DEEP GRAPH LEARNING IN MOBILE HEALTH

A Dissertation
Presented to
The Academic Faculty

By

Guimin Dong

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Engineering and Applied Science
Department of Engineering Systems and Environment

University of Virginia

© Guimin Dong 2022

APPROVAL SHEET

This

Dissertation

is submitted in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

Author: Guimin Dong

Advisor:	Mehdi Boukhechba
Advisor:	Laura E. Barnes
Committee Member:	Mehdi Boukhechba
Committee Member:	Laura E. Barnes
Committee Member:	Afsaneh DoryabJundong Li
Committee Member:	Jundong Li
Committee Member:	Tariq Iqbal
Committee Member:	

Accepted for the School of Engineering and Applied Science:

[illegible]

Jennifer L. West, School of Engineering and Applied Science

May 2022

I’m not super. Any talents I have, I worked for – it seems a long time since I thought of myself as a hero.

Oliver Queen

For my wife Ke

ACKNOWLEDGMENTS

It gives me great pleasure to express my gratitude to the committee members for their time and support. To begin, I would want to thank my advisors, Dr. Mehdi Boukhechba and Dr. Laura Barnes, for their continual support and assistance during my time at UVA. I would not have been able to complete my dissertation without their assistance.

Additionally, I would like to express my gratitude to my committee members, Dr. Afsaneh Doryab, Dr. Jundong Li, and Dr. Tariq Iqbal, for their guidance and advice during my graduate studies. Their expertise serves as a significant source of inspiration for my studies. I would also like to thank other graduate colleagues, Dr. Lihua Cai, Mingyue Tang, and Zhiyuan Wang, for their collaboration, inspiration, and support. What's more, I would like to express my gratitude to other coauthors for their assistance and insights gained while doing research together, as well as to other individuals who assisted me during my Ph.D. studies.

Most importantly, I want to convey my heartfelt gratitude to my wife for her unwavering encouragement, support, and trust.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

DGL	Deep Graph Learning
GRL	Graph Representation Learning
GNN	Graph Neural Network
LSTM	Long Short-term Memeory
MIL	Multiple Instance Learning
FL	Federated Learning
HPA	Hypothalamic-pituitary-adrenal
MAE	Mean Absolute Error
SVM	Support Vector Machine
MLP	Multi Layer Perceptron
ILI	Influenza-like Illnes
PPG	Photoplethysmogram
CNN	Convolutional Neural Network
GAT	Graph Attention Network
GIN	Graph Isomorphic Network
SLA	Self-Loop-Augmented
ECG	Electrocardiogram

RIP	Respiratory Inductive Plethysmography
GPS	Global Positioning System
CR	Consistency Regularization
KD	Knowledge Distillation
EMA	Ecological momentary assessment
GIT	Graph Instance Transformer
GSSL	Contrastive Self-supervised Learning
GCN	Graph Convolution Network
MGIL	Multiple Graph Instance Learning
BCE	Binary Cross Entropy
KD	Knowledge Distillation
EMA	Ecological momentary assessment
GIT	Graph Instance Transformer
GRU	Gated Recurrent Neural Network

ABSTRACT

Mobile devices such as smartphones, smartwatches, and other wearable devices are equipped with a rich set of sensors that can collect human behavioral and physiological data continuously and unobtrusively. Data collected by using the embedded sensors (e.g., accelerometer, GPS sensor, and Bluetooth sensor) in mobile devices has been leveraged in a plethora of healthcare-related fields, including but not limited to physical state inference, mental health monitoring, and mobile intervention. Despite the recent achievements and advancements in mobile health (mhealth), wide adoption of mhealth remains a challenge. First of all, handcrafted feature engineering and conventional deep neural networks (e.g., Multi-Layer Perceptron, Convolutional Neural Network) are restricted to generating sufficient representations of raw mobile sensing data, making it difficult to capture complex interdependence within human behaviors. Secondly, complete responses of high-frequency ecological momentary assessments (EMAs) in the wild are impractical due to heavy user burden and low user engagement, resulting in sparsely annotated mobile sensing data at different levels of granularity. Last but not least, current centralized training of machine learning models can expose sensitive information of mobile users to privacy risks due to data breaches and misexploitation, preventing widespread use of mobile sensing. This research demonstrates a set of deep graph learning systems to overcome the above mentioned challenges and presents a state-of-the-art modeling paradigm in mobile sensing from a topological perspective.

Chapter 1

Introduction

Mobile sensing with embedded sensors (e.g., accelerometer, GPS, and Bluetooth) can be harnessed to unobtrusively collect fine-grained information about users' contexts and behaviors. Behavior markers can be extracted from mobile sensor data to represent human states and model human behaviors. Downstream machine learning techniques have been widely studied to recognize human behavioral patterns based on the extracted behavioral features. Given multi-modal sensors, handcrafted features are usually extracted to build machine learning models and to quantify the contribution of represented human behaviors to predict outcomes such as health and wellbeing [1, 2, 3]. For example, motion features such as magnitude of acceleration, can be extracted from accelerometer to study their correlations with different user contexts (e.g., location, activity, social context) [4]. Instead of using handcrafted features that are based on heuristics and domain knowledge, high level features can be automatically extracted from mobile sensing data, and leveraged to improve generalization of predictive modeling using deep learning models. For example, Boukhechba et al. built convolutional neural network (CNN) models based on photoplethysmogram (PPG) data to infer ambulatory activities [5]. Handcrafted feature engineering and conventional deep neural networks (e.g., Multi-Layer Perceptron, Convolutional Neural Network) are limited to generate sufficient representations of raw mobile sensing data in every modality, making it difficult to capture complex interdependence within human behaviors. On one hand, low level features extracted by handcrafted feature engineering can miss information about the inter-dependencies in different sensing modalities. On the other hand, the complex interactions within data points in and among different sensing streams may not be sufficiently embedded in deep features by CNN/RNN, especially when the data have hidden structural patterns and are generated from non-Euclidean domains [6]. Some discrete sensor signals, such as GPS and Bluetooth streams, can be represented as graphs to encode the structural interdependence and information generated from different domains. Handcrafted feature engineering and conventional neural networks, in this case, cannot produce representative embeddings for downstream learning tasks.

Deep Graph Learning (DGL) [7], including Graph Representation Learning (GRL) [8] and Graph Neural Networks (GNNs) [6], is an emerging research field in machine learning and has attracted wide attentions in solving inference and prediction problems with non-Euclidean graph structured data. In general, GRL is an unsupervised feature learning process that generates embeddings that map graphs into low-dimensional numerical vector spaces such that this embedding can optimally preserve the intrinsic graph properties [8]. And GNNs can learn powerful graph structure representations supervisedly or semi-supervisedly, encoding both node proximity and sub-graph structure given graph inputs [9]. With recent advancements in DGL, we propose a general representation learning framework that uses graphs to represent raw mobile sensing data and uses deep graph learning techniques to generate high-level features to capture complex human behavior dynamics.

The motivation for using DGL for mobile sensing data is illustrated as follows: in a multi-modal sensing environment, some mobile sensing data can be naturally represented as graph structure data, such as GPS trajectory and Bluetooth encounter network. The topological or geographic structures can be captured by using DGL [10]. With graph representation, topological features can be extracted by using Node and Graph embedding methods. For example, given a graph input, we can use Node2Vec [11], which is a Node embedding method, to generate a numerical representation for each node, such that the topological structure of nodes can be expressed in the numerical space. DGL framework can capture inter-dependence relationships between neighborhood of human states [6].

Human behaviors have spatial and temporal variations through time, making the corresponding graph representations have temporal topological changes. The DGL frameworks can extract the spatio-temporal features given an input of a sequence of graphs [12]. For example, integrating with Long short term memory (LSTM) [13], temporal GNN [14] can extract spatiotemporal features to predict health related targets. DGL can be used to generate structured and informative representations of diverse unstructured mobile sensing data (e.g., text messages, app usage records, daily schedule). DGL can also extract high-order and complex interaction patterns between different human states [9]. Higher-order and complex interactions between node neighborhoods can be captured by adding more GCN layers. Heterogeneous graphs can be constructed to represent both hierarchical human state interactions and heterogeneous human state interactions. For example, given the calendar information of a mobile user, we can use bipartite graphs to represent the user's daily schedule and further infer some relevant health outcomes, such as social anxiety.

In DGL, Graph Neural Networks (GNNs), which is a generalized version of Convolutional Neural Networks (CNNs) [6], can capture deep features of complex interactions and interdependence between human states. For example, GNNs can capture deep features of complex interactions and interdependence between human states. For example, CNN can capture high-level features from the multivariate time series data generated by the accelerometer, gyroscope, and magnetometer by learning the weights in the kernels. In the multivariate time series (MTS), we can treat each timestamp as a node and link two consecutive timestamps with an edge, which will give us a line structure. The high-level feature can also be extracted by using GNN with a graph representation of MTS. When we segment MTS into appropriate bins and construct graphs by regarding each bin as a node and transitions between bins as edges, GNN can capture the complex interaction and interdependence pattern of human states. DGL can generate semantic meaning (i.e., purpose, health state) for human behaviors [15]. For example, in the application of DGL to GPS location data, if a person's GPS trajectory graph is simple with a small number of nodes and a low node degree (like a line or circle), this person could be more likely to keep a daily route activity. If the GPS trajectory graph is complex with a large number of nodes and a high node degree, this could be more like having leisure time to explore the area.

DGL can infer demographic information which is helpful to develop personalized health intervention [16]. By leveraging calendar information, GRL can represent the schedule graph of people. This schedule graph can be utilized to generate embeddings to infer demographic information, such as gender, professional status, income, and so on. Furthermore, without demographic information input from participants, this embedding can be used to cluster people into different groups to develop a personalized healthcare service. DGL can be used to capture contextual information (i.e., location and time) about human behaviors. For example, students could use mail apps more frequently on campus during the daytime and use social apps and entertainment apps more frequently off campus after school. Spatiotemporal information can be embedded into latent representations by analyzing app usage trajectories and app attributes [17]. DGL also has good interpretability. Not only can each node have semantically interpretable information by evaluating node importance, but the global and local graph structures can also provide explainable information. For example, a complete graph of physical activity transitions can imply the robustness of a person. For GPS trajectory graph, highlighted sub-graphs can imply geographical information and also functionality of this sub-area (i.e. shopping mall and cinema are usually near each other).

The DGL for mobile health is investigated in this dissertation. The first study aims to make predictions of salivary cortisol levels of pancreatic cancer patients by using GRL to generate robust feature representations of human physical activities that are captured by mobile sensing [18]. The second study is to use a graph neural network (GNN) framework to infer the existence of influenza-like symptoms based on people's multi-modal daily mobile sensing data [19]. The third study, motivated by Multiple Instance Learning, proposes a semi-supervised graph instance transformer (semi-GIT) to overcome the label sparsity at different levels of granularity for more robust mental health inference [20]. Based on the empirical evaluation of the previously investigated deep graph learning systems by using mobile sensing data collected in the wild, we demonstrate that our deep graph learning systems can outperform traditional machine learning models with handcrafted feature engineering. Even though the superior performance of DGL was demonstrated in the previous studies, there are multiple challenges when applying DGL to real-world health inference problems. First and for most, even though mobile sensing has been shown as a promising solution in a plethora of healthcare-related fields with passively and unobtrusively collected data, wide adoption of mobile sensing applications remains challenging due to privacy concerns. Where the data is stored, who has access to them, and how they are used are among the typical aspects of users' concerns. What's more, training machine learning models, especially neural networks, requires a large quantity of labeled data and it is challenging to meet this requirement through tracking many users for extended periods of time, since this massive amount of data cannot be leveraged in supervised learning methods. Therefore, in the fourth study, we introduce incremental semi-supervised federated learning for privacy-preserving DGL in mobile health.

The remainder of this work is presented as follows. In chapter 2, we summarize recent works on DGL, Multi Instance Learning and Federated Learning. In chapter 3, we present our model of using GRL to predict saliva cortisol level for pancreatic cancer patients. In chapter 4, we present our multi-modal GNN for influenza-like symptom recognition. In chapter 6, we present our semi-supervised graph instance transformer for mental health inference. In chapter 5, we present our work of incremental federated learning for mobile health. And in chapter 7 and chapter 8 we provide discussion to illustrate the broader impact, limitation and future research directions, and after that we draw conclusion about this dissertation.

Chapter 2

Literature Review

2.1 Multi-modal Deep Learning

Unlike natural language processing and computer vision, most practical applications using sensing data are multi-modal, necessitating the deployment of properly designed deep learning algorithms to efficiently fuse these diverse data sources and obtain enhanced results. On a per-sample basis, Liu et al. suggested a novel multiplicative multimodal technique that may find more significant data modality [21]. They also tested a unique strategy for automatically selecting mixes of modalities in order to capture potential cross-modality correlations and interdependence. Using multiple physiological data sources such as ECG, accelerometer, and respiration data acquired from wearable devices, Chakraborty et al. suggested a multichannel convolutional neural network architecture for identifying state of mind [22]. In a tri-modal architecture, Kampman et al. integrated video clips, audio, and text data to predict Big Five Personality Trait scores [23]. They used stacked convolutional neural layers on each data modality and concatenated the outputs of all channels with fully connected layers to produce a 9.4% improvement over the best individual modality model. Li et al. preprocessed multi-channel EEG data into grid-like frames and constructed a hybrid deep learning model using CNN and RNN layers. They used this strategy in neurology and psychiatry to detect emotional disorders using emotion recognition tests. Salekin et al. used a similar CNN+RNN architecture to estimate newborn pain using video data, integrating facial expression and body movement signals into a three-channel model that included face, body, and face+body hybrid channels. [24].

2.2 Deep Graph Learning

Deep Graph Learning (DGL) has been used to encode multiple data types in machine learning tasks in order to tackle a variety of real-world challenges, including social network recommendation [25], drug-to-drug interaction prediction [26], and knowledge reasoning [27]. The authors of these publications used graph representation to express the complicated interdependence of distinct entities in a variety of challenges. Graph neural networks [6] GNN is capable of extracting graph structural information and composite linkages from graph-structured data automatically. For example, in the graph representation of road networks, road segments reflect the network's geographical information, while the edges connecting neighboring segments show their connectedness [28]. By learning spatiotemporal patterns buried in complicated road networks, the scientists used GNNs to forecast traffic flow. Road network embeddings are discovered using GNNs to construct semantic representations of road segments, allowing for the use of the structural functions of road networks in transportation planning [29]. Fan et al. encoded user-to-user and user-to-item links in social networks using graphs and used GNNs to produce social recommendations based on user interactions [25]. In this dissertation, we evaluate multiple the state-of-art GRL and GNN methods on different mobile sensing data collected in the wild, and demonstrate the outperformance of deep graph learning.

Graphs are unique data structures that consist of a set of nodes and edges and can represent numerous connected structures, such as social networks, protein-to-protein interactions, human skeletal systems, etc [6]. Given the graph structured data input that generated from non-Euclidean domain space, conventional DNNs (e.g., CNN and RNN) are unable to perform regular convolution operations or recurrent connection learning for long-term dependence. Graph neural networks (GNNs), as specific models in the deep learning family, can process graph-structured information by mapping graph input to numerical spaces. In this section, we provide briefly summary of current GNNs models. The diagrams of four types of GNNs model is shown in Fig.Figure 2.1. More comprehensive reviews of GNNs can be found in these works [9, 30, 6].

Preliminary of GNNs: given a graph, we usually denote the graph as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. Based on the node and edge construction, graphs can be categorized as follows:

- Directed/Undirected Graphs: in directed graphs, edges have directional information of pointing from one node to another, implying that messages can only transmit by following this direction. In undirected graphs, there is no such direction information in the edges, implying that messages can transmit in any direction.
- Weighted/Unweighted Graphs: weighted graphs refer to the graph with attributed edges, which determine how messages can transit through the networks. Unweighted graphs indicate a graph with unattributed edges, in which case we can just use an adjacency matrix with 0/1 entries to describe the connectivity of graphs.
- Homogeneous/Heterogeneous Graphs: homogeneous graphs include nodes and edges that have the same types, while heterogeneous graphs consist of nodes and edges that have the same types.
- Static/Dynamic Graphs: the topological structures of graphs cannot vary through time changes in static graphs. In dynamic graphs, their topological structures can change with time.

