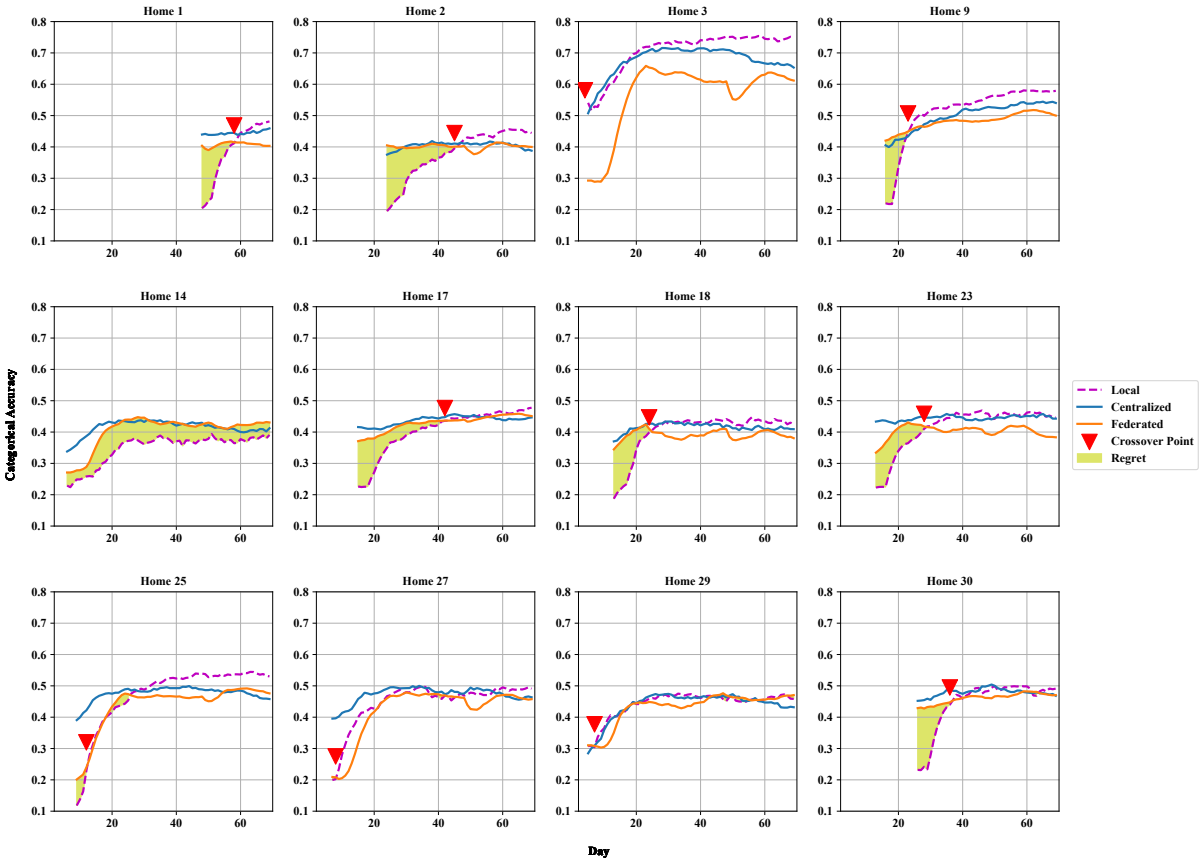

Figure 4.4: Accuracy on test data for a selection of 12 out of the 30 homes in our dataset with local training (magenta) vs centralized training (blue) and federated training (orange). The cross-point shows the first time that the local accuracy reaches the federated accuracy. Regret is the area between local accuracy and federated accuracy when local accuracy is lower. than federated accuracy.

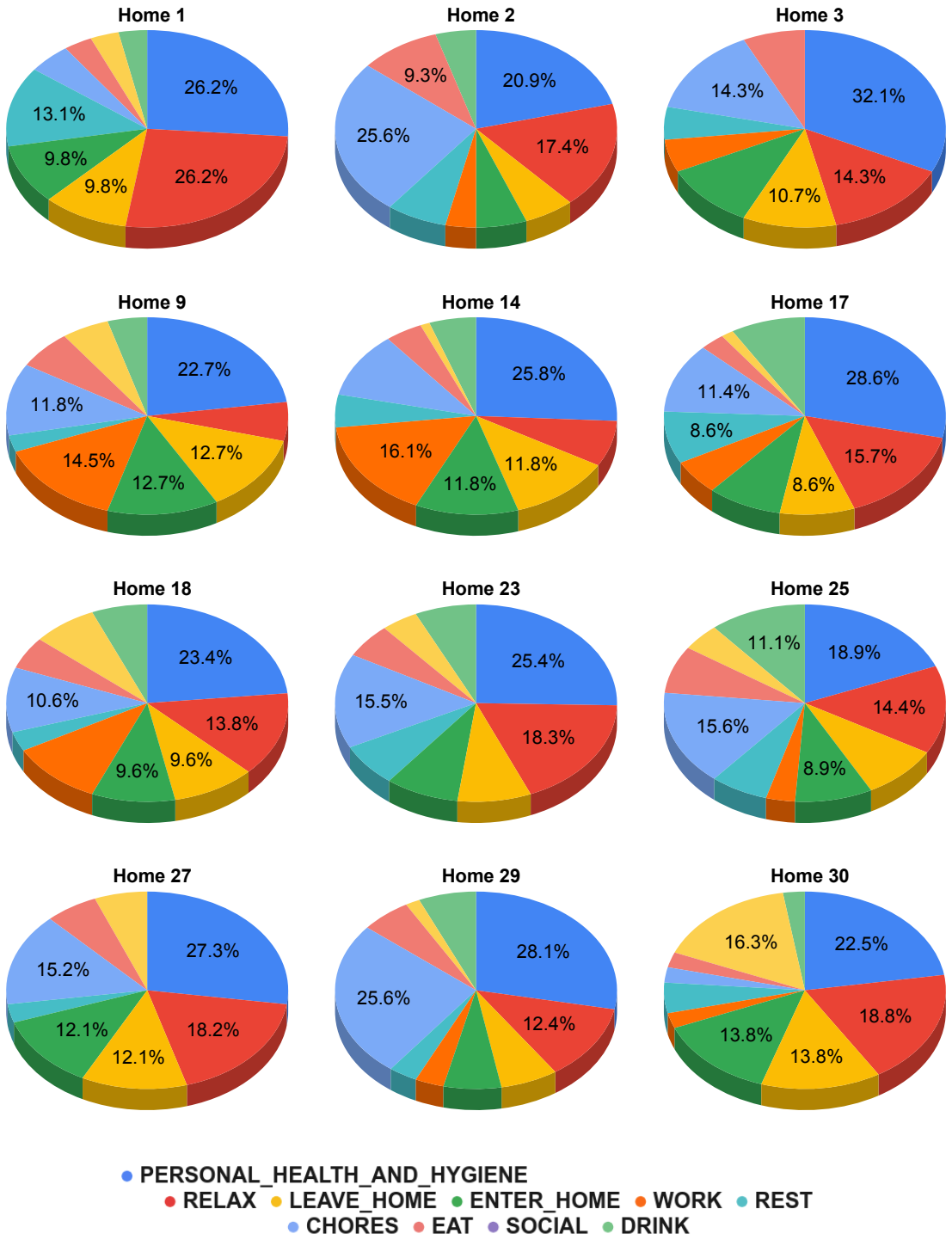

Figure 4.5: Activity proportions for the same set of homes as shown in Fig 4.4.

CHAPTER 5: PREDICTING COVID-19 PANDEMIC USING HUMAN'S CULTURAL BEHAVIOR DIMENSIONS, COMPARTMENTAL MODELS AND DEEP LEARNING

Throughout the Covid-19 pandemic, a significant amount of effort had been put into developing techniques that predict the number of infections under various assumptions about the public policy and non-pharmaceutical interventions. While both the available data and the sophistication of the AI models and available computing power exceed what was available in previous years, the overall success of prediction approaches was very limited. In this paper, we start from prediction algorithms proposed for XPrize Pandemic Response Challenge and consider several directions that might allow their improvement. Then, we investigate their performance over medium-term predictions extending over several months. We find that augmenting the algorithms with additional information about the culture of the modeled region, incorporating traditional compartmental models and up-to-date deep learning architectures can improve the performance for short term predictions, the accuracy of medium-term predictions is still very low and a significant amount of future research is needed to make such models a reliable component of a public policy toolbox.

Learning-based models for predicting the number of infections

In this section, we describe four alternative learning-based models for predicting the number of infected people in a given day of the pandemic in a particular area. The models assume that we have access to two streams of daily data. The *context stream* provides information about the total number of infected and dead people. This stream describes the basic context in which the prediction needs to be made - for instance, the total number of people already infected affects the number

of people infected in the following days. We will include in this stream also the specific identifiers or the country and geographical region to which this data refers. The *action stream*, in contrast, describes the specific actions, typically non-pharmaceutical interventions that authorities enacted in a given area. Following the Pandemic Response Challenge setup, we extract the context and action stream from the Oxford Covid-19 Government Response Tracker (OxCGRT) data [36]. The action stream data includes 12 Non-pharmaceutical Intervention (NPI) columns: school closing, workplace closing, cancel public events, restrictions on gatherings, close public transport, stay at home requirements, restrictions on internal movement, international travel controls, public information campaigns, testing policy, contact tracing, and facial coverings. We also use context columns for each region, such as country name, region name, geo ID, date, confirmed cases, confirmed deaths, and population. We ignored countries or regions for which no number of cases or deaths are available and filled the empty or missing values on NPI columns in the data with 0 for each region or country.

Finally, by having access to the country and geographical region, it is possible to extend the context and action stream with other information that can be looked up from other databases or web services. For instance, should we want to investigate the hypothesis that the weather affects the spread of an infection, this information could be, for example, brought in by correlating the geographical identifier with an external weather service.

Having access to the stream of information contained in the context and the action stream and whatever auxiliary data the system might choose to look up, the objective of the predictive model is to predict the number of infections in the next day, and through extrapolation, for a larger period into the future.

The notations used in this paper are as follows: for a region r with population P^r, we refer to the values of NPI columns for day t by NPI_t^r which is a vector of length 12. Each element i is an

integer between 0 and NPI_MAX_i. We denote the number of Covid-19 cases at day t by nC_t^r.

Learning based epidemiological models

Traditional epidemiological models, such as the SIR compartmental model aims to predict the evolution of the epidemic based on first principles and a relatively few number of human-understandable parameters. The input to these models is usually the current state of the pandemic and they are calibrated by human experts based on the infectiousness of the disease.

In contrast, learning based models use significantly more complex models with a very large number of parameters, such as deep neural networks. These parameters are not individually human-interpretable and the only realistic way to acquire them is through the use of learning. Often, these models look at the pandemic as a function unfolding in time, thus they take an input either a sliding window of the recent history of the pandemic at every timestep, or maintain internally a memory of it.

The official examples and the majority of entries to the Pandemic Response Challenge were such learning-based systems (although it is difficult to know how much human-expert data was individually incorporated). In the reminder of this section we will discuss four possible models (a model presented as the "official´´ one in the competition and three models designed by our team), presenting their architecture and design rationale in a comparable way (see Figure 5.1).

LSTM-UT-Cogn

The first model we are describing was developed by the organizers of the Pandemic Response Challenge and provided to the teams that qualified to the finals of the competition [63] to serve as a metric for prescriptive measures. Although this model did not directly compete in the challenge,

it was clearly seen as a state-of-the-art model at that point in the pandemic.

The model, shown in Figure 5.1-bottom-left, is unusual in that it is using two separate branches for the context and the action data, with the predicted value being the proportion of new people infected from the population that is currently not infected (naturally, the absolute number of infected people can be calculated from this value). The input of the first branch is the infection ratio from the context stream which is processed by an LSTM layer, followed by a dense layer with one node and soft plus activation. In the second branch, the model takes as inputs the NPIs from the action stream, which is processed by a LSTM followed by a dense layer with a single node and sigmoid activation function. The outputs of context branch h and action branch g are combined using a lambda layer implementing the formula $(1 - g) \times h$ to produce the output of the model. The model was trained on sequences of length of 21 days.

The remaining three models were developed by our team, partially as part of our competition entry, partially through later improvements.

LSTM-Baseline

The simplest, baseline model we are proposing also uses a LSTM network which in recent years had became the most popular way in the deep learning community to process data from time series that is presented to the model one at a time. In this, simplest model (Figure 5.1-top-left), we investigate the hypothesis that the LSTM network can learn how to select the important information from the combined context and action stream without any further input from the modeler. Before inputting our data to LSTM, we have to preprocess it such that the values are normalized. To achieve this, we use the same "Infection Ratio" column that is evaluated as follows: First, we compute the infected proportion by dividing the number of cases by population in region r and day