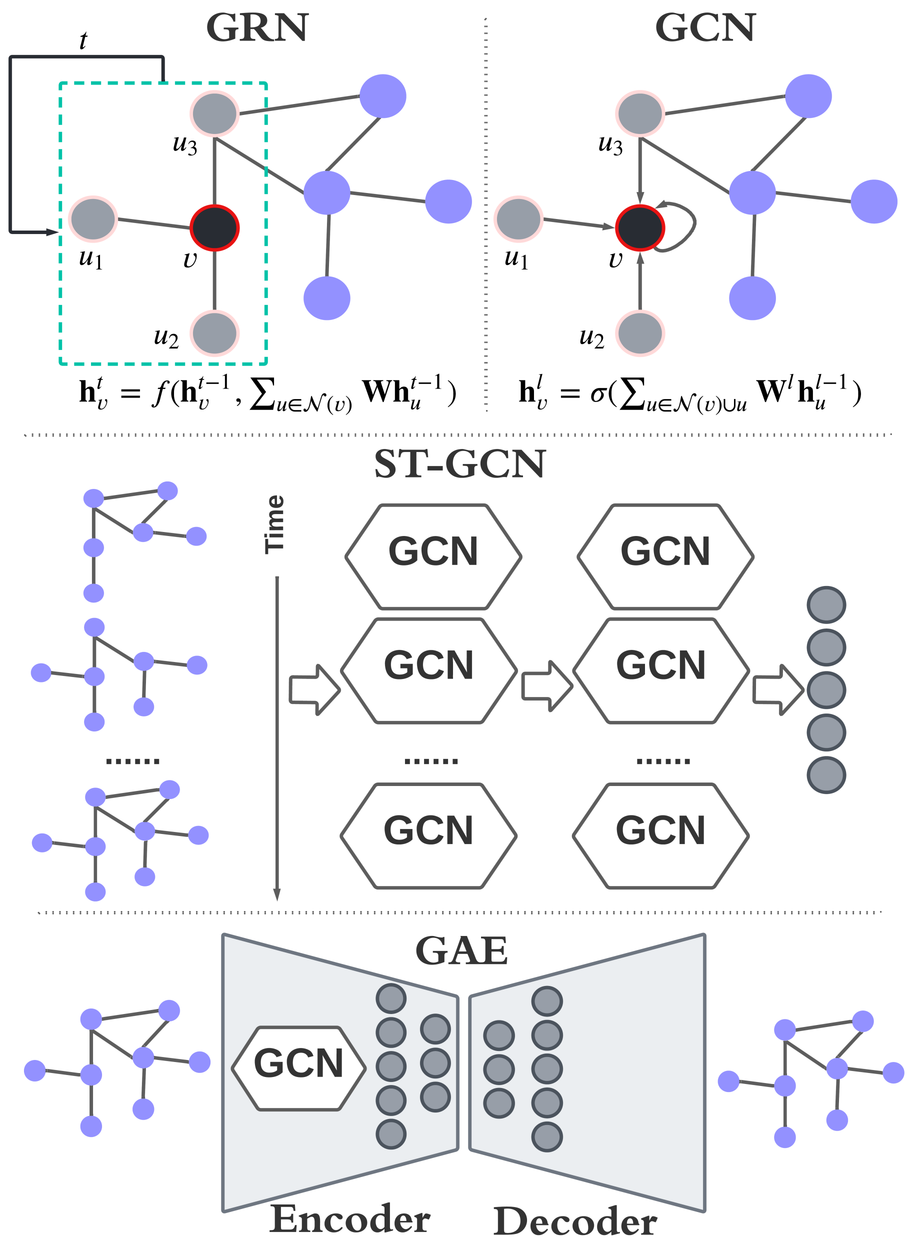
GRN
GCN
t
u_3
u_1
v
u_2
$\mathbf{h}_v^t = f(\mathbf{h}_v^{t-1}, \sum_{u\in\mathcal{N}(v)} \mathbf{W}\mathbf{h}_u^{t-1})$
$\mathbf{h}_v^l = \sigma(\sum_{u\in\mathcal{N}(v)\cup u} \mathbf{W}^l\mathbf{h}_u^{l-1})$
ST-GCN
Time
GCN
......
GAE
Encoder
Decoder

Figure 2.1: Graph Neural Network Architectures.

Due to being able to conduct convolution on graphs, GNNs can capture complex interactive relationships between nodes and produce high-level representations of the graph input. The core mechanism of GNNs is to iteratively aggregate neighborhood information for each node and then integrate the aggregated information into the original node representation through neural information propagation. Differentiated by the aggregation and node update operations, GNNs can be classified into the following categories [6]:

- Graph Recurrent Neural Networks (GRNs): GRNs, pioneer works of GNNs, apply recurrent neural framework to learn node representations with shared function. For example, Gated GNN (GGNN) [31] applies a gated recurrent unit (GRU) [32] as a shared parametrized function to update node representation by considering previous node hidden state and the hidden state of its neighbors.

- Graph Convolutional Neural Networks (GCNs): GCNs define a neighborhood information aggregation process to update node representation, and stack multiple graph convolutional layers to generate high-level node embeddings. GCNs can be classified to spectral-based and spatial-based. Rely on graph signal processing algorithm, spectral-based GCNs perform graph convolutions by processing the input graph signals through a set of learnable filters to aggregate information [9]. For example, ChebNet [33] approximates the spectral filter by the Chebyshev polynomials of the diagonal matrix of eigenvalues, rather than explicitly computing the graph Fourier transform. Spatial-based GCNs consider node's spatial relation to define graph convolutions. For example, NN4G [34] accumulates node's neighborhood information as graph convolutions and uses residual connections and skip connections to reduce long-term forgetting over stacking of layers.

- Spatio-Temporal Graph Neural Networks (ST-GNNs): Graphs in multiple real-world situation have both spatial and temporal variations, such as traffic network, social network and skeleton-based human actions. In the dynamics of graphs, their topological structures change over time with varying node distributions and different edge connecting relations. ST-GNNs aim to model the spatial and temporal dependencies in dynamic graphs. A dynamic graph can be represented as an ordered list or an asynchronous set of graphs. In the design of ST-GNNs, GCNs and RNNs are usually integrated such that topological features can be extracted from the graph input by using GCNs and the temporal dependencies between graphs can be capture by using RNNs. For example, Yu et al. [35] propose Spatio-Temporal Graph Convolutional Networks (STGCN) for traffic forecasting. STGCN integrates temporal gated convolution layers and spatial graph convolution layer to fuse features from both spatial and temporal domains.

- Graph Autoencoders (GAEs): GAEs are highly correlated to graph representation learning, which aims to preserve the high-dimensional complex graph information involving node features and link structures in a low-dimensional embedding space. The GAEs are similar to other types autoencoders (AE). They both consist of a encoder and a decoder, where the encoder compresses information into a latent space and the decoder try to reconstruct the original feature or structural from the latent space. Kipf et al. [36] proposed the vanilla GAE/VGAE structure, which predict the links between nodes as the reconstruction process. Modern GAE models utilized more useful information in the network like semantic context conditions [37], neighborhood information [38], and others [39, 40].

2.3 Federated Learning

As more and more fine-grained personal data are being collected and applied in different health inference tasks, privacy protection has become an essential demand by the data owners. However, many traditional privacy preserving methods, such as k-anonymity, suffer from severe accuracy loss due to inevitable information loss and corrupted data utility [41]. To address this problem, the concept of Federated Learning (FL) is first proposed to train high-quality shared global model by leveraging local data from distributed clients without explicitly exposing private data to central servers or other clients [42]. FL has been widely applied in many fields such as natural language processing (NLP) [43], Internet of Things [44], etc. Due to its non-exposure of private data, FL has been proved effective in information sensitive fields (i.e., healthcare[45]). For example, Vaid et al. predicted mortality in hospitalized COVID-19 patients within seven days by logistic regression and Multi-Layer Perceptron (MLP) federated models [46]. Motivated by the classic FL framework, in this dissertation, we propose FedMobile which is an incremental semi-supervised federated learning algorithm that can train models periodically by using newly incoming batch of mobile sensing data in a decentralized online fashion.

2.4 Multiple Instance Learning

Multiple Instance Learning (MIL) is a form of weakly supervised learning, which assigns labels to multiple instance bags (i.e., each bag contains multiple instances rather than single instances). This formulation naturally fits various problems in image classification [47], and document classification [48], due to the absence of labels for the vast majority of learning instances. MIL has shown remarkable effectiveness on a variety of tasks using datasets with sparse labeling. Babenko et al. used a machine learning technique rather than typical supervised learning to produce greater performance in a real-time object tracking test [49]; Sun et. al. combined MIL with CNNs and reached state-of-the-art result on both low resolution (CIFAR) and high resolution (ILSVRC2015) image classification tasks [49]. To further accommodate the permutation invariance of bags/sets of instances, Zaheer et al. suggested a deep network architecture capable of operating sets in machine learning tasks without regard for the order of the elements in the sets [50]. While MIL is widely used in picture and text classification applications, little work has been done integrating MIL with graph type data. In this dissertation, we integrate GNN, Set Transformer[51], and MIL to address weakly supervised graph instance bag inputs.

Chapter 3

Using Graph Representation Learning to Predict Salivary Cortisol Levels in Pancreatic Cancer Patients

3.1 Background and Motivation

Pancreatic cancer is one of the worst types of cancer, with a 5-year survival rate of about 9%. Although it is the eleventh most prevalent kind of cancer globally, it is the seventh largest cause of cancer-related mortality [52]. Pancreatic cancer incidence and death rates have climbed dramatically in the United States of America since 2000, according to research. [53]. Notably, early evidence indicates that cortisol may have an effect on how pancreatic tumors react to cancer therapy. [54].

Hormones in the body act as chemical messengers, facilitating complicated processes such as immune system function and behavior. Cortisol, a glucocorticoid hormone, is a major product of the hypothalamic-pituitary-adrenal (HPA) axis and is necessary for the ”fight or flight” response to occur. Cortisol is released in reaction to stimuli, which boosts glucose levels in the circulation, providing an instant energy source for the body’s big muscles. While cortisol primes the body for action, chronically elevated cortisol levels may impair immune system function, cause gastrointestinal disorders, cardiovascular disease, infertility, and sleeplessness [55]. This is especially troublesome while treating pancreatic cancer, since an increase in cortisol levels has been shown to accelerate tumor development [56].

Although no research have been conducted on this subject, mobile technology may offer a low-cost and accessible method for continually estimating cortisol levels in the body. Existing research on the use of mobile sensing in health has mostly relied on sensors to approximate behaviors via the use of a tiered, hierarchical framework in which features are taken from raw sensor data and converted to markers of behavioral states[57]. The theoretical foundation for approximating hormone levels using sensors is based on this hierarchical architecture. Cortisol levels typically peak in the morning and subsequently decline during the day, however experiential variables like as physical activity and reaction to acute and chronic stresses contribute to individual variability in cortisol levels throughout the day [58]. Cortisol levels are therefore the result of a known diurnal pattern and behavioral states (e.g., physical activity, stress reaction) that a wearable device can monitor. The behavioral traits collected from sensor data may be utilized to estimate cortisol levels in patients with pancreatic cancer.

At the moment, the major techniques for cortisol measurement are blood and saliva samples, which are cumbersome to obtain and expensive to evaluate. The difficulty associated with collecting biospecimens is magnified in people with pancreatic cancer due to the disease's influence on their health, everyday functioning, and overall quality of life. Biospecimen collection is also unfeasible for real-time cortisol monitoring due to the time required to finish tests. Monitoring cortisol levels using mobile technology has the potential to increase our knowledge of the tumor development trajectory and the possible role it plays in anticancer treatment response.

The purpose of this research is to provide a method for correlating passively detected raw actigraphy data with salivary cortisol levels in newly diagnosed patients with pancreatic cancer. The ultimate purpose of this study is to show that, in place of serum or saliva collection, passively obtained data may be utilized to estimate in situ the underlying circulating cortisol level in cancer patients. No published study has addressed the feasibility of predicting underlying hormone levels using passively acquired activity data.

3.2 Method

We present the general predictive modeling technique in this part, which takes raw sensor data as input and trains a machine learning model to forecast salivary cortisol levels. Additionally, we give thorough details regarding the handcrafted feature engineering techniques used in this work. Finally, we will discuss Graph Representation Learning (GRL) and the techniques associated with it including Graph2Vec [59], FeatherGraph [60], GeoScattering [61] and NetLSD [62] which were implemented in this study.

As shown in Figure 3.7, the predictive modeling process consists of 4 steps. The first phase involves passively generating sensor data from users' ActiGraph devices, which include an accelerometer, a light sensor, and an inclinometer. The raw sensor data is preprocessed in the second stage utilizing time-window segmentation to remove noise from the data. The devices are configured to sample at a predefined period (e.g., 30 seconds, 1 minute), which was conducted in this research using the ActiLife software [1]. ActiLife enables the retrieval, sampling, aggregation, and synchronization of raw actigraphy data. We used a one-minute epoch to collect and synchronize the raw actigraphy data in this investigation. In step three, numerous feature engineering approaches such as handcrafted engineering (i.e. time domain features, frequency domain features) and Graph Representation Learning may be employed to extract relevant features. The retrieved characteristics are then employed in the predictive modeling training procedure in the fourth stage. To predict salivary cortisol levels, the machine learning model with the lowest testing error is

[1]https://www.actigraphcorp.com/actilife/

chosen.

Handcrafted feature engineering can be applied to extract temporal and spectral features from raw sensor data manually, as shown in TABLE Table 3.1. Generally, given a multi-dimensional time series data $\mathbf{X} \in \mathbb{R}^m$, $\mathbf{x}_i$, where $i = 1, 2, ..., m$ and m is the dimensions of Actigraphy data, represents one dimensional time series data collected from one of the sensors. And $\mathbf{x}_{it}$ represents the ith dimension of the Actigraphy Data at time t. Let t_s represents the time when we record participants' salivary cortisol levels. Then we set the time $t_s^{'}$, where $t_s^{'} \in [0, t_s)$, such that, in the time window $t_s - t_s^{'}$, handcrafted-featurized value $\mathbf{H}$ can be extracted by applying feature engineering functions $\mathcal{F}(\mathbf{X}_{t_s':t_s}) = [f_1(\mathbf{X}_{t_s':t_s}), ..., f_k(\mathbf{X}_{t_s':t_s})]$, where $\mathbf{H} \in \mathbb{R}^{km}$ and k is the number of the features extracted from each sensor and $\mathbf{X}_{t_s':t_s}$ is the segment of the time series framed by the time window. The handcrafted features extracted from time and frequency domains are shown in TABLE Table 3.1. The handmade features found were chosen based on earlier research demonstrating the potential usefulness of mean values, variance indicators, as well as max/min and distribution symmetry in data reflecting behavior. Due to the fact that salivary cortisol levels are continuous, our predictive modeling aim is to develop regression models that fit cortisol levels with the least amount of testing error. Automatic Feature Engineering: Learning to Represent Graphs GRL may be used to extract structural and synthetic characteristics in lower dimensions in order to capture the interaction and transitions between distinct perceived behaviors[63]. GRL can be extended as a technique for learning a mapping that embeds graphs into low-dimensional numerical vector spaces in such a way that the intrinsic graph features are preserved best [8]. The embedded features created by GRL may then be used to develop predictive models in downstream machine learning models. For GRL in this study as shown in Figure 3.1, given a sequence of time series data, We use G-Means clustering to automatically maximize the amount of cluster labels to give to each time point [64], and then convert the sequence of cluster labels into undirected and unweighted graphs that reflect state transitions in the time series data. Then, using graph

Table 3.1: Handcrafted Features

Domains	Features
Time Domain	Mean
	Standard Deviation
	Maximum
	Minimum
	Peak to Peak
	Shannon Entropy
	Mean Absolute Difference
Frequency Domain	# Maxpeaks
	# Minpeaks
	Kurtosis
	Skewness
	Absolute Energy

inputs, GRL algorithms are trained to build embedding vectors for use as independent variables in modeling regressors. The motivation of including Graph2Vec, FeatherGraph, GeoScattering and NetLSD are the following: 1) Graph2Vec and NetLSD is the representative methods based random walk and matrix factorization respectively; 2) FeatherGraph [60] and GeoScattering [61] have achieved state-of-art result in graph embedding task The GRL methods implemented in this study are described below.