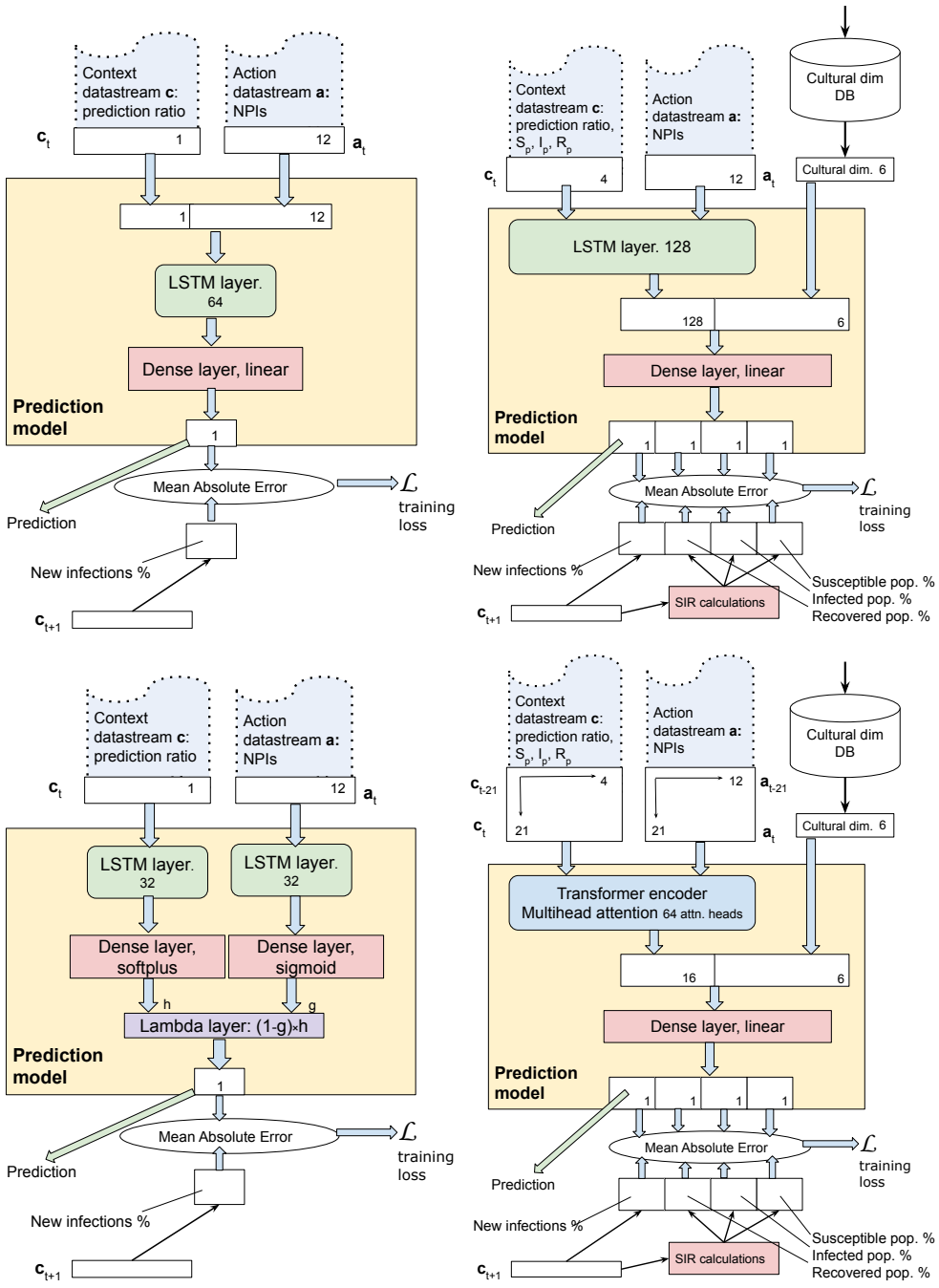

Figure 5.1: The architecture of the compared models: LSTM-Baseline (top-left), LSTM-UT-Cogn (bottom-left), LSTM-CultD-SIR (top-right) TRANSENC-CultD-SIR (bottom-right).

t. In other words, we compute $a_t^r = \frac{n\mathcal{C}_t^r}{P_t^r}$. Then, the smoothed version of a_t^r is computed for each day by getting the average of these values in a 7-day time window. Next, we compute the percent change by $\nabla_{t+1}^r = \frac{a_{t+1}^r - a_t^r}{a_t^r}$

The model processes streams of data of a width of 13, with the first column being the value for the infection ratio while the rest of the columns representing the NPIs. This input goes into an LSTM node with 64 nodes, and they are processed by a Dense layer with just one neuron that outputs "Infection Ratio" for the next day. We use the L1 loss between the output of the model and the real value of the "Infection Ratio" to train the network. During the inference, we only have access to NPI columns on each day in the future and use the prediction of the network as the "Infection Ratio" input for the following days. We also clip the network's output between 0 and 2 during inference to make sure that the outputs do not diverge. This is especially important when we use the model for longer predictions.

Taking into account culture

It had been an important part of the public narrative of the pandemic that various aspects of the interventions such as mask wearing, refraining from large gatherings, adherence to social distancing rules and vaccinations are culture-dependent.

Unfortunately, quantifying various aspects of the culture as relates to the pandemic is not easy. Furthermore, similar cultures can accommodate very different public policies, as illustrated by the case of Scandinavian countries where culturally similar countries like Sweden and Norway chose to adapt different policy approaches. Nevertheless, the hypothesis that taking the cultural aspects into consideration can improve the prediction accuracy is definitely worth considering. A problem in implementing such a system is that in the social sciences culture is often discussed in qualitative, narrative form. There are relatively few examples of quantitative models of human interactions.

One of the efforts that had attempted to assign numerical values to aspects of the culture of various nations is that cultural dimensions model [41] which attempted to quantify natural culture along six numerical dimensions. Public databases are available at a nation-state level. While this model had received significant criticism over the years, among other things for the choice of a nation as the resolution of the model. For instance, the model does not differentiate between California and Alabama in the United States. However, to our best knowledge this is the only culture quantification model for which public databases exist for the majority of regions.

We note that we do not make any assumptions about the impact of the cultural dimension values on our prediction - we add these values to the system and allow it to learn their possible impact.

Adding compartmental models

We propose a method that leverages compartmental models that are mathematical models used for predicting pandemics. The proposed method not only does allow us to combine data available with mathematical models, but also generalizes to evolving nature of the pandemic since the trend is learned through available data. This allows us to better address the impact of decisions such as school closing. Furthermore, this approach is preferred when there is not much data since instead of learning the whole pattern in a black box model, we try to estimate the parameters of a compartmental model which is well studied and can describe why the model believes the trend is going to change in a particular way. In our implementation, we consider SIR models. SIR model assumes that the number of people are fixed. We denote the population by P. Everyone is in one of three states: Susceptible (S), Infected (I), and Recovered (R). We add these columns to the data for each country using the following equations:

$$S_t = S_{t-1} - \textit{newCases}_t \tag{5.1}$$

where $S_0 = \textit{Population}$

$$I_t = I_{t-1} - \frac{1}{d} \times I_{t-1} + \textit{newCases}_t - \textit{newDeaths}_t \tag{5.2}$$

where $1/d$ is the daily recovery rate and d is the average number of days required to recovering.

$$R_t = \textit{Population} - S_t - I_t \tag{5.3}$$

The transition from these states can be modeled by parameters α and β as described below:

$$S' = -\alpha \times S \times I \tag{5.4}$$

$$I' = \alpha \times S \times I - \beta \times I \tag{5.5}$$

$$R' = \beta \times I, \tag{5.6}$$

where S', I', and R' are the rate of change in value of S, I, and R respectively. We train both LSTM and Transformer networks to take input from last T days and predict the value of S_p, I_p, R_p which are susceptible fraction of the population, infected fraction of the population and recovered fraction of the population for the next day(s). The model looks at NPI and all other data and outputs S_p, I_p, and R_p.

LSTM based predictor using cultural dimensions and the SIR model

In this model, we use compartmental model to create new columns susceptible fraction of the population (S_p), infected fraction of the population (I_p), and recovered fraction of the population (R_p). We initialize S_p with 1, and I_p and R_p with 0. Then, we calculate these values for the next rows based on equations 1, 2, and 3 over population for each country or region. We use 14 as

the average number of days required for recovery to compute recovery rate in equation 2. We use these columns alongside the infection ratio column as context input. We concatenate the context input and action input (NPI columns) of 21 previous days and feed it to a LSTM layer. Then, we concatenate new features such as cultural features of Hofstede dimensions as constant features to the output of the LSTM layer and feed them to a dense layer with 4 nodes. The model is trained with Adam optimizer and mean absolute error and outputs the infection ratio, S_p, I_p, and R_p. See Figure 5.1-top-right.

Transformer encoder based predictor using cultural dimensions and the SIR model

This model is similar to LSTM-CultD-SIR, but we use a transformer encoder layer instead of the LSTM layer. The difference is that this model can read the whole sequence all at the same while the LSTM-CultD-SIR model reads that sequence one by one. Transformers were first introduced in [89]. Transformer is a model architecture that instead of using recurrent networks uses an attention mechanism to learn relations between input and output. The main advantage of the transformers is their ability to see the sequence of data in parallel and learn very long-term interactions. We propose using the multi-head attention module from the transformer architecture to train the predictor model. To the best of our knowledge, we are the first to use attention models on NPI features and combining the attention model's output with the cultural dimensions and the SIR model for prediction of new Covid-19 cases.

The transformer layer includes attention and normalization part and a feed forward part. The attention and normalization part includes a multi head attention layer, dropout and normalization layer. The feed forward layer is a sequence of two dense layers one with ReLU activation and the other one with linear activation, a dropout layer and a normalization layer. See Figure 5.1-bottom-right.

To evaluate the predictive accuracy of the model, we need a metric that smooths out daily variations and is comparable across regions with different populations. Thus for a region r with a population P_r we will use the average number of cases per 100k people over a span of 7 days:

$$\text{Cumul-7DMA-MAE-per-100K}_{(r)} = \sum_{d \in \text{D}} |\bar{y} - \bar{\hat{y}}| \times \frac{100000}{P_r}, \tag{5.7}$$

where $\bar{x}$ is the 7-day moving average on x and denotes the population in region r.

We trained all the models separately by using a few months of data from all the countries and regions. For every model, we use 1000 epochs for training with early stopping with patience 20 that restores best weights. We split the training data into training and validation with 90% and 10% rate, respectively, with a batch size of 32. The learning rate is also 0.001.

We run two different experiments, with the training data and test data being chosen from different calendar months. Experiment E2020 used data from January 2020 to July 2020 for training and data from August 2020 to end of December 2020 for evaluation. Experiment E2021 used data from the whole year of 2020 and made predictions for January to April 2021. See Figure 5.2.

Based on our experiments, our TRANSENC-CultD-SIR approach had the lowest cumulative mean absolute error per 100k over 7 days. See Figure 5.3 for the overall cumulative 7 day moving average mean absolute error per 100k for both of the experiments. Also, Figure 5.4 shows the color scaled version of this metric for all the countries and regions around the world for E2020 experiment. The green nodes are showing the regions or countries with the lowest error (0 to 2k) per 100k, and red nodes are showing 8k or more error per 100k. We find that the TRANSENC-CultD-SIR approach has the lowest number of red-orange nodes which means that it has the better

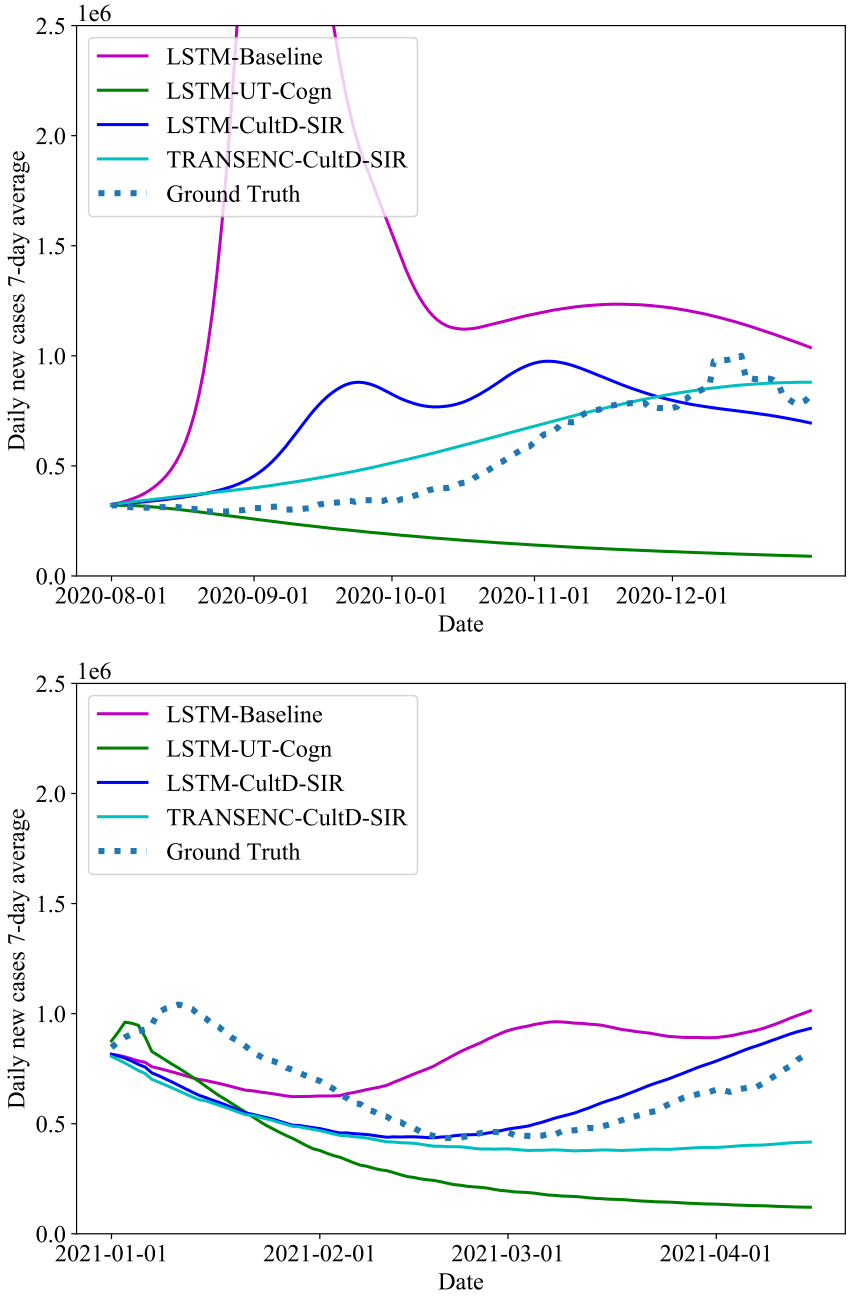

Figure 5.2: Average of 7-day predicted daily new cases over all countries using our predictors, LSTM-CultD-SIR and TRANSENC-CultD-SIR and two baselines LSTM-Baseline and LSTM-UT-Cogn. **Top:** E2020, **Bottom:** E2021.