Graph2Vec learns k-dimensional numerical representations from graphs [65]. Assume a set of undirected and unweighted graphs $\mathcal{G} = \{\mathbf{G}_1, ..., \mathbf{G}_n\}$, where $\mathbf{G}_i$=$\{\mathbf{V}_i, \mathbf{E}_i, \lambda_i\}$, $\mathbf{V}_i$ is a set of nodes, $\mathbf{E} \in (\mathbf{V}_i \times \mathbf{V}_i)$ is a set of edges, and $\lambda_i : \mathbf{V}_i \to \mathcal{L}$ assigning an unique label from vocabulary $\mathcal{L}$ to each node in $\mathbf{V}_i$ for $i = 1, ..., n$. Graph2Vec performs as a function $f(\mathbf{G})$ generating the output in k-dimensional vector space, analogous to Doc2vec [59] which maps documents to numerical spaces. Rooted subgraphs are sampled and relabeled by Weisfeiler-Lehman (WL) kernel to train a skipgram model such that $\mathbf{G}_i$ and $\mathbf{G}_j$ are more similar the $f(\mathbf{G}_i)$ and $f(\mathbf{G}_j)$ are closer in k-dimensional space[65] .

FeatherGraph uses feature functions of node features in conjunction with random walk weights to feature each node neighborhood and then uses mean pooling to average node

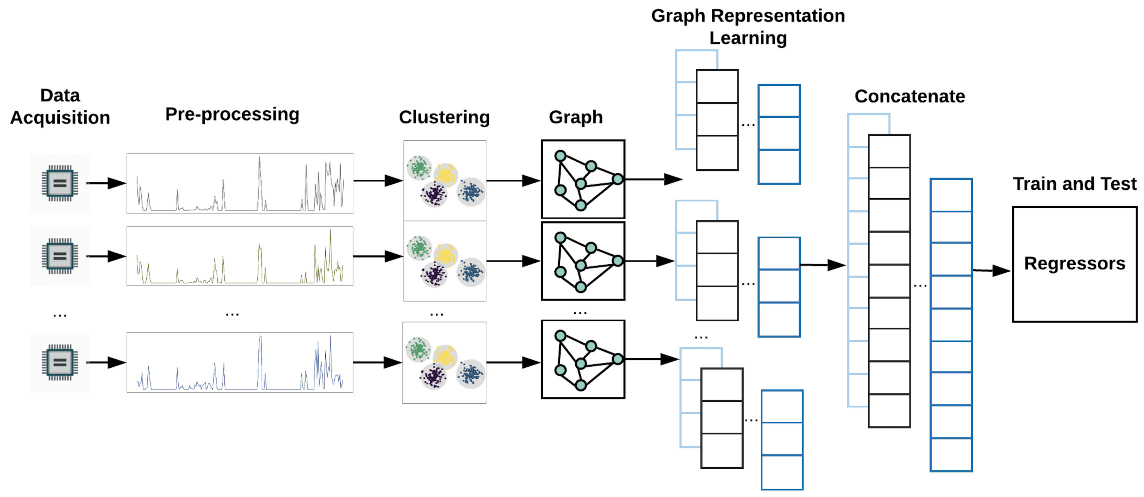

Figure 3.1: Feature Extraction by Graph Representation Learning.

level features to get graph level features. [60]. Assume an unweighted and undirected graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ is the set of nodes and $\mathbf{E}$ is the set of edges. For node $v \in \mathbf{V}$, we describe a node feature as a random variable X and specify feature vector $\mathbf{x}_v$, where $\mathbf{x}_v \in \mathbb{R}^{|\mathbf{V}|}$. Given the source node u and target node w, where $u, w \in \mathbf{V}$ and $\sum_{w \in \mathbf{V}} P(w|u) = 1$, and evaluation point $\theta \in \mathbb{R}$ we define real and imaginary part of the $r - scale$ random walk weighted feature function for node u as

$$Re(E(e^{i\theta X}|G, u, r)) = \sum_{w \in \mathbf{V}} \hat{\mathbf{A}}^r_{u,w} cos(\theta \mathbf{x}_w), \tag{3.1}$$

$$Im(E(e^{i\theta X}|G, u, r)) = \sum_{w \in \mathbf{V}} \hat{\mathbf{A}}^r_{u,w} sin(\theta \mathbf{x}_w) \tag{3.2}$$

where $\hat{\mathbf{A}}^r_{u,w} = (\mathbf{D}^{-1}\mathbf{A})^r$. Then the concatenated vector of $Re(\bullet)$ and $Im(\bullet)$ will be used as feature vector for the node u [60].

GeoScattering can extract numerical embeddings from graphs by using the moments of wavelet transformed features [60]. Let $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{W})$ be a graph, where $\mathbf{V}$ is the set of vertices $\{v_1, v_2, ..., v_n\}$, $\mathbf{E}$ is the set of edges (v_l, v_m) with $1 \leq l, m \leq n$, and $\mathbf{W}$ is the weight matrix $\mathbf{W} = \{w(v_l, v_m) = 1, : (v_l, v_m) \in \mathbf{E}\}$. Then define $n \times n$ lazy random walk matrix as $\mathbf{P} = \frac{1}{2}(\mathbf{I} + \mathbf{A}\mathbf{D}^{-1})$, where $\mathbf{I}$ is the identity matrix of $\mathbf{G}$, $\mathbf{A}$ is the adjacency matrix and $\mathbf{D}$ is the diagonal degree matrix. With the definition of wavelet transform matrix at scale 2^j,

$$\mathbf{\Psi}_0 = \mathbf{I} - \mathbf{P},\ \mathbf{\Psi}_j = \mathbf{P}^{2^{j-1}}(\mathbf{I} - \mathbf{P}^{2^{j-1}}), j \geq 1, \tag{3.3}$$

and signals $\mathbf{x}(v_l)) = deg(v_l)$ defined on graph $\mathbf{G}$, the "zero" order scattering moments,first order geometric scattering moment and second order geometric scattering moment can be defined respectively as

$$S\mathbf{x}(q) = \sum_{l=1}^{n} \mathbf{x}(v_l)^q, \tag{3.4}$$

$$S\mathbf{x}(j,q) = \sum_{l=1}^{n} |\mathbf{\Psi}_j \mathbf{x}(v_l)|^q, \tag{3.5}$$

$$S\mathbf{x}(j,j^{'},q) = \sum_{l=1}^{n} |\mathbf{\Psi}_{j'} |\mathbf{\Psi}_j \mathbf{x}(v_l)||^q \tag{3.6}$$

, for $1 \leq j < j^{'} \leq J$ and $1 \leq q \leq Q$. Finally, the collection $S\mathbf{x} = \{S\mathbf{x}(q), S\mathbf{x}(j,q),$ $S\mathbf{x}(j,j,^{'} q)\}$ provides the set of features to describe graph $\mathbf{G}$ [60].

NetLSD generates eigenvectors from the factorized normalized Laplacian matrix of the input graph's adjacency matrix and calculates the heat kernel trace by using the eigenvectors to represent the input graph [62]. Consider an undirected graph and unweighted graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ is the set of vertices and $\mathbf{E}$ is the set of edges. Then denote the adjacency matrix of a graph $\mathbf{G}$ as $\mathbf{A}$ and diagonal matrix as $\mathbf{D}$ with the degree of node i as entry D_{ii}, a graph's normalized Laplacian is the matrix can be expressed as $\mathcal{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$. $\mathbf{L}$ can be factorized as $\mathbf{L} = \Phi\Lambda\Phi^T$, where Λ is a diagonal matrix on the sorted eigenvalues $\lambda_1 \leq ... \leq \lambda_n$ of which $\phi_1, ..., .\phi_n$ are the corresponding eigenvectors. Then the $n \times n$ heat kernel matrix at each vertex at time t can be calculated by

$$H_t = \sum_{j=1}^{n} e^{-t\lambda_j} \phi_j \phi_j^T \tag{3.7}$$

, where $(\mathbf{H})_{ij}$ shows how much heat transferred from v_i to v_j at time t. Finally a heat trace signature of graph $\mathbf{G}$ can be used the graph embedding, which consists of heat traces $tr(\mathbf{H}_t)$ at different time t [62].

3.3 Data and Experiments

The inclusion criteria included:(1) age $\geqslant$ 18 years; (2) insomnia symptoms for $\geqslant$6 months; (3) sleep $\leqslant$ 6.5 hours per night; (4) sleep disturbances (or associated daytime fatigue) that cause significant distress or impairment in social, occupational, or other areas of functioning, as determined by at least a a subthreshold level of severity on the Insomnia Severity Inventory [66] (i.e., a score of $\geqslant$ 8 out of 22; (5) ability to provide informed consent; (6) histological or cytological proof of pancreatic adenocarcinoma; (7) borderline resectable, locally advanced or metastatic pancreatic cancer and are not candidates for upfront, curative surgical resection; (8) have not received any therapy (cytotoxic chemotherapy, immunotherapy, radiotherapy, biologic therapy, or other investigational therapy directed towards their pancreatic cancer) for their cancer prior to enrollment; and (9) are candidates for systemic therapy based on investigator evaluation. Sleep-related criteria were included since one of the study's primary objectives was to better understand the association between sleeplessness symptoms and pancreatic tumor progression. To account for any confounding variables that may artificially inflate salivary cortisol levels, participants who were active smokers (and who expressed an unwillingness to stop smoking during the research time) were excluded from the trial, along with those on corticosteroids. We recruited ten persons (6 males and four females) newly diagnosed with pancreatic cancer from a cancer clinic using these criteria. The participants' average age is 64.6 years. Nine individuals identified as White and one as mixed. The average pm standard deviation of participants' sleep time is 411 pm 135.53 minutes. On average, participants were diagnosed with pancreatic cancer 15 days before to enrollment in the research. The detailed demographics information is

Table 3.2: Demographics Information of the Participants

Age	Sex	Ethnicity	Income ($)	Cancer Stage
75	male	white	$100,000+$	1b
58	female	white	$50,000-75,000$	3
67	male	multiracial	$30,000-50,000c$	2
81	female	white	$\leq 30,000$	NA
50	male	white	$\leq 30,000$	NA
71	female	white	$100,000+$	NA
47	male	white	$50,000-75,000$	2
76	male	white	$75,000-100,000$	NA
66	male	white	$75,000-100,000$	2
55	female	white	$100,000+$	1

shown in Table Table 3.2.

Clinical study coordinators contacted participants during a routine medical appointment. After participants signed the agreement form, coordinators demonstrated how to appropriately collect and retain saliva samples (through salivettes) to achieve the most accurate salivary cortisol measurements. For five consecutive days, participants were instructed to take three samples each day (at wake, 5pm, and 9pm) in order to capture the diurnal variability in cortisol levels[55]. Saliva samples were obtained on average at the following times: 7:51 AM, 5:43 PM, and 9:43 PM. For all samples, participants were told not to wash their teeth, consume coffee, engage in strenuous activity, or have a substantial meal within an hour before to the collection of the sample. Each sample was obtained by placing a sterile cotton swab from the salivette tube beneath the participant's tongue for 1-2 minutes and then placing it back into the salivette tube. After collecting the sample, participants documented the time and date of collection and immediately placed all samples in their refrigerator. Additionally, participants were requested to wear an actigraphy device (ActiGraph GT9X) on their non-dominant hand for the duration of the research. After the

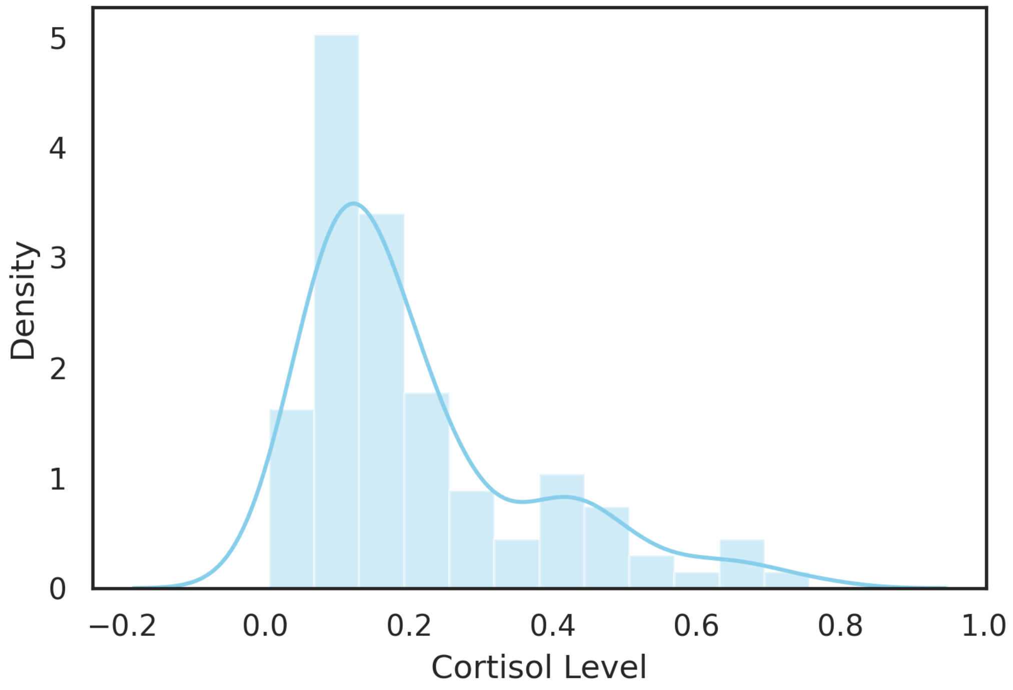

Figure 3.2: Density Plot of Saliva Cortisol Level

fifth day of collection, participants got a postage-paid mailer to return their saliva samples and actigraph. Salivettes were kept at -80 degrees Celsius until they were sent for analysis.

Saliva tests were performed by a third-party vendor (Salimetrics.com), which specializes in providing precise findings for biomarkers detected in saliva samples. Prior to completing the experiment, samples were thawed to room temperature, vortexed, and then centrifuged for 15 minutes at about 3,000 RPM (1,500 x g). Salivary cortisol levels were determined using a high-sensitivity enzyme immunoassay. Each determination required a sample test volume of $25mul$ of saliva. The assay has a lower limit of sensitivity of $0.007mug/dL$, a standard curve range of $0.012 - 3.0mug/dL$, an average intra-assay coefficient of variation of 4.60 percent, and an average inter-assay coefficient of variation of 6.00 percent, which meets the manufacturers' criteria for accuracy and repeatability in salivary bioscience and exceeds the applicable NIH guidelines for Increasing Reproducibility through Rigor The density plot of cortisol level is shown in Figure 3.2.

3.4 Results and Analysis

In this section, we present the implementation and results of our predictive modeling approaches to predict salivary cortisol levels at each time point. We also discuss hyperparameter selection (window size and sensor combination), graph construction and model evaluation.

Experiments

Following the predictive modeling process in Figure 3.7, we first collected multi-dimensional time series Actigraphy data, which are pre-processed raw sensor data generated from ActiGraph sensors.