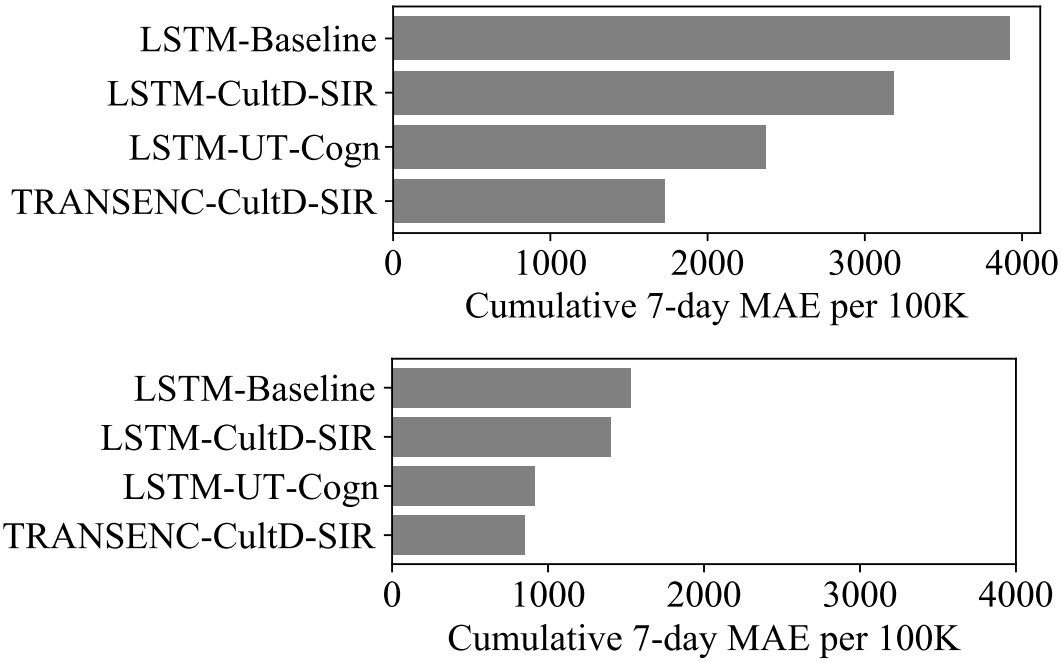

Figure 5.3: Cumulative 7-day mean absolute error per 100k for each prediction approach. **Top:** E2020. **Bottom:** E2021.

performance comparing to other approaches.

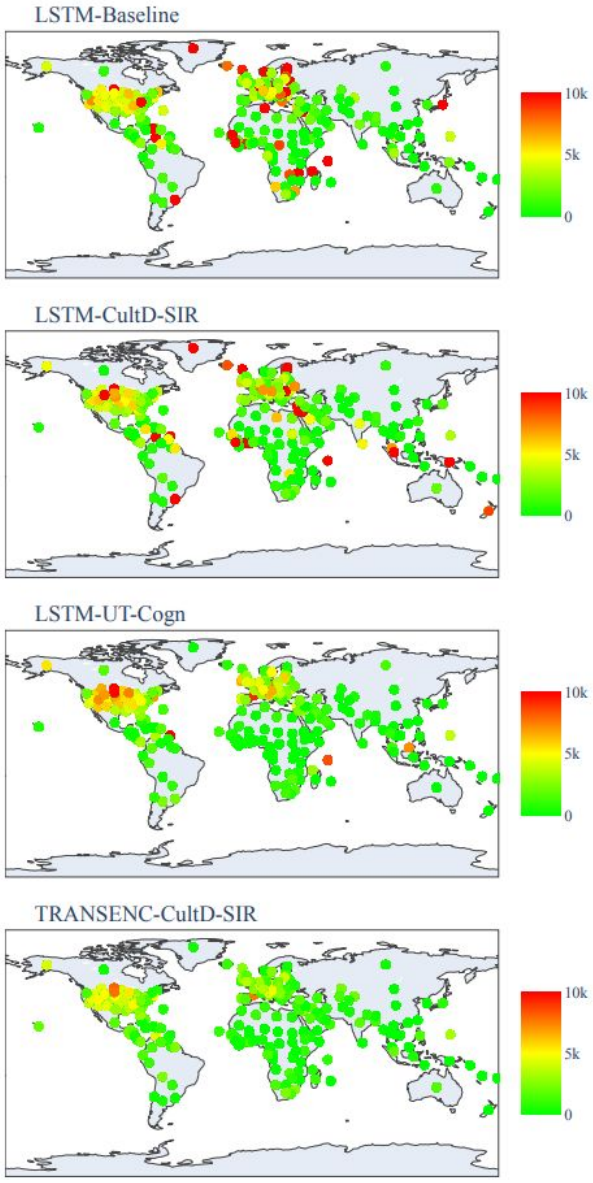

Figure 5.4: Color scaled cumulative 7-day mean absolute error per 100k per country or region based on each prediction approach. Green color shows zero to 2k and red color shows 8k or more cumulative 7-day mean absolute error per 100k.

CHAPTER 6: CONCLUSION

In this dissertation, first we presented a user modeling approach to create synthetic data of users' daily requests. To achieve this goal, we leveraged two real-world and two simulated datasets. Next, we proposed an LSTM-based and a probability-based approach for local caching of the potential future request. We then implemented two different LSTM architectures and reported prediction accuracy on all the datasets for both architectures. We implemented various caching strategies in the next step in our experiments. We compared our algorithm with three baselines based on final score and showed that the LSTM-based and majority voting approaches outperform other approaches. Especially for the higher quality formats, the difference is much more significant. Also, the results of the LSTM-based and majority voting methods for simulated datasets can almost attain the oracle performance. We also showed that the prediction accuracy is higher when we have a bigger dataset with more data points.

However, collecting data from users' daily activities is impossible due to serious privacy concerns. As a second step in this dissertation, we considered techniques for learning a human activity predictor for a smart environment in a realistic scenario where the privacy of the users must be weighted against the advantage offered by cloud based, collaborative learning models. We designed an activity predictor using state-of-the-art deep recurrent neural networks and trained it in three separate training scenarios: local, centralized and federated. A novel aspect of our work is that in contrast to previous studies we carefully accounted for what training data is available for the environments at every point in time. Our experiments had shown that there is only a minor difference between the centralized and federated approach, thus the greater privacy of federated learning would make it the preferred cloud based model. Furthermore, our experiments had also shown that the local training model will overtake the accuracy of the federated model for almost all the cases. In fact, for a significant subset of the environments, this crossover points happens within a couple of days

of the deployment. To allow the user to predict this and use it to maximize his privacy, we trained a classifier that can predict the early crossover based on the first days' data, with no disclosed information.