For feature engineering, we compared two strategies: handcrafted feature engineering and graph representation learning. For handcrafted feature engineering, we applied the feature extraction functions as shown in TABLE Table 3.1 and then scale features into unit variance. To learn about graph representations, we first conduct G-means clustering on the time series data and then label each time series data point with a cluster label. As shown in Figure 3.3, each graph's vertices correspond to distinct clusters (participants' states) in the related input time series data, while the edges correspond to transitions between the vertices (states). The vertices indicate clusters with semantic meanings that may be deduced from the cluster centers as shown in TABLE Table 3.3.

The vertices in Acceleration graph, as shown in Figure 3.3, indicate physical activities such as walking, sitting, or running. For instance, the vertex 2 of Acceleration clustering has low coordinate values, indicating that the activity has a limited range of body movement (such as walk or sit). The transition between vertex 2 and vertex 3 is shown by the edge between them. Similarly, each vertex in the graph of $Steps$ indicates a cluster of the number of steps taken in one minute (since we sample raw sensor data with processing epoch of 1 minute). The edges between vertex 2 and vertex 4 implies the participant change their

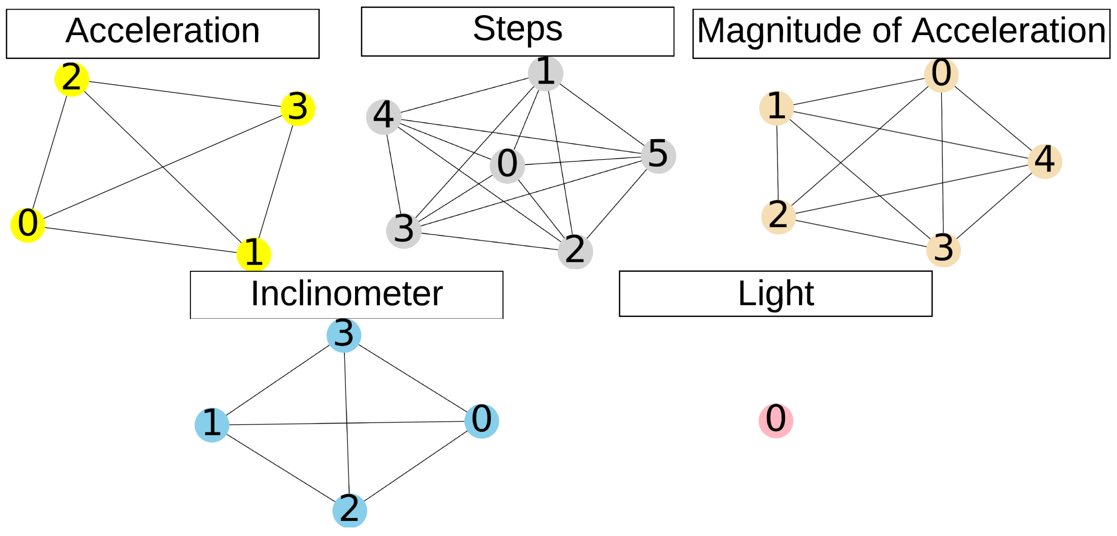

Figure 3.3: Sampled Graphs Visualization from one Observation in the Actigraphy Data. The Acceleration graph represents acceleration states and state transitions, generated by using G-means clustering for Axis x, Axis y, Axis z 3-D dimensional accelerometer data. Magnitude of acceleration is vector magnitude of accelerometer's Axis x, Axis y, Axis z data , which equals to $\sqrt{[(Axis\ x)^2 + (Axis\ y)^2 + (Axis\ z)^2]}$. And Magnitude of Acceleration plot shows the state transitions of the magnitude. Steps represents the step count state transitions. Inclinometer and Light represents the inclinometer and light sensor data state transitions respectively.

average step frequency from $0.17/minute$ to $12.44/minute$ as shown in TABLE Table 3.3. This illustration can also be applied to explain Magnitude graph in Figure 3.3: the vertices indicate activities of varying degrees of vigorousness, while the edges reflect the transitions between them; the lower the cluster center value in the Magnitude graph, the more sedentary the activity represented by the vertex should be. The vertices in $Inclinometer$, as shown in Figure 3.3 represent the postures of $standing$, $lying$, $sitting$ and off(inactive) which are detected by using Inclinometer. The categorization accuracy of postures using the Inclinometer was confirmed using approximately 90% [67], as a result, we are not need to use G-means to give cluster labels. Furthermore, the edges represent the transitions between observed postures. For the light sensor, the vertex in Light graph shown in Figure 3.3 represent the cluster of the unit of illumination. High unit of illumination implies a bright environment, such as outdoor, and low unit of illumination implies a dark environment,

Table 3.3: Sampled Cluster Centers / Representation. The entries in the the columns of Acceleration, Steps, Magnitude of Acceleration and light represent the values of cluster centers for each cluster; and the entries in Inclinometer represent the label of each vertex. To transfer time series data to the graphs, we firstly applied G-means clustering to automatically generate the best cluster allocation. For example, the optimal number of clusters for acclerometer is 4 and the optimal number of clusters for Light is 1, which could because the light sensor did not have too much variations during a day and the patients spend most of the times in the hospitals.

Vertex	Acceleration	Steps	Magnitude of Acceleration	Inclinometer	Light
0	(257, 264, 365)	2	1736	Sitting	0
1	(523, 513, 686)	8	25	Standing	NA
2	(16, 13, 25)	0	806	Lying	NA
3	(802, 930, 991)	6	417	Inactive	NA
4	NA	12	1208	NA	NA
5	NA	4	NA	NA	NA

such as home. As show in TABLE Table 3.3, the value of this single $Light$ cluster center is 0, indicating that the participant is in a dimly lit setting with much variance. Then, we feed the graphs into the graph representation learning algorithm in order to produce unsupervised learnt graph features.

Given that saliva cortisol level is recorded at time t_s and ActGraphy data $\mathbf{X}$, the window size w defines the feature space $X_{t_s-w:t_s}$ such that we extract features from $\mathbf{X}$ within the time range between the time $t_s - w$ and the time when cortisol level is recorded. To determine the optimal window size and sensor combination, we evaluate Random Forest performance with various sensor combinations (Acc+Inclin+Light, Acc+Inclin, and Acc) and increasing window sizes (from 0.5 hour to 12 hour increase by 0.5 hour) using 10-fold cross validation with Mean Absolute Error (MAE). As shown in Figure 3.4, the optimal selection of window size and sensor combinations for handcrafted feature engineering is $9.5\ hr$ with Accelerometer only, and the lowest MAE is 0.090 as shown in TABLE Table 3.4. As shown in Figure 3.5, the optimal selection of window size and sensor combinations for graph representation learning by using GeoScattering is $9.5\ hr$ with Accelerometer only, and the lowest MAE is 0.087 as shown in TABLE Table 3.4.

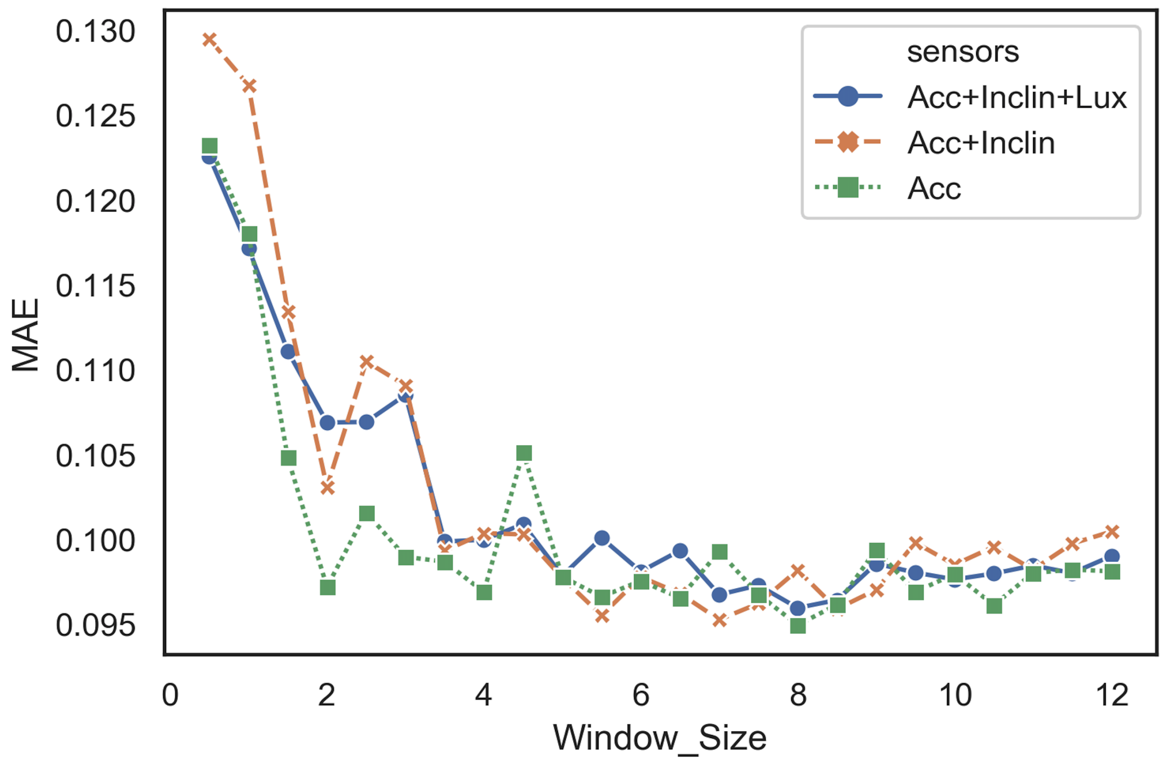

Figure 3.4: Sensors and Window Size Selection for Handcraft Feature Extraction

Table 3.4: Window Size (hr) Selection for Handcrafted and Graph Representation Learning Feature Extraction by Random Forest

Sensors	**Handcrafted**		**GeoScattering**	
	Window Size	MAE	Window Size	MAE
Acc+Inclin+Light	8.0	0.091	4.5	0.090
Acc+Inclin	7.5	0.091	6.0	0.089
Acc	**9.5**	**0.090**	**9.5**	**0.087**

To investigate the association between activities and cortisol levels, we used a Random Forest model on handmade features (Acc+Inclin+Light) and picked the top 15 features from 120 features sorted according to feature relevance, as shown in Figure 3.6. These 15 most important features, $peak\ to\ peak$, $absolute\ energy$ and max all of these measures quantify the amplitude of motion signals, indicating that physical activity habits are significant predictors of cortisol levels. Except for the inclinometer, the chosen features from the accelerometer have a greater priority than the selected features from the inclinometer. $Inclinometer - Lie - absolute\ energy$. This implies that accelerometer features are strong

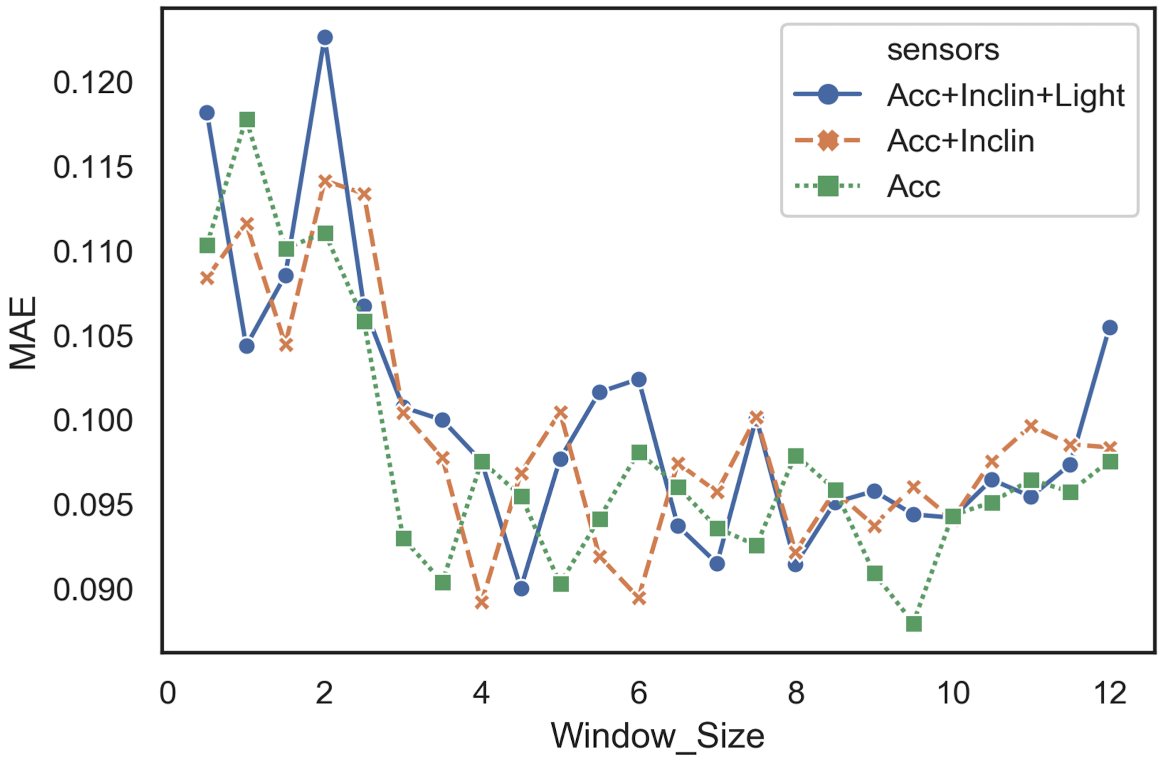

Figure 3.5: Sensors and Window Size Selection for Graph Representation Learning Feature Extraction

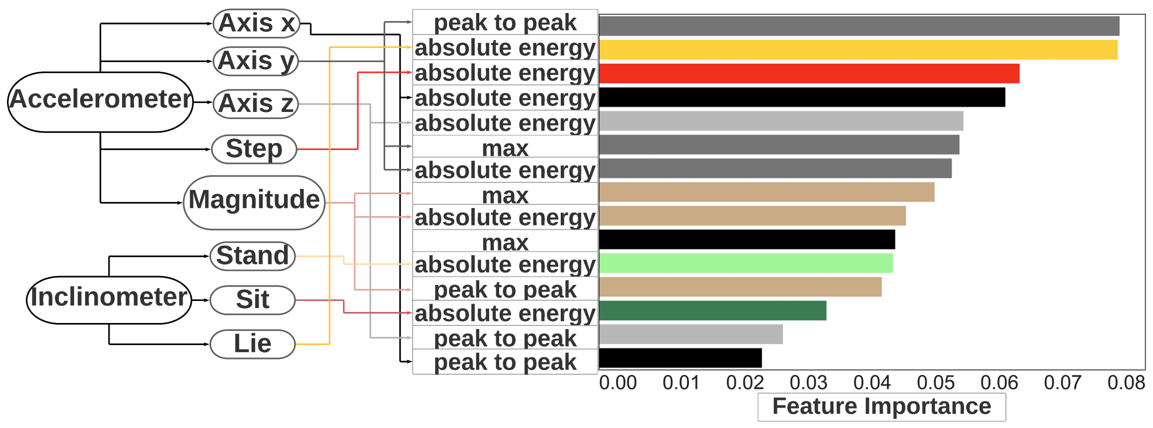

Figure 3.6: Random Forest Feature Importance: the features from the same Actigraphy sensor are decorated with the same color.

predictors of cortisol levels than are light and inclinometer sensors.