Finally, we studied the humans' cultural behavior for an applications in a larger environment such as a region or country and considered an important application of predicting a pandemic and prescribing intervention plans. We described the design of several pandemic prediction models and compared them with each other. As a comparison point, we used the model that was used as the official predictor for the finals of the XPrize Pandemic Response Challenge. The models introduced in this paper build on and improve our submission to this competition. By testing the models on data that extends several months after the competition, we can make several observations that can serve as lessons for modeling approaches in the future. First, models that are finely tuned to predict over the spans of days and weeks accurately can diverge significantly over the span of months. Second, sophisticated machine learning models such as transformer-style multi-head attention replacing LSTMs can produce an iterative improvement if everything else is equal but are not making a decisive difference in prediction accuracy. Third, while the canonical models of prediction such as the SIR compartmental model cannot, by themselves, provide an accurate prediction, they can serve a useful role in preventing runaway errors in the models. Finally, while cultural factors are clearly influencing the evolution of the pandemic, we do not yet have a method to incorporate this information in a rigorous manner.

LIST OF REFERENCES

[1] Giovanni Acampora et al. “A survey on ambient intelligence in healthcare”. In: *Proceedings of the IEEE* 101.12 (2013), pp. 2470–2494.

[2] Nasser Alshammari et al. “Openshs: Open smart home simulator”. In: *Sensors* 46.7 (2017), p. 1003.

[3] Talal Alshammari et al. “SIMADL: simulated activities of daily living dataset”. In: *Data* 3.2 (2018), p. 11.

[4] Sercan O Arik et al. “Interpretable sequence learning for COVID-19 forecasting”. In: *Advances in Neural Information Processing Systems* (2020).

[5] Marcelo G Armentano et al. “Special issue on knowledge discovery and user modeling for smart cities”. In: *Personal and Ubiquitous Computing* 24.4 (2020), pp. 437–439.

[6] Eugene Bagdasaryan et al. “How to backdoor federated learning”. In: *Proc. of Int’l Conf. on Artificial Intelligence and Statistics*. 2020, pp. 2938–2948.

[7] Nikola Banovic et al. “Modeling and understanding human routine behavior”. In: *Proc. of the 2016 CHI Conf. on Human Factors in Computing Systems*. 2016, pp. 248–260.

[8] Yoshua Bengio et al. “Predicting Infectiousness for Proactive Contact Tracing”. In: *Int’l Conf. on Learning Representations*. 2020.

[9] Giles Birchley et al. “Smart homes, private homes? An empirical study of technology researchers’ perceptions of ethical issues in developing smart-home health technologies”. In: *BMC medical ethics* 18.1 (2017), pp. 1–13.

[10] Thomas Burns et al. “Exploring the Predictability of Temperatures in a Scaled Model of a Smarthome”. In: *Sensors* 21.18 (2021), p. 6052.

[11] Thomas Burns et al. “IoT Augmented Physical Scale Model of a Suburban Home”. In: *2020 IEEE Int’l Conf. on Communications Workshops (ICC Workshops)*. 2020, pp. 1–5.

[12] Georgia Chalvatzaki et al. “LSTM-based network for human gait stability prediction in an intelligent robotic rollator”. In: *Proc. of Int’l Conf. on Robotics and Automation (ICRA)*. 2019, pp. 4225–4232.

[13] Zheng Chen et al. “User intention modeling in web applications using data mining”. In: *World Wide Web* 5.3 (2002), pp. 181–191.

[14] Sungjoon Choi, Eunwoo Kim, and Songhwai Oh. “Human behavior prediction for smart homes using deep learning”. In: *IEEE Int’l Symposium on Robot and Human Interactive Communication (RO-MAN)*. 2013, pp. 173–179.

[15] *Consumer data privacy in a networked world: A framework for protecting privacy and promoting innovation in the global digital economy*. Tech. rep. White House, Washington, DC, 2012, pp. 1–62.

[16] Diane J Cook. “Learning setting-generalized activity models for smart spaces”. In: *IEEE Intelligent Systems* 2010.99 (2010), p. 1.

[17] Diane J Cook and Narayanan C Krishnan. *Activity learning: discovering, recognizing, and predicting human behavior from sensor data*. John Wiley & Sons, 2015.

[18] Diane J Cook et al. “CASAS: A smart home in a box”. In: *Computer* (2013), pp. 62–69.

[19] Jessamyn Dahmen and Diane Cook. “SynSys: A synthetic data generation system for healthcare applications”. In: *Sensors* 19.5 (2019), p. 1181.

[20] Jessamyn Dahmen et al. “Activity learning as a foundation for security monitoring in smart homes”. In: *Sensors* 17.4 (2017), p. 737.

[21] Nicholas G Davies et al. “Effects of non-pharmaceutical interventions on COVID-19 cases, deaths, and demand for hospital services in the UK: a modelling study”. In: *The Lancet Public Health* 5.7 (2020), e375–e385.

[22] Jonas Dehning et al. “Inferring COVID-19 spreading rates and potential change points for case number forecasts”. In: *medRxiv* (2020).

[23] Thomas G Dietterich. “Ensemble methods in machine learning”. In: *Proc. of the Int’l Workshop on Multiple Classifier Systems*. 2000, pp. 1–15.

[24] Christopher Ifeanyi Eke et al. “A survey of user profiling: State-of-the-art, challenges, and solutions”. In: *IEEE Access* 7 (2019), pp. 144907–144924.

[25] Eric Elliott et al. “Peer-to-Peer Energy Trading and Grid Impact Studies in Smart Communities”. In: *2020 Int’l Conf. on Computing, Networking and Communications (ICNC)*. 2020, pp. 674–678.

[26] Sarah Fallmann et al. “Reality and perception: Activity monitoring and data collection within a real-world smart home”. In: *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. IEEE. 2017, pp. 1–6.

[27] Christopher Feltner et al. “Smart walker for the visually impaired”. In: *ICC 2019-2019 IEEE Int’l Conf. on Communications (ICC)*. IEEE. 2019, pp. 1–6.

[28] Seth Flaxman et al. “Estimating the effects of non-pharmaceutical interventions on COVID-19 in Europe”. In: *Nature* 584.7820 (2020), pp. 257–261.

[29] Yannick Francillette et al. “Modeling, learning, and simulating human activities of daily living with behavior trees”. In: *Knowledge and Information Systems* 62 (2020), pp. 3881–3910.

[30] Enrique Frias-Martinez, Sherry Y Chen, and Xiaohui Liu. “Survey of data mining approaches to user modeling for adaptive hypermedia”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 36.6 (2006), pp. 734–749.

[31] Robin C Geyer, Tassilo Klein, and Moin Nabi. “Differentially private federated learning: A client level perspective”. In: *arXiv preprint arXiv:1712.07557* (2017).

[32] Shalini Ghosh et al. “Contextual LSTM (CLSTM) models for large scale NLP tasks”. In: *arXiv preprint arXiv:1602.06291* (2016).

[33] Ian J Goodfellow et al. “Generative adversarial networks”. In: *arXiv preprint arXiv:1406.2661* (2014).

[34] Yu Guan and Thomas Plötz. “Ensembles of deep LSTM learners for activity recognition using wearables”. In: *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1.2 (2017), p. 11.

[35] Ramanathan Guha et al. “User modeling for a personal assistant”. In: *Proc. of the Eighth ACM Int’l Conf. on Web Search and Data Mining*. 2015, pp. 275–284.