We deployed numerous machine learning algorithms in the downstream predictive modeling process to evaluate the performance of handmade feature engineering and graph representation learning techniques, as shown in Figure 3.7. To demonstrate the general pre-

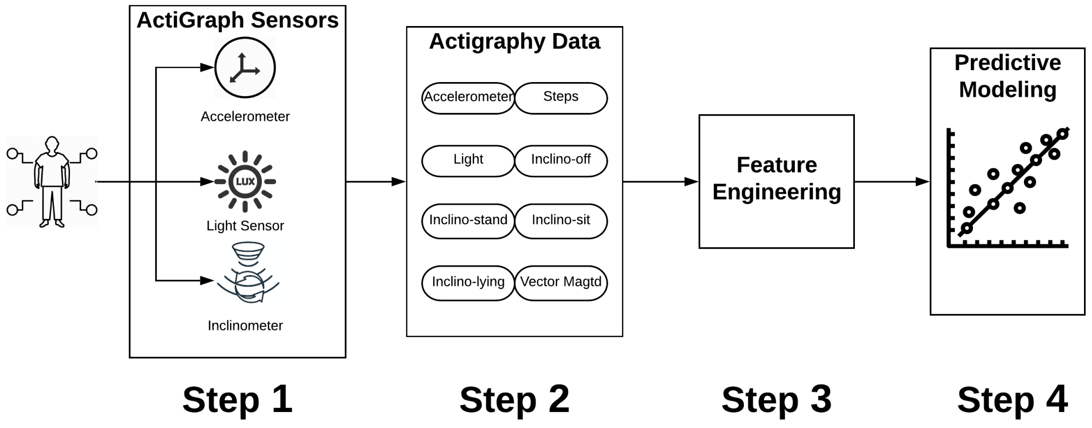

Figure 3.7: Predictive Modeling Pipeline of Salivary Cortisol Levels-A General process.

dictive framework, as shown in Figure 3.7, we applied standard regression models to our dataset. These regression models include Lasso Regression [68], Ridge Regression [69], Regression of Support Vector Machine (SVM) [70], Regression of Random Forest (RF) [71], Regression of Xgboost (XGB) [72] and Regression of Multi Layer Perceptron (MLP) [73]. Thus for the combinations of feature engineering methods and machine learning models training with the features selected by $9.5\ hr$ window size, we use 10-fold cross valuation with MAE, RMSE and MAPE to assess the predictive performance of these combinations.

The results of comparing the feature engineering methods across the different machine learning models are shown in TABLE Table 3.5. We use the mean value of cortisol as the vanilla model with 10 fold cross validation, the mean MAE $\pm$ std of the vanilla model is 0.126$\pm$0.018, RMSE 0.158 $\pm$0.034 is and MAPE is 59.571 $\pm$9.612. We use the machine learning models with handcrafted features as baseline models. Values in bold in TABLE Table 3.5 indicate the optimal fit between the feature engineering methods (i.e., handcrafted and automatic feature engineering approaches) and machine learning models (i.e., Lasso, Ridge, SVM, Random Forest, Xgboost, MLP). In general, machine learning models with handmade and automated feature engineering were able to predict cortisol hormone levels with usually low MAE, RMSE, and MAPE values. For each machine learning model individually, graph representation learning approaches, particularly FeatherGraph

Table 3.5: MAE, RMSE, MAPE (mean ± std) comparison between Handcrafted and Graph Representation Learning Feature Extraction

Metrics	**Featurization**	**Machine Learning Models**					
		Lasso	Ridge	SVM	Random Forest	Xgboost	MLP
MAE	Handcrafted	0.125±0.026	0.106±0.057	0.104±0.014	0.090±0.017	0.099±0.029	0.112±0.034
	Graph2Vec	0.124±0.032	0.124±0.038	0.118±0.022	0.092±0.026	0.091±0.031	0.125±0.030
	FeatherGraph	0.124±0.032	0.096±0.020	**0.098±0.018**	0.090±0.019	0.089±0.026	**0.097±0.028**
	GeoScattering	**0.123±0.025**	**0.095±0.022**	0.108±0.013	**0.087±0.029**	**0.085±0.037**	0.119±0.038
	NetLSD	0.124±0.032	0.122±0.020	0.118±0.028	0.091±0.026	0.093±0.017	0.128±0.032
RMSE	Handcrafted	0.159±0.014	0.141±0.021	0.153±0.017	0.140±0.013	0.139±0.019	0.169±0.059
	Graph2Vec	0.158±0.032	0.156±0.044	0.155±0.049	0.146±0.035	0.134±0.039	0.159±0.043
	FeatherGraph	0.156±0.042	**0.131±0.013**	**0.132±0.018**	0.136±0.017	0.138±0.014	**0.150±0.051**
	GeoScattering	**0.155±0.034**	0.150±0.036	0.135±0.035	**0.130±0.036**	**0.131±0.037**	0.167±0.150
	NetLSD	0.156±0.037	0.132±0.035	0.139±0.038	0.132±0.033	0.138±0.030	0.187±0.082
MAPE(%)	Handcrafted	45.357±7.568	36.986±6.142	38.612±8.547	39.145±9.573	39.147±10.472	45.201±11.354
	Graph2Vec	43.739±9.957	45.171±8.153	37.738±12.346	32.443±8.248	36.539±10.724	48.744±8.124
	FeatherGraph	42.181±8.236	**32.456±7.341**	**29.388±6.932**	**31.172±8.970**	31.088±9.991	**44.763±9.048**
	GeoScattering	**41.609±7.251**	35.031±8.881	34.021±5.841	35.673±7.724	**28.992±8.896**	49.035±8.074
	NetLSD	43.379±10.236	36.031±9.887	35.378±9.132	34.079±9.316	34.016±9.165	46.775±10.561

and Geoscattering, generated the lowest mean MAE, RMSE, and MAPE.

3.5 Summary

In this article, we present a generic predictive modeling procedure for saliva cortisol levels based on passively detected actigraphy data. The suggested method allows researchers to optimize the window size for feature space and sensor selection in order to determine the window size and sensor combination that produces the lowest testing error. By creating unsupervised learnt graph features to train machine learning models with the lowest MAE, we show that GRL outperforms handcrafted feature engineering.

The suggested procedure has the potential to enhance pancreatic cancer therapy, which currently has one of the lowest 5-year survival rates of any cancer location. Specifically, given that cortisol has been shown to promote tumor development and impair response to cancer therapy, developing a method for continually monitoring cortisol levels may aid in clinical decision-making, hence improving cancer outcomes. For instance, knowing that a patient's cortisol level has been raised for an extended period of time may prompt clinicians to prescribe behavioral therapies known to alleviate the stress response.

Mobile sensing and machine learning have the ability to solve a need in cancer by offering a method for in situ prediction of saliva cortisol levels. According to the procedure described in this research, extracting features from accelerometer data with a 9.5-hour window size offered adequate feature space for fitting regression models. The findings of Graph Representation Learning reveal that when accelerometer data is used to predict cortisol levels, the automated feature engineering techniques FeatherGraph and Geoscattering produced the lowest error. Additionally, the random forest feature importance analysis revealed that features capturing motion signal amplitude contributed the most to the prediction of saliva cortisol levels, indicating that physical motion may be a major predictor of salivary cortisol levels.

The suggested methodology makes a significant contribution by providing researchers and doctors with a method for possibly tracking cortisol levels in the body with low measuring load. By using passive data streams from wearables, it is possible to circumvent wasteful and expensive saliva collection and testing. Providing an easy and accessible method of simulating cortisol is critical for cancer patients, who face much more burden and discomfort than the general population.

Although this study used data from ten recently diagnosed patients with pancreatic cancer, our findings suggest the viability of a technique that employs passively sensed data from wearables to estimate salivary cortisol levels throughout the day. Our results imply that autonomous feature engineering and machine learning techniques may be used to predict cortisol levels from temporally dense actigraphy data, however more research should be conducted on a larger sample size to confirm the robustness of these findings.

Along with the above, the present work should be assessed in light of some constraints. Due to the tiny size of the learning samples, generalizability may be limited. This may be reduced in the future by increasing the number of participants or by investigating the use of other machine learning algorithms capable of training more robust models on tiny datasets. The semantic information included in each vertex of the graphs is ignored by graph representation learning algorithms. This implies that graph representation learning cannot capture the ideas represented by vertices. In addition, hyperparameter optimization (such as $min\ observation$, $max\ depth$ in G-Means) and feature selection were not performed given their needs for a powerful computing source.

In a research of real-time predictive modeling of saliva cortisol levels, multimodal sensors (heart rate, galvanic skin reaction) may be used to extract additional relevant information and therefore improve prediction accuracy. Additionally, utilizing transfer learning, tailored machine learning models may be utilized to develop customised cortisol level machine learning models to address individual features.

Finally, our findings indicate that the degree of inaccuracy associated with forecasting cortisol levels using passively acquired data fluctuates during the day. Future research should evaluate the therapeutic consequences of bigger vs smaller prediction mistakes, which will very certainly vary depending on how the information is utilized in a clinical environment. For example, if the aim is to understand the overall trajectory of a patient's cortisol levels during therapy, forecasts with a higher (vs. a lower) measurement error are still necessary. By contrast, such forecasts are of little benefit when the goal is to act when cortisol levels reach a given number or time point with a small margin of error.

Chapter 4

Multi-modal Graph Neural Network for Influenza-like Symptom Recognition

4.1 Background and Motivation

From the Spanish flu to SARS and swine flu, worldwide pandemics produced by influenza viruses have wreaked havoc on human civilization, claiming thousands of lives and wreaking havoc on businesses [74]. These pandemics placed existing public health and socioeconomic institutions under the scrutiny, resulting in several policy adjustments aimed at mitigating future pandemics [75].

However, the continuing COVID-19 pandemic shows previously unknown weaknesses in the virus's present containment mechanisms, owing to the virus's delayed onset of symptoms and ease of transmission, and provides far larger hurdles to governments' public health response [76]. In the United States, the Centers for Disease Control and Prevention (CDC) launched the United States Influenza Surveillance System to collect and analyze influenza-related data and to monitor influenza activity [1]. The CDC collects and reports hierarchically aggregated outpatient data on ”influenza-like illness” (ILI) from local hospitals

[1]CDC-Flu Activity and Surveillance https://www.cdc.gov/flu/weekly/index.htm.

and public health systems [77]. Due to the fast spread of influenza viruses, various public health initiatives may be delayed or rendered ineffective. Thus, it is vital to build intelligent influenza surveillance systems capable of monitoring influenza activity constantly, automatically detecting early ILI in the general population, and correctly forecasting influenza epidemics [78].

Numerous studies have been conducted on influenza activity monitoring and detection of influenza-like symptoms. Forsad et al. evaluated a contactless syndromic surveillance platform comprised of a microphone array and a thermal camera installed in a university hospital's public waiting area for the purpose of monitoring influenza infection scenes continuously and passively by characterizing the captured influenza bio-clinical signals [79]. The Google search and Wikipedia pageview patterns for influenza-related phrases, as well as ILI-related linguistic signals gathered from social media platforms, are utilized to develop natural language processing models for forecasting influenza dynamics[80]. Numerous data sources, including air quality and insurance records, have been used to forecast the likelihood of influenza epidemics [81]. While the approaches described above are capable of detecting ILI and forecasting influenza trends in a promising manner, their estimates have some limitations. For example, Google Flu Trend has been stopped owing to its errors and lack of repeatability [82]. Additionally, putting edge sensors in hospitals to detect influenza might result in erroneous estimates that cannot be applied to a broader population.n [79].

Mobile sensing offers a solution for ILI detection and monitoring by gathering both behavioral and physiological data produced by human users constantly and unobtrusively. [83]. The accelerometer, GPS sensor, and Bluetooth sensor included in mobile devices (e.g., personal cellphones and smartwatches) have been used to monitor human behavior and track everyday activities. The data collected by these embedded sensors may be utilized to determine a person's health condition, monitor mental health states, and administer medical therapies. [1, 2, 3]. Historically, handmade features have been taken from mobile sensing data to study patterns of human activity. For instance, motion characteristics such as acceleration magnitude can be collected from an accelerometer in order to investigate their relationships with various user scenarios (e.g., location, activity, social context) [4]. Rather of manually creating features based on heuristics and domain expertise, high-level complex features may be retrieved automatically from mobile sensing data and used to increase the generalization of predictive modeling using deep learning algorithms. For example, Boukhechba et al. built convolutional neural network (CNN) models based on photoplethysmogram (PPG) data to infer ambulatory activities [5]. However, with multi-modal mobile sensing data, classic handmade feature engineering and generic deep learning approaches (e.g., CNNs) are constrained. On the one hand, handmade feature engineering might exclude information regarding the interdependence of many sense modalities. On the other hand, complex relationships between data points inside and across sensing streams cannot be naturally stored in deep features by CNN, much more so when the data contains hidden structural patterns and is derived from non-Euclidean domains [6]. Given that some discrete sensor signals, such as GPS and Bluetooth streams, can be naturally represented as graphs, which encode the structural interdependence and information generated by non-euclidean spaces, we proposed an end-to-end graph neural network (GNN) framework for inferring the presence of influenza-like symptoms using people's daily multimodal mobile sensing data.

Our contributions in this work are summarised below:

- By using multi-modal mobile sensing data, we present an end-to-end Graph neural network architecture for modeling human activities. The objective is to extract high-level characteristics characterizing dynamic interactions between human states in order to forecast ILI symptoms automatically. To our knowledge, this is the first time that Graph neural networks have been used to infer human health conditions using mobile sensing. This end-to-end GNN architecture is readily generalizable and applicable to additional mobile sensing applications, such as mental health symptom monitoring [84, 85, 86, 87].

- We illustrate the proposed GNN framework's performance by applying it to a large mobile sensing dataset gathered in the field for the purpose of recognizing influenza-like symptoms. The findings indicate that GNN models beat baseline techniques based on handmade features by a substantial margin.
- Rather of manually characterizing nodes in input graphs, we use a graph representation learning approach called node embedding to build topological embeddings of nodes automatically. Our findings suggest that approaches for graph representation learning may be used to build embeddings for each unique human state.
- To promote model interpretability and confidence in 'black box' GNNs, we use interpretable GNN approaches. We demonstrate the transparency of GNN models in order to help policymakers and medical professionals better understand this framework.

4.2 Method

In this part, we describe our technique for comparing handmade feature engineering and graph representation methods for modeling human behaviors and determining whether or not individuals have at least one influenza-like symptom based on daily mobile sensing data. Fever, feeling feverish/chills, cough, sore throat, runny or stuffy nose, muscular or body pains, headaches, and weariness are all considered influenza-like symptoms [2]. The presence of influenza-like symptoms may impair everyday mobility, social contacts, and physical activity of individuals. When individuals experience weariness or muscular discomfort, for example, they may become more sedentary and avoid needless travel, even if they are unaware of these early stage flu symptoms. Significant decreases in daily social interaction time and average contact length were seen in persons with influenza-like illness in [88]. We hypothesize that people's mobility behaviors, social contacts, and physical activities are associated with and can be utilized to correctly identify influenza-like symp-

[2]CDC-Flu symptoms https://www.cdc.gov/flu/symptoms/symptoms.html.

toms. We employ GPS sensors to monitor people's daily movement patterns to capture human mobility behaviors; Bluetooth sensors to approximate social contacts; and activity data generated from accelerometer and gyroscope data to represent physical activities. For mobility data, we applied DBSCAN [89] to cluster GPS coordinates into visited places, and form sequences of daily place visits. Activities (i.e $in - vehicle, running, walking$) were recognized by using Google Activity Recognition API [90] for Andriod devices and CMMotionActivity API for iOS devices [91]. We denoted daily place visit trace as $\mathbf{X_p}$, daily Bluetooth encounter traces as $\mathbf{X_b}$, and daily activity trace as $\mathbf{X_a}$. Our prediction task can be generalized as learning a function $\mathcal{F}(\bullet|\boldsymbol{\theta})$ given the inputs of $\mathbf{X_p}, \mathbf{X_b}, \mathbf{X_a}$ to predict existence of influenza-like symptoms denoted as $y \in \{0, 1\}$, such that $\mathcal{H}(y, \hat{y})$ can be minimized, where $\mathcal{H}(\bullet)$ is the cross entropy loss function. We propose to compare two feature engineering methods: 1) using handcrafted feature engineering as shown in section subsubsection 4.2; and 2) using Graph representation to automatically extract high level features in section Table 4.2.

Handcrafted Feature Engineering

To quantify human movement and develop a prediction model for the detection of influenza-like symptoms, we propose to calculate fine-grained handmade characteristics from GPS coordinates (e.g., number of visited places), which have been studied in [3]. The details about mobility features and description are shown in Table Table 4.1. Similarly, social interaction (e.g., number of encountered Bluetooth devices) and physical activity features (e.g., entropy of physical activity) are also extracted and shown in Table Table 4.1.
Graph-based Feature Engineering While handmade feature engineering may result in finer-grained and more interpretable features, they are often not generalizable across populations since they are developed utilizing domain expertise and intuition [92]. Additionally, handmade traits may not accurately reflect the dynamic properties of human movement, social relationships, and physical activity. For example, given the varied number of sites visited

Behavior	Feature	Description
Mobility	radius_gyration	radius of gyration
	number_visits	total number of places visited
	max_distance	maximum distance of place visited from centroid
	travel_distance	total distance traveled
	max_distance_home	maximum distance of place visited from home
	travel_distance_std	standard deviation of distance of trips
	entropy_visits	randomness of places visited
	unique_visits	number of unique place
	multiple_place	number of placed visited at least twice
Social Interaction	num_encounter	number of Bluetooth devices detected
	entropy_encounter	randomness of Bluetooth devices
	unique_encounter	number of unique Bluetooth devices detected
Physical Activity	act_entropy	randomness of physical activity

Table 4.1: Description of Handcrafted Features

by each individual, the effect of moving from one place to the next cannot be realistically represented through handmade feature engineering. To address this limitation, we represent daily GPS trajectory, Bluetooth encounters, and Physical activities as graphs, as shown in Fig Figure 4.1 and apply GNNs to automatically generate deep features for our influenza-like symptoms recognition task.

We use $\mathcal{G}_p = (V_p, E_p, \mathbf{X_p})$ to represent daily GPS trajectory, where $V_p = \{v_{p_1}, v_{p_2}...\}$ and $E_p = \{e_{p_1}, e_{p_2}...|e_{p_i} \in V_p \times V_p\}$ are the set of nodes (places visited) and edges (transitions between places), respectively; and $\mathbf{X_p} = \{x_{p_1}, x_{p_2}, ... \,|x_{p_i} \in \mathbb{R}^d\}$ are the feature vectors associated with each node in $\mathcal{G}_p$, where d is the node features dimension. We represent each detected Bluetooth device as a node in the graph and use edges to denote the adjacency of two identified Bluetooth devices in the Bluetooth encounter network. We use $\mathcal{G}_b = (V_b, E_b, \mathbf{X_b})$ to denote daily Bluetooth encounters , where $V_b = \{v_{b_1}, v_{b_2}...\}$ and $E_b = \{e_{b_1}, e_{b_2}...|e_{b_i} \in V_b \times V_b\}$ are the set of nodes (scanned devices) and edges (adjacency), respectively; and $\mathbf{X_b} = \{x_{b_1}, x_{b_2}, ...|x_{b_i} \in \mathbb{R}^d\}$ are the feature vectors associated

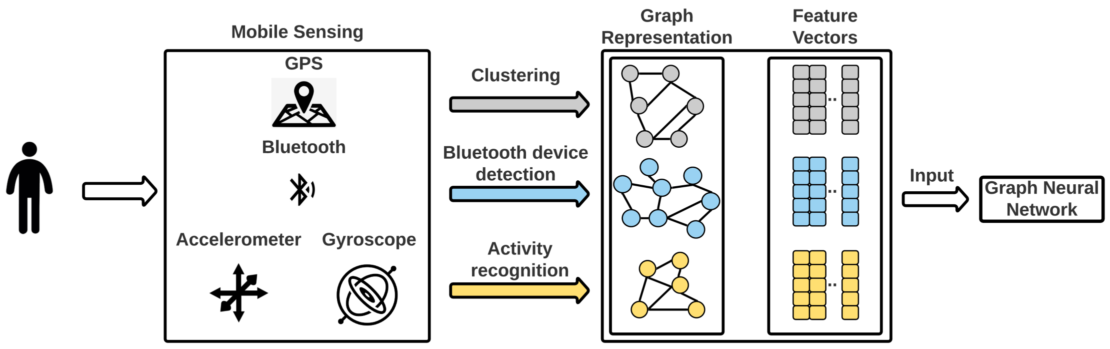

Figure 4.1: Graph Based Behavior Modeling using Mobile Sensing.

with each node in $\mathcal{G}_b$, where d is the node features dimension. To represent physical activity transitions, we encode each activity label as node in the transition graph denoted as $\mathcal{G}_a = (V_a, E_a, \mathbf{X_a})$, where $V_a = \{v_{a_1}, v_{a_2}...\}$ and $E_a = \{e_{a_1}, e_{a_2}...|e_{a_i} \in V_a \times V_a\}$ are the set of nodes (physical activities) and edges (transitions between physical activities), respectively; and $\mathbf{X_a} = \{x_{a_1}, x_{a_2}, ...|x_{a_i} \in \mathbb{R}^d\}$ are the feature vectors associated with each node in $\mathcal{G}_a$, where d is the node features dimension. All the graphs $\mathcal{G}_p, \mathcal{G}_b$, and $\mathcal{G}_a$ are unweighted and non-directional. As shown in Fig Figure 4.1, behavior represented graphs and their corresponding feature vectors can be fed into graph neural networks, discussed in section Figure 4.2, to build predictive models.

Graph Neural Networks and Node Embedding We use graph neural networks (GNNs) to efficiently extract high-level topological properties from non-euclidean areas given daily movement patterns, social contacts, and physical activities. GNNs, in particular, can capture high-order information about neighbor interactions in graphs and aggregate local node attributes to build graph-level numerical representations [6]. In this paper, we demonstrate how to formulate this prediction problem as graph classification given a multi-channel input of graph structured data, $\mathcal{G}_p, \mathcal{G}_b$ and $\mathcal{G}_a$. As shown in Fig Figure 4.2, to predict the presence of influenza-like symptoms, we propose an end-to-end Graph neural network architecture with multi-channel input. Additionally, the feature vectors of each node in a

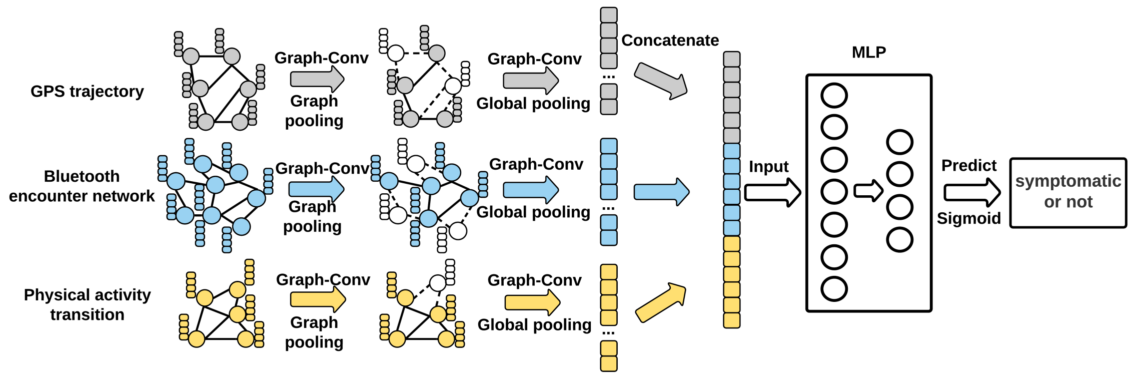

Figure 4.2: Multi-Channel Graph Neural Network for Influenza-like Symptom Recognition.

network may be automatically trained using Node Embedding, which maps each node into numerical space in such a way that nodes with comparable topological structures are closed in the embedding numerical space. The details of node embedding selection will be demonstrated in section subsubsection 4.4. We compared and contrasted the performance of four distinct state-of-the-art node embedding approaches (Node2Vec, Walklets, NodeSketch, and BoostNE) and four different graph convolutional layers (GCN, GaphSAGE, GAT, and GIN), as stated in the following subsections.

Graph Convolutional Layers

Graph Convolutional Networks (GCNs) [93] are neural networks that perform graph convolution operations on graph structured inputs. GCNs can infer node level embeddings based on the features of node neighborhoods. Given a graph $\mathcal{G}(V, E, \mathbf{X})$, we denote the adjacency matrix and degree matrix of $\mathcal{G}$ as A and D respectively. Then layer-wise propagation in a GCN can be expressed as

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}). \tag{4.1}$$

We set each node in $\mathcal{G}$ as self-connected, and define $\tilde{A} = A + I$, where I is the identity matrix. $\tilde{D} = D + I$, and $W^{(l)}$ is the weight matrix which will be learned in the training process. l indicates the lth layer of graph convolution operations. $H^{(l)}$ is the matrix output with activation from lth graph convolution layer, and $H^{(0)} = \mathbf{X}$. $\sigma(\bullet)$ is the activation function, such as $ReLU(x) = max(0, x)$. GCNs combine node characteristics from the nodes' first-order neighborhood using a single layer of graph convolution. Additional convolution layers may be used to accomplish higher level neighborhood aggregation..

GraphSAGEs [94] makes use of a variety of aggregation functions to include information about node neighborhoods into the search depth. Unlike GCNs, which aggregate first-order neighborhood information about nodes, GraphSAGEs use forward propagation to concatenate the characteristics of each local neighborhood along the propagation trace, allowing for the application of higher-order topological properties to aggregated node attributes. The GraphSAGE convolution can be expressed as

$$h_v^{(l+1)} = \sigma(W^{(l)} \bullet CONCAT[h_v^{(l)}, F^{(l)}(\{h_u^{(l)}, \forall u \in \mathcal{N}(v)\}]) \tag{4.2}$$

for each node v in V, where $CONCAT$ is the concatenation operation and $F^{(l)}(\bullet)$ is the aggregation function in layer l, which can be mean, max, sum, and LSTM [94]. $\mathcal{N}(\bullet)$ represents the set of neighborhoods of the node v.

Graph Attention Networks (GATs) [95] make use of attention processes in order to generate node representations. When dealing with sequential observations such as text data, attention is often achieved by assigning a higher weight to the most significant aspects in the inputs, so that greater attention is focused on the key elements during prediction. GATs change attention mechanisms by learning the weights associated with node neighborhoods; during the aggregation process, more weights are applied to node neighborhoods that have a greater effect on the node. The graph attention convolution can be defined as

$$z_v = \sigma(\sum_{u \in \mathcal{N}_{(v)}} \alpha_{v,u} W X_t), \tag{4.3}$$

$$h_v^{(l+1)} = \sigma(\sum_{u \in \mathcal{N}_{(v)}} \alpha_{v,u}^{(l)} W^{(l)} h_u^{(l)}), \tag{4.4}$$

where $h_v^{(0)} = x_v$, and $\alpha_{v,u}^{(l)}$ is the attention of node v in the lth layer, which can be further expressed as

$$\alpha_{v,u} = \frac{exp(\eta(\mathbf{a}^T[Wx_v, Wx_u)]}{\sum_{u' \in \mathcal{N}(v)} exp(\eta(\mathbf{a}^T[Wx_v, Wx_{u'}])}. \tag{4.5}$$

$$\alpha_{v,u}^{(l)} = \frac{exp(\eta(\mathbf{a}^{(l)T}[W^{(l)}h_v^{(l)}, W^{(l)}h_u^{(l)}])}{\sum_{u' \in \mathcal{N}(v)} exp(\eta(\mathbf{a}^{(l)T}[W^{(l)}h_v^{(l)}, W^{(l)}h_{u'}^{(l)}])}. \tag{4.6}$$

Here η is LeakyReLU activation function, while $\mathbf{a}$ is a weight vector to parametrize the attention mechanism.

Graph Isomorphic Networks (GINs) [96] generalize Weisfeiler-Lehman graph isomorphism test to better discriminate graphs. GINs apply multi-layer perceptrons (MLP) to approximate the composition function as shown below:

$$h_v^{(l+1)} = MLP^{(l)}((1 + \epsilon^{(l)}) \bullet h_v^{(l)} + \sum_{u \in \mathcal{N}(v)} h_u^{(l)}), \tag{4.7}$$

where $\epsilon^{(l)}$ is a learnable parameter. GINs are argued as powerful as the Weisfeiler-Lehman graph isomorphism test for graph classification tasks [96].

Node Embedding

Node embeddings are used to infer attribute attributes for nodes in graphs by transferring graph structured data to numerical spaces. By utilizing the relationships between nodes and graph structures, node embeddings may automatically extract high-level node properties.

[10]. In general, there are two categories of Node embedding methods: 1) Deep walk [97]; 2) matrix factorization [98]. We applied two node embedding methods from each category listed below:

1. **DeepWalk**

   (a) *Node2Vec* [11] employs an exploration-exploitation technique that takes into account both homophily and structural equivalence in graphs. Node2Vec can produce continuous feature representations to encode the immediate local structure of nodes and global network neighborhoods by doing depth first and breadth first searches on the node neighborhoods.

   (b) *Walklets* [99] samples node neighborhoods context with skipping over nodes in each random walk. Multi-scale relationships can be encoded by subsampling fixed length of path in the node neighborhoods context generated from skipping random walks.

2. **Matrix factorization**

   (a) *NodeSketch* [100] starts lower-order node embedding with generating Self-Loop-Augmented (SLA) adjacency matrix and then using hashing functions to map each SLA adjacency vector into Hamming space to approximate the similarity of each adjacency vector.

   (b) *BoostNE* [101] conducts a sequence of non-negative matrix factorization to the residual of the connectivity matrix approximated from previous step to generate sequence of weak embeddings. Then the sequence of weak embeddings will be ensembled to generate fine-grained embedding representations.

4.3 Data and Experiments

The data for this article is a subset of a multicohort research that examined the use of mobile sensing methods for early sickness diagnosis. A total of 2700 individuals from 24 states in the United States were recruited to participate in this research for a period of up to one year. We excluded individuals who did not have at least 14 days of GPS, Bluetooth, activity, or self-reported influenza-like symptom data. Finally, we will preserve data from 448 participants for our current study, which will run from February 15th 2019 through April 30th 2020. The mean age of the participants is $40.71(sd = 11.52)$. 65.36% of the participants are female, with White being 66.2%, African American 19.3%, Asian 7.3%, multiple races 3.3%, Hispanic 3.5%, and others 0.4%. Among the 448 participants, 52% are full time workers, 20% are full time students, 12% are working part time, 7% have retired, 5% are care takers, and 4% are temporarily unemployed. More details about the larger study can be provided by the corresponding author.

Participants were invited to download and use ReadiSens, a mobile sensing application, for a period of up to four months as part of the data gathering procedure. ReadiSens is a cross-platform software developed on top of Sensus [102] that collects GPS location data every 30 minutes; Bluetooth Encounters data every 15 minutes in Android and every 30 minutes in iOS; and activity data whenever activities change in Android and every two hours in iOS. Every day at 8 p.m., ecological momentary assessments (EMAs) were administered to gather self-reported symptoms of fever, cough, trouble breathing, exhaustion, muscular pains, headache, sore throat, runny nose, nausea, and diarrhea. Periodically, data were transferred to Amazon Web Services's Simple Storage Service (AWS S3) (S3). To ensure participants' data security and privacy, all data were encrypted and anonymized. Before the data was saved on AWS, all identifying information was removed. By eliminating the integer components of their longitudes and latitudes, GPS data were anonymised. Readisense hashed the Bluetooth name and MAC addresses automatically.