[36] Thomas Hale et al. “A global panel database of pandemic policies (Oxford COVID-19 Government Response Tracker)”. In: *Nature Human Behaviour* 5.4 (2021), pp. 529–538.

[37] Andrew Hard et al. “Federated learning for mobile keyboard prediction”. In: *arXiv preprint arXiv:1811.03604* (2018).

[38] Abdelsalam Helal, Andres Mendez-Vazquez, and Shantonu Hossain. “Specification and synthesis of sensory datasets in pervasive spaces”. In: *2009 IEEE Symposium on Computers and Communications*. IEEE. 2009, pp. 920–925.

[39] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[40] Geoffrey M Hodgson. “The ubiquity of habits and rules”. In: *Cambridge journal of economics* 21.6 (1997), pp. 663–684.

[41] Geert Hofstede. “Culture’s consequences: International differences in work-related values”. In: *Journal of Service Research* 5 (1984).

[42] Terence KL Hui, R Simon Sherratt, and Daniel Díaz Sánchez. “Major requirements for building Smart Homes in Smart Cities based on Internet of Things technologies”. In: *Future Generation Computer Systems* 76 (2017), pp. 358–369.

[43] Ji Chu Jiang et al. “Federated Learning in Smart City Sensing: Challenges and Opportunities”. In: *Sensors* 20.21 (2020), p. 6230.

[44] Long Jin et al. “Understanding user behavior in online social networks: A survey”. In: *IEEE Communications Magazine* 51.9 (2013), pp. 144–150.

[45] Xiaoyong Jin, Yu-Xiang Wang, and Xifeng Yan. “Inter-Series Attention Model for COVID-19 Forecasting”. In: *Proc. of the 2021 SIAM Int’l Conf. on Data Mining (SDM)*. 2021, pp. 495–503.

[46] James Johndrow et al. “Estimating the number of SARS-CoV-2 infections and the impact of mitigation policies in the United States”. In: *Harvard Data Science Review* (2020).

[47] Parikshit Juluri, Venkatesh Tamarapalli, and Deep Medhi. “Measurement of quality of experience of video-on-demand services: A survey”. In: *IEEE Communications Surveys & Tutorials* 18.1 (2016), pp. 401–418.

[48] Tim LM van Kasteren, Gwenn Englebienne, and Ben JA Kröse. “Human activity recognition from wireless sensor network data: Benchmark and software”. In: *Activity recognition in pervasive intelligent environments*. 2011, pp. 165–186.

[49] Siavash Khodadadeh et al. “Detecting unsafe use of a four-legged walker using IoT and deep learning”. In: *ICC 2019-2019 IEEE Int’l Conf. on Communications (ICC)*. 2019, pp. 1–6.

[50] Jakub Konečný et al. “Federated learning: Strategies for improving communication efficiency”. In: *arXiv preprint arXiv:1610.05492* (2016).

[51] Yu Kong et al. “Action Prediction From Videos via Memorizing Hard-to-Predict Samples”. In: *Proc. of AAAI Conf. on Artificial Intelligence (AAAI-2018)*. 2018.

[52] Hugo Larochelle and Yoshua Bengio. “Classification using discriminative restricted Boltzmann machines”. In: *Proc. of Int’l Conf. on Machine Learning (ICML-2008)*. 2008, pp. 536–543.

[53] Nicolas Le Roux and Yoshua Bengio. “Deep belief networks are compact universal approximators”. In: *Neural Computation* 22.8 (2010), pp. 2192–2207.

[54] Sheng Li and Handong Zhao. “A Survey on Representation Learning for User Modeling”. In: *Proc. of the Twenty-Ninth Int’l Joint Conf. on Artificial Intelligence*. 2020, pp. 4997–5003.

[55] Zhifang Liao et al. “TW-SIR: time-window based SIR for COVID-19 forecasts”. In: *Scientific reports* 10.1 (2020), pp. 1–15.

[56] Daniele Liciotti et al. “A sequential deep learning application for recognising human activities in smart homes”. In: *Neurocomputing* 396 (2020), pp. 501–513.

[57] Jason Ling et al. “Predicting the temperature dynamics of scaled model and real-world iot-enabled smart homes”. In: *2019 IEEE Global Communications Conf. (GLOBECOM)*. 2019, pp. 1–6.

[58] Magnus S Magnusson. “Discovering hidden time patterns in behavior: T-patterns and their detection”. In: *Behavior research methods, instruments, & computers* 32.1 (2000), pp. 93–110.

[59] Atalanti A Mastakouri and Bernhard Schölkopf. “Causal analysis of Covid-19 spread in Germany”. In: *Advances in Neural Information Processing Systems* (2020).

[60] Brendan McMahan et al. “Communication-efficient learning of deep networks from decentralized data”. In: *Artificial Intelligence and Statistics*. 2017, pp. 1273–1282.

[61] Mihir Mehta et al. “Early stage machine learning–based prediction of US county vulnerability to the COVID-19 pandemic: machine learning approach”. In: *JMIR public health and surveillance* 6.3 (2020), e19446.

[62] Matteo Mendula et al. “Interaction and Behaviour Evaluation for Smart Homes: Data Collection and Analytics in the ScaledHome Project”. In: *Proc. of the 23rd Int’l ACM Conf. on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. 2020, pp. 225–233.

[63] Risto Miikkulainen et al. “From Prediction to Prescription: Evolutionary Optimization of Nonpharmaceutical Interventions in the COVID-19 Pandemic”. In: *IEEE Transactions on Evolutionary Computation* 25.2 (2021), pp. 386–401.

[64] Bryan Minor, Janardhan Rao Doppa, and Diane J Cook. “Data-driven activity prediction: Algorithms, evaluation methodology, and applications”. In: *Proc. of ACM SIGKDD Int’l Conf. on Knowledge Discovery and Data Mining*. 2015, pp. 805–814.

[65] Kelly R Moran et al. “Epidemic forecasting is messier than weather forecasting: the role of human behavior and internet data streams in epidemic forecast”. In: *The Journal of infectious diseases* 214.suppl_4 (2016), S404–S408.

[66] Nafisa Mostofa et al. “A Smart Walker for People with Both Visual and Mobility Impairment”. In: *Sensors* 21.10 (2021), p. 3488.

[67] Nafisa Mostofa et al. “IoT-Enabled Smart Mobility Devices for Aging and Rehabilitation”. In: *ICC 2020-2020 IEEE Int’l Conf. on Communications (ICC)*. 2020, pp. 1–6.

[68] Haider Mshali, Tayeb Lemlouma, and Damien Magoni. “Adaptive monitoring system for e-health smart homes”. In: *Pervasive and Mobile Computing* 43 (2018), pp. 1–19.

[69] Haider Mshali et al. “A survey on health monitoring systems for health smart homes”. In: *International Journal of Industrial Ergonomics* 66 (2018), pp. 26–56.

[70] United Nations. “World Population Ageing 2019 (ST/ESA/SER.A/)”. In: *Department of Economic and Social Affairs, Population Division* (2020).

[71] Maja Pantic et al. “Human computing and machine understanding of human behavior: A survey”. In: *Artifical intelligence for human computing*. 2007, pp. 47–71.