Symptom	Node Number			Average Node Degree			Connectivity			Assortativity		
	GPS	Bluetooth	Activity	GPS	Bluetooth	Activity	GPS	Bluetooth	Activity	GPS	Bluetooth	Activity
NO	6.32	180.70	5.02	3.25	2.79	4.96	1.08	1.03	2.87	-0.33	-0.06	-0.10
YES	6.00	157.45	4.86	2.91	2.74	4.82	1.07	1.02	2.86	-0.39	-0.14	-0.11

Table 4.2: Compare the average graph metrics of GPS trajectory, Bluetooth encounter networks, and Physical activity transitions between days of participants with influenza-like symptoms and those without any symptoms.

In this section, we illustrate the detailed information about data preprocessing, graph modeling, GNN implementation, model selection, interpretability of GNN and hyperparameter sensitivity analysis.

Data Preprocessing and Statistics

After data preprocessing, we get 8,415 observations, of which 6,923 are non-symptomatic (no influenza-like symptoms recorded) and 1,492 (17.7% of all samples) are symptomatic (at least one influenza-like symptom reported). We transform daily GPS trajectories, Bluetooth interactions, and activity transition traces to graph representations. Figure Figure 4.3 provided one sample of the constructed graphs for each sensing modality.

To highlight and compare the graph features of robust and symptomatic samples, we give in Table Table 4.2 summary statistics for graph metrics. The average node number, node degree, connectedness, and assortativity are consistently lower in symptomatic instances than in non-symptomatic cases. These inconsistencies suggest that symptomatic patients have fewer vertices and edges in their networks, meaning that people with influenza-like symptoms may have less mobility, social engagement, and various physical activities. Due to the uneven proportion of symptomatic and non-symptomatic patients in our tests, we oversampled the mild instances. To conduct a fair comparison of several GNNs [103], we divided the whole dataset into 90% training data and 10% testing data, and within the 90% training data, we selected 10% training data as validation data and rebalanced the remaining training data using an oversampling approach. After oversampling, we ended up

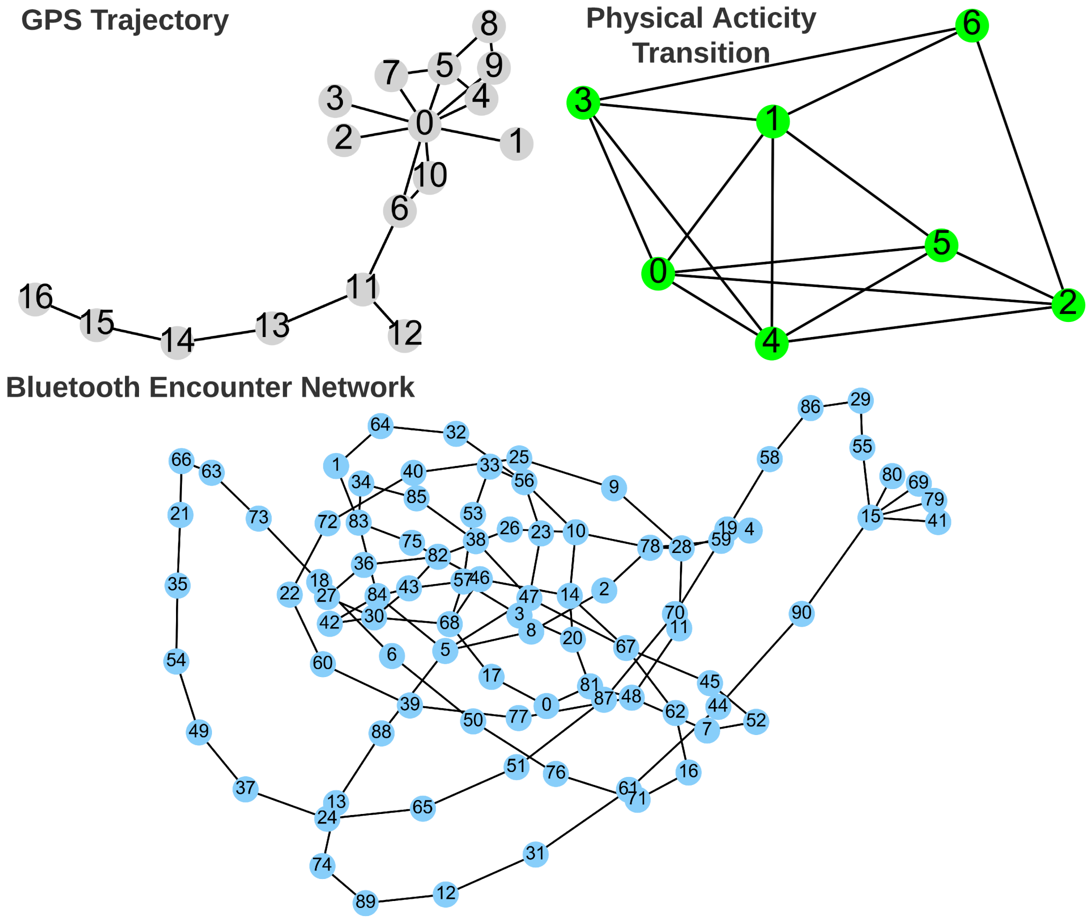

Figure 4.3: Samples of daily location trajectory, Bluetooth encounter network, and physical activity transitions graphs.

with 10,239 training samples, 692 validation samples, and 841 testing samples. To cope with missing networks in the data set, we imputed naïve graphs consisting of two nodes connected by a single edge.

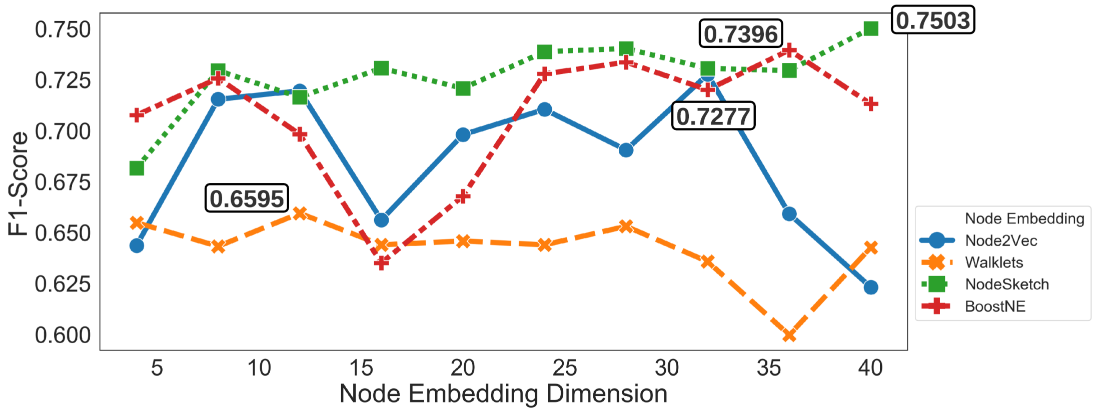

Figure 4.4: F1-score comparison for the combinations of Node Embedding methods and embedding dimensions. GNNs with two GCN layers were built with TopK (ratio = 0.75) graph pooling layer and global sum pooling layer. The training processes were set using 40 epochs, 64 batch size, 10 step patience for early stopping, and learning rate 0.001 for Adam optimizer.

4.4 Results and Analysis

Node Embedding Selection

To determine the optimal node embedding approach with suitable embedding dimensions, we ran one run of each of the aforementioned node embedding techniques using GCNs with increasing embedding dimensions. For 20% random sampled data, we employed two layers of GCNs with TopK graph pooling and a global sum pooling layer. The F1-scores obtained throughout these tests are shown in Figure Figure 4.4. Matrix approximation techniques (NodeSketch, BoostNE) outperformed DeepWalk techniques (Node2Vec, Walklets). This might be because DeepWalk algorithms often need a large number of sampled node neighborhoods to construct the embedded nodes' contexts. However, since the average degree of nodes in our data is modest, sampling random walks may be inadequate to acquire appropriate latent representations. Based on its performance, we choose NodeSketch with a 40-embedding-dimension.

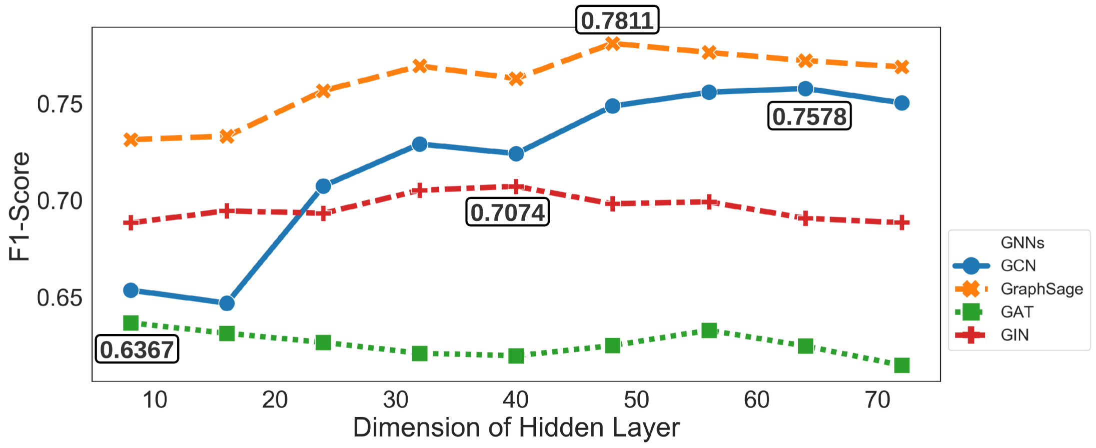

Figure 4.5: F1-score comparison for the dimensions of neurons for the hidden layers in GNNs. GNNs consisted of two Graph convolutional layers with TopK (ratio = 0.75) graph pooling layer and global sum pooling layer using the entire dataset. The training processes were set using 80 epochs, 256 batch size, 15 step patience for early stopping and learning rate = 0.001 for Adam optimizer.

Hidden Layer Dimension Selection

The dimension of neurons in the hidden layers may have an effect on the prediction performance of deep learning algorithms. High-dimensional neural networks often offer more prediction potential, but need bigger training data. We modified the size of graph convolutional layers in this part to determine the number of dimensions for each GNN. GCN, GraphSAGE, GAT, and GIN were all configured with the same TopK graph pooling layer and global sum pooling layer, and the GNNs were trained with the whole dataset in a single run. To extract node embeddings, NodeSkech was used. F1-scores were reported Figure Figure 4.5. We observed that the F1-scores generally increased with increasing dimension of hidden layers. However GAT and GIN were much less sensitive to the variation of hidden layer dimensions. We selected dimensions 64 for GCN, 48 for GraphSAGE, 8 for GAT, and 40 for GIN.

Pooling layer selection In this section, we fine-tuned the GNNs with different combinations of graph pooling layers and global pooling layers, as shown in Figure Figure 4.2. Graph

Graph Convolutional Layer	**Graph and Global Pooling Layer**					
	TopK Pooling			SAG Pooling		
	GlobalSum	GlobalMax	GlobalAttention	GlobalSum	GlobalMax	GlobalAttention
GCN	0.7578	**0.8617**	0.7551	0.7881	0.8591	0.8081
GraphSAGE	0.7811	**0.8856**	0.7905	0.8068	0.8792	0.7524
GAT	0.6367	0.6309	0.6072	0.5882	**0.6409**	0.6206
GIN	0.7072	0.7313	0.6925	0.7243	**0.7441**	0.7244

Table 4.3: F1-score comparison to select best combination of graph pooling layer and global pooling layer in Figure Figure 4.2 in GNNs. The dimensions of each GNN were set up based on the results from section subsubsection 4.4. TopK pooling ratio = 0.75. The training processes were set with 80 epochs, 256 batch size, 15 step patience for early stopping and learning rate = 0.001 for Adam.

pooling layer, including TopK pooling [104] and SAG Pooling [105] in this study, is used to choose the most significant nodes (therefore constructing smaller networks) based on prediction performance. TopK pooling is a technique for selecting the top K nodes in graphs by ranking the projected scalar values of each node's characteristics. SAG pooling employs an attention mechanism to prioritize nodes according on their self-attention ratings. Global pooling layer (read-out layer), including GlobalSum, GlobalMax, and GlobalAttention [31] in this study, is to derive graph-level characteristics from node-level ones. GlobalSum and GlobalMax generate graph level representations by calculating the sum and maximum of each node feature in the feature vectors, respectively, whilst GlobalAttention generates self-attentioned graph level representations using neural networks. [31]. The results compared by F1-score were shown in Table Table 4.3. GlobalMax pooling achieved the greatest results in both TopK and SAG pooling. This might be because GlobalMax is capable of extracting the most important characteristics from a node feature vector in order to more accurately describe graphs.

To compare handcrafted and graph based behavior modeling, we select the following baseline models for handcrafted features: 1) Logistic regression [106], 2) Support vector machine (SVM) [107], 3) Random forest [71], 4) Xgboost [72], 5) Multi Layer Perceptron (MLP) [108], all of which have been widely applied and investigated in mobile sensing studies[109, 110, 111]. Alternatively, we replaced missing values in handmade features with 0 and rebalanced the training data set using oversampling. We configured the GNNs using the optimal node embedding approach, their own hidden layer dimensions and layer pooling combinations as discussed in section subsubsection 4.4, subsubsection 4.4 and subsubsection 4.4. The following hyperparameters are set: TopK pooling ratio = 0.75, epochs = 80, batch size = 256, early stopping tolerance = 15, and learning rate = 0.001. To evaluate the performance of handmade and graph-based approaches, we ran a tenfold cross validation. Precision, recall, F1-score, and AUC-score are used as comparative measures.

The results are shown in Table Table 4.4. We observe that except GAT, all other GNNs outperformed the baseline models with handcrafted feature engineering by all evaluation metrics. Except for GAT, we see that all other GNNs beat handmade feature engineering-based baseline models on all assessment measures. The training and validation loss plots (Figure Figure 4.6 created from one cross validation cycle indicate that the GCN training loss continued to drop, while the validation loss peaked between epoch 50 and 60, signaling that GCN may have been overfitted in the training data set after epoch 60.

After 80 epochs, the training loss reduced steadily, but the validation loss remained same. This might be because GIN had a much larger number of parameters to tweak, since GIN used MLPs as the composition function. Due to the fact that neither the training nor validation loss improved for GAT, early termination was enabled at 68 training epochs. GraphSAGE outperformed the other GNNs in virtually all training epochs, since it generated less training and validation loss than the other GNNs. GraphSAGE demonstrated the highest predictive value because it can combine higher level node neighborhood information, allowing for the classification of more complicated interactions between nodes. Additionally, since the input graphs often contain a high number of nodes but a limited number of node degrees, GraphSAGE can better capture the topological structures of node neighbors, while GCN could aggregate only local node structures.