[72] Homin Park et al. “Energy-efficient privacy protection for smart home environments using behavioral semantics”. In: *Sensors* 14.9 (2014), pp. 16235–16257.

[73] Roger Parker. “Human behavior modeling: The necessity of narrative”. In: *Computer Architecture in Industrial, Biomechanical and Biomedical Engineering*. 2019.

[74] Alex Pentland and Andrew Liu. “Modeling and prediction of human behavior”. In: *Neural computation* 11.1 (1999), pp. 229–242.

[75] Ori Plonsky et al. “Psychological forest: Predicting human behavior”. In: *Proc. of the AAAI Conf. on Artificial Intelligence*. Vol. 31. 1. 2017.

[76] Zhaozhi Qian, Ahmed M Alaa, and Mihaela van der Schaar. “When and How to Lift the Lockdown? Global COVID-19 Scenario Analysis and Policy Assessment using Compartmental Gaussian Processes”. In: *Advances in Neural Information Processing Systems* (2020).

[77] Parisa Rashidi and Alex Mihailidis. “A survey on ambient-assisted living tools for older adults”. In: *IEEE Journal of Biomedical and Health Informatics* 17.3 (2012), pp. 579–590.

[78] Doug Riecken. “Introduction: personalized views of personalization”. In: *Communications of the ACM* 43.8 (2000), pp. 26–28.

[79] Kristie Seymore, Andrew McCallum, Roni Rosenfeld, et al. “Learning hidden Markov model structure for information extraction”. In: *AAAI-99 workshop on machine learning for information extraction*. 1999, pp. 37–42.

[80] Kinza Shafique et al. “Internet of things (IoT) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5G-IoT scenarios”. In: *Ieee Access* 8 (2020), pp. 23022–23040.

[81] Mrinank Sharma et al. “How Robust are the Estimated Effects of Nonpharmaceutical Interventions against COVID-19?” In: *Advances in Neural Information Processing Systems* (2020).

[82] Ben Shneiderman. “Human-centered artificial intelligence: Three fresh ideas”. In: *AIS Transactions on Human-Computer Interaction* 12.3 (2020), pp. 109–124.

[83] Burrhus Frederic Skinner. *Science and human behavior*. 92904. 1965.

[84] Nitish Srivastava et al. “Dropout: a simple way to prevent neural networks from overfitting”. In: *Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958.

[85] VS Subrahmanian and Srijan Kumar. “Predicting human behavior: The next frontiers”. In: *Science* 355.6324 (2017), pp. 489–489.

[86] Alon Talmor et al. “CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge”. In: *Proc. of the Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 2019, pp. 4149–4158.

[87] Emmanuel Munguia Tapia, Stephen S Intille, and Kent Larson. “Activity recognition in the home using simple and ubiquitous sensors”. In: *Proc. of Int’l Conf. on Pervasive Computing*. 2004, pp. 158–175.

[88] Niall Twomey et al. “Unsupervised learning of sensor topologies for improving activity recognition in smart environments”. In: *Neurocomputing* 234 (2017), pp. 93–106.

[89] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[90] Shiqiang Wang et al. “Adaptive federated learning in resource constrained edge computing systems”. In: *IEEE Journal on Selected Areas in Communications* 37.6 (2019), pp. 1205–1221.

[91] Shuohang Wang and Jing Jiang. “Learning Natural Language Inference with LSTM”. In: *Proc. of Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2016, pp. 1442–1451.

[92] Gregory L Watson et al. “Pandemic velocity: Forecasting COVID-19 in the US with a machine learning & Bayesian time series compartmental model”. In: *PLoS computational biology* 17.3 (2021), e1008837.

[93] Krist Wongsuphasawat et al. “LifeFlow: visualizing an overview of event sequences”. In: *Proc. of the SIGCHI Conf. on human factors in computing systems*. 2011, pp. 1747–1756.

[94] Shaoen Wu et al. “Survey on prediction algorithms in smart homes”. In: *IEEE Internet of Things Journal* 4.3 (2017), pp. 636–644.

[95] Congxi Xiao et al. “C-Watcher: A Framework for Early Detection of High-Risk Neighborhoods Ahead of COVID-19 Outbreak”. In: *arXiv preprint arXiv:2012.12169* (2020).

[96] Jun Xu et al. “Real-time prediction of taxi demand using recurrent neural networks”. In: *IEEE Transactions on Intelligent Transportation Systems* 19.8 (2017), pp. 2572–2581.

[97] *Data minimization - UK Information Commisioner's office*. https://ico.org.uk/about-the-ico/news-and-events/ai-blog-data-minimisation-and-privacy-preserving-techniques-in-ai-systems/.

[98] *European Union's General Data Protection Regulation*. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679.

[99] Arnold YS Yeung et al. "Machine Learning–Based Prediction of Growth in Confirmed COVID-19 Infection Cases in 114 Countries Using Metrics of Nonpharmaceutical Interventions and Cultural Dimensions: Model Development and Validation". In: *Journal of Medical Internet Research* 23.4 (2021), e26628.

[100] AA Zaidan and BB Zaidan. "A review on intelligent process for smart home applications based on IoT: coherent taxonomy, motivation, open challenges, and recommendations". In: *Artificial Intelligence Review* 53.1 (2020), pp. 141–165.

[101] Sharare Zehtabian et al. "Modeling an intelligent controller for predictive caching in AR/VR-enabled home scenarios". In: *Pervasive and Mobile Computing (PMC)* (2021).

[102] Sharare Zehtabian et al. "Predicting infections in the Covid-19 pandemic - lessons learned". In: *arXiv preprint arXiv:submit/4057182* (2021).

[103] Sharare Zehtabian et al. "Predictive Caching for AR/VR Experiences in a Household Scenario". In: *Proc. of Int'l Conf. on Computing, Networking and Communications (ICNC-2020)*. 2020, pp. 591–595.

[104] Sharare Zehtabian et al. "Privacy-Preserving Learning of Human Activity Predictors in Smart Environments". In: *Proc. of IEEE Int'l Conf. on Computer Communications (INFOCOM-21)*. 2021.

[105] Sharare Zehtabian et al. "Supporting rehabilitation prescription compliance with an IoT-augmented four-legged walker". In: *2nd Workshop on AI for Aging, Rehabilitation and Independent Assisted Living (ARIAL'18)*. 2018.

[106] Rowan Zellers et al. “From Recognition to Cognition: Visual Commonsense Reasoning”. In: *The IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[107] Xiaoli Zhang et al. “Enabling Execution Assurance of Federated Learning at Untrusted Participants”. In: *Proc. of IEEE Int’l Conf. on Computer Communications (InfoCom-2020)*. 2020.

[108] Yue Zhao et al. “Federated learning with non-iid data”. In: *arXiv preprint arXiv:1806.00582* (2018).

[109] Difan Zou et al. “Epidemic model guided machine learning for COVID-19 forecasts in the United States”. In: *medRxiv* (2020).

[110] Ingrid Zukerman and David W. Albrecht. “Predictive statistical models for user modeling”. In: *User Modeling and User-Adapted Interaction* 11.1-2 (2001), pp. 5–18.