Category	**Model**	**Metrics**			
		Precision	Recall	F1	AUC
Baseline	Logistic Regression	0.5673 ± 0.0096	0.6139± 0.0171	0.5896 ± 0.0129	0.6532 ± 0.0208
	SVM	0.5823 ± 0.0102	0.6414± 0.0177	0.6104 ± 0.0135	0.6838 ± 0.0133
	Random Forest	0.6617 ± 0.0308	0.6252±0.0222	0.6429 ± 0.0257	0.7339 ± 0.0151
	Xgboost	0.6047 ± 0.0111	0.6610± 0.0175	0.6316 ± 0.0137	0.7354 ± 0.0156
	MLP	0.6407 ± 0.0212	0.6679± 0.0198	0.6501 ± 0.0186	0.7443 ± 0.0162
GNN	GCN	0.8079 ± 0.0246	0.8549 ± 0.02446	0.8307 ± 0.0236	0.9148 ± 0.0195
	GraphSAGE	**0.8546 ± 0.0210**	**0.8991 ± 0.0164**	**0.8762 ± 0.0184**	**0.9539 ± 0.0132**
	GAT	0.6087 ± 0.0065	0.6872 ± 0.0133	0.6455 ± 0.0058	0.6725 ± 0.0091
	GIN	0.7176 ± 0.0153	0.7733 ± 0.0223	0.7443 ± 0.0161	0.8179 ± 0.0192

Table 4.4: Results comparison between baseline models with handcrafted feature engineering and GNNs

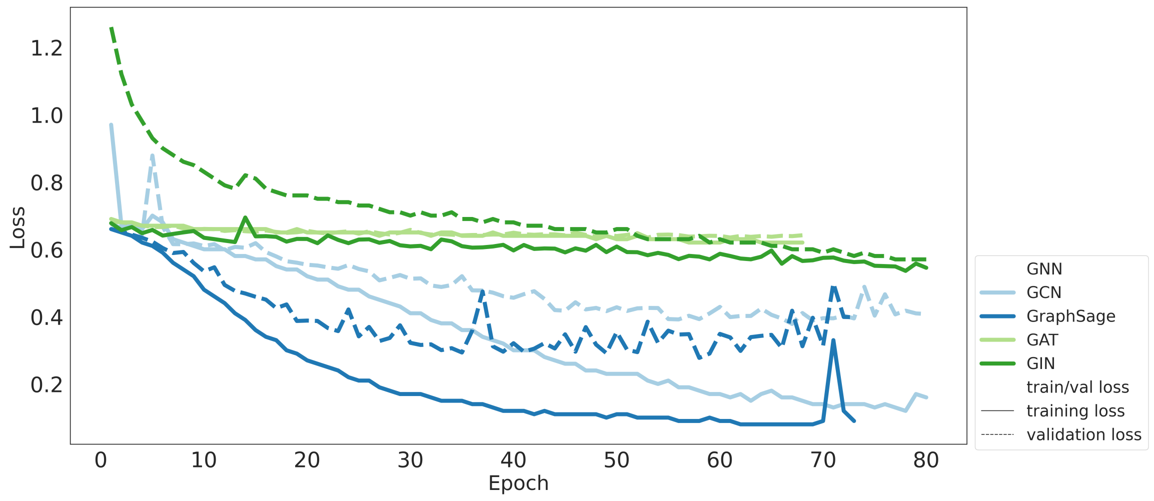

Figure 4.6: Training and validation loss comparison of GNNs over epochs.

Interpretability of Graph Neural Networks

To build trust in the black box GNN approaches, and increase the transparency of GNN's decision making mechanism, we applied Class Activation Mapping (CAM) [112] to interpret contributions of the node features and graph structures to GraphSAGE's prediction. CAMs have been frequently utilized in image recognition to demonstrate how convolutional neural networks may be used to perform classification tasks using high-level picture characteristics [113]. In this work, we use the GNN version of CAM to uncover critical node properties for classifying graphs. GNN-based CAMs may examine the output of the previous graph convolutional layer and associate the learnt feature weights with the input node attribute to determine the node relevance for graph categorization. We define GRL generated features as the latent representation output from Graph Convolutional layer without loss of generality. In this example, we use graphs to represent people's mobility behaviors, the proximity of social interactions, and physical activities. We utilize interpretable GNN techniques to demonstrate the interpretability of the automatically learned features generated from GNN.

The representations of daily mobility, Bluetooth encounters, and physical activities as graphs are shown in Fig Figure 4.1. NodeSketch [100] is applied to generate the node embedding. Graph structure data (adjacency matrix) and feature vector (node embedding) for each node are fed into Graph Neural Networks to automatically generate deep features for influenza-like symptoms recognition. All the graphs are unweighted and non-directional. Here we use **GraphSAGEs** [94] to build GNN model. We generated the heat maps of node importance in graphs of daily base observations from both symptomatic and non-symptomatic cases which were randomly selected, as shown in Figure Figure 4.7. In the non-symptomatic cases, we observed that nodes from different cases, different graphs, and different sub-graph structures can have different importance to the symptom recognition task. Important nodes were distributed globally in multiple sub-structures in each graph. This implied that multiple nodes in each non-symptomatic graph could have a high contribution to symptom recognition. In the symptomatic cases, important nodes only showed in a limited number of sub-graph structures, implying that only a small number of nodes corresponding to few human states and state transitions could have high importance in classifying symptomatic cases. The interpretation of GNNs in this influenza-like symptom recognition task is illustrated as following:

- **Mobility Behavior Interpretation:** the visiting pattern graph in the non-symptomatic case showed with more complex and higher node degrees than the visiting pattern graph in the symptomatic case, implying that a healthy person could visit many places one day, but for some people who have influenza-like symptoms, they may avoid visiting many places. In the non-symptomatic case, as shown in Figure 4.7 (a), we observed that multiple visited places (node 1, 2, 3) had high contribution to classify non-symptomatic cases. In this case, this person started from place 0, which usually the home or dorm for a student. Then, two places 1 and 2 with high node importance were visited, which could be a gym or coffee shop. Then this person went back to home/dorm and then went out to start a day. This visiting pattern could reflect a robust person's routine visiting pattern. In the symptomatic case, this person went out from home to place 1, where she could buy some food, and then went back to home. Then this person left home to visit place 2, where she could buy some necessary stuff, and then went back again. The place 3 with high node importance could be a clinic and, after the visited clinic, she went to a drugstore to pick up drugs. This repetitive go-out and back-to-home pattern could reflect that she would not go out except it was necessary. From another perspective, usually, people will go out to finish all business and then go back home. This observation implied that both global and local graph structures can impact node importance.

- **Social Interaction Interpretation:** In non-symptomatic case, multiple nodes could have high importance to infer a healthy state. We could explain that, without any influenza symptoms, people could be more likely to be in an environment with a higher probability of social interactions (with a plethora of detected Bluetooth devices). That made the important Bluetooth nodes distribute globally in the graph. In the symptomatic case, a small number of detected Bluetooth encounters with high node degree implied a lower probability of diverse social interaction because people might stay in a determined place (like home). Only one detected encounter showed high importance implied that this node could represent a Bluetooth device at home.

- **Physical Activity Interpretation:** in the non-symptomatic case, multiple activities

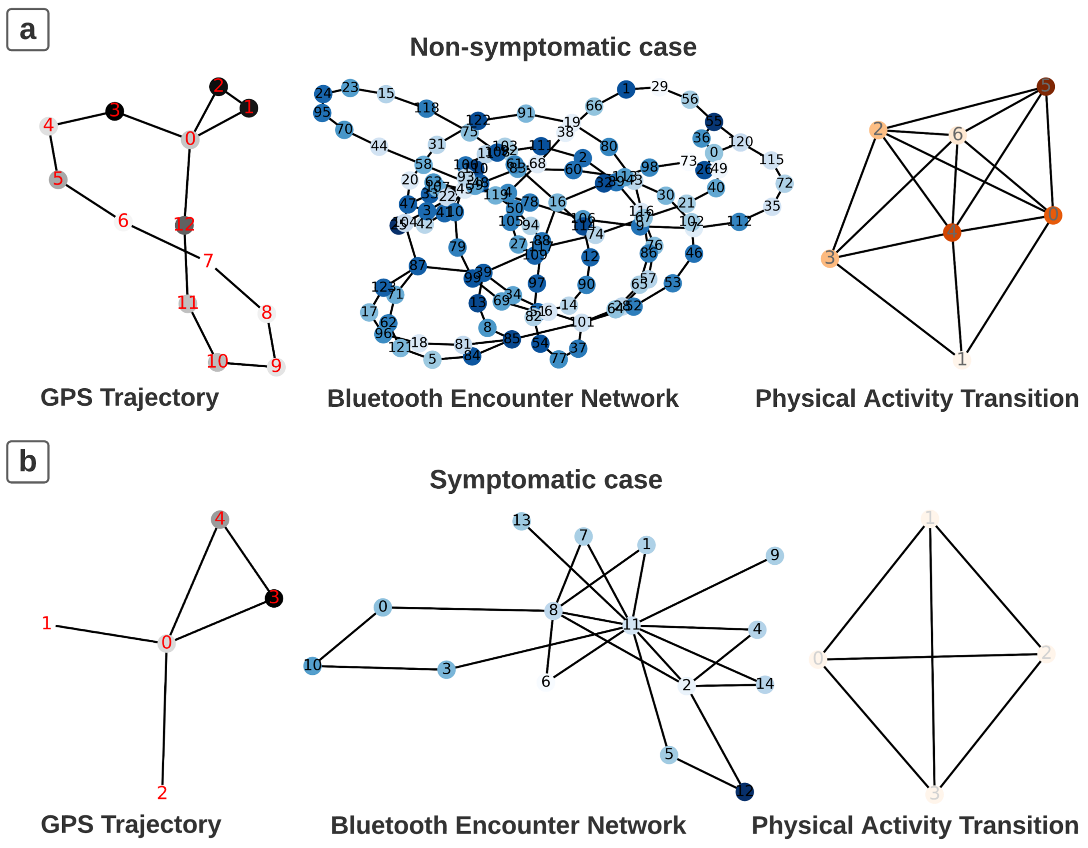
a
Non-symptomatic case
GPS Trajectory
Bluetooth Encounter Network
Physical Activity Transition
b
Symptomatic case
GPS Trajectory
Bluetooth Encounter Network
Physical Activity Transition

Figure 4.7: Interpretable GraphSAGE: (a) is a sampled Non-symptomatic case, and (b) is a sampled Symptomatic case. The darker the nodes the more important of this node will be to predict existence of influenza-like symptoms. The numbers shown in each node indicate the order of being in each node.

showed high importance for the symptom recognition task, implying that some activity states, such as running, driving, were more important to infer a non-symptomatic health state than other activities. Compared with mobility and social interactions, physical activity was less important in predicting influenza-like symptoms in symptomatic cases. This implied that some basic and sedentary physical activities, such as walking and keeping still, did not help to discriminate symptomatic behavior every well.

Comparison of Interpretability between Handcrafted Features and Features generated from GNNs

The feature importance plot is shown in Figure Figure 4.8. The comparison of interpretability between handcrafted features and features generated from GNNs is discussed below.

- **GNN can have better interpretability by providing semantic explanation of human behaviors** [15]: As we demonstrate in Section Interpretability of Graph Neural Networks, the GNN based interpretability method can generate the semantic meaning of human behaviors. In an ideal situation, if we can get access to the real GPS coordinates, location information of visited places can be precisely inferred. For example, we can infer the functionality of visited places from their semantic labels (i.e., library, hospital). With this information, node importance can be more accurately interpreted. However, handcrafted features only provide summary statistics of human behaviors, which cannot provide semantic information of the behaviors' context.
- **The Interpreted information of GNN can be more comprehensive than engineered features**: the engineered features, though easily understandable, can only capture low-level features of human behaviors and generate biased behavior representation. For example, 3 $unique - visits$ of the workplace, home, and hospital can be embedded with totally different human behaviors from 3 $unique - visits$ of home, gym, and coffee shop. In an ideal case, GNN can keep track of all human states with semantic information and the transitions between the states, which is more sophisticated than handcrafted features.
- **Interpretability of GNN can show better visual interpretation of the GNN learned features:** comparing with Figure Figure 4.8, node importance visualization can demonstrate each node's contribution to the machine learning task. The sub-structure of the graph can also be visualized and provide interpretable information about the

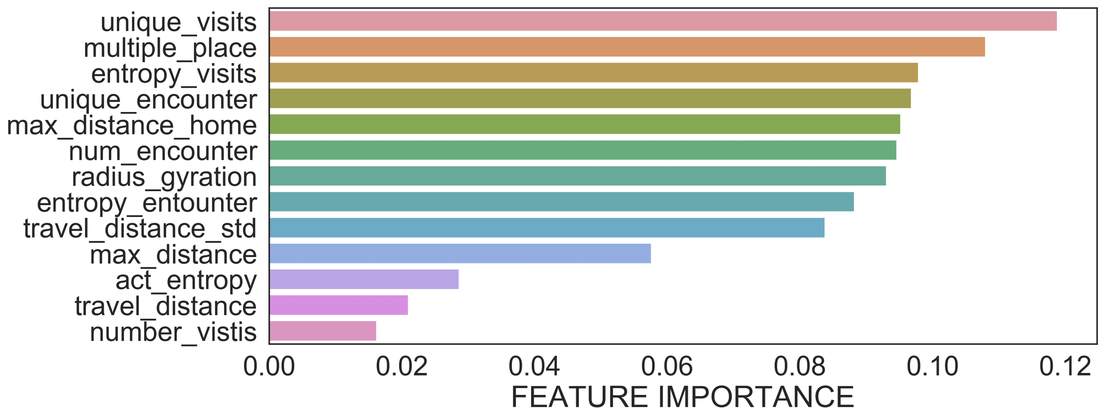

Figure 4.8: Feature Importance plot of Random Forest for Handcrafted Features.

interactions of human states. However, engineered features cannot show these granular changes of human states as the node importance plot.

Hyperparameter Sensitivity Analysis

We explored numerous crucial hyperparameters that might affect the prediction performance of the fine-tuned GraphSAGE model. The findings are shown in Figure 4.9. In plot (a), we see that increasing the learning rate results in inferior prediction performance and also results in a bigger variation in validation loss, indicating that the model's predictions are more variable. In figure (b), varying TopK ratios result in a range of F1-scores between 0.87 and 0.91. There is no apparent pattern for how the preference for TopK ratios affects prediction performance, however 0.6 TopK ratio produces the highest F1-score of 0.91 in the studies. This meant that the top 60% of nodes contributed more to predicting the presence of influenza-like symptoms than the remaining nodes. As seen in panel (c), GraphSAGE may achieve equal validation prediction performance with 50% of data as the training set as models trained with more than 50% data. This has practical implications since samples are costly to collect in mobile sensing and human-centered computing.

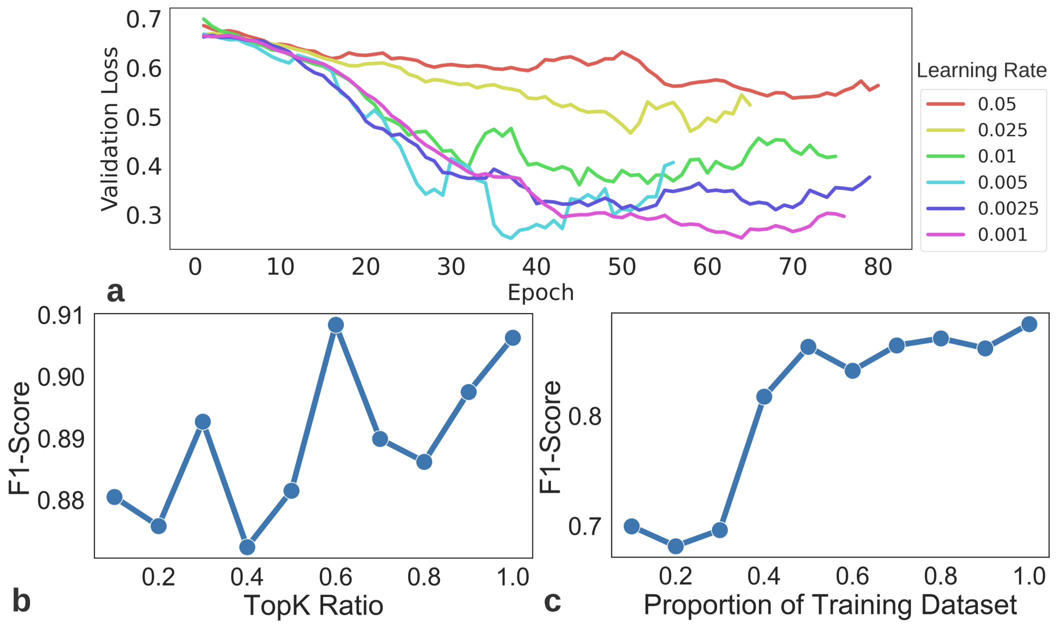

Figure 4.9: Hyperparameter sensitivity analysis for GraphSAGE: a) learning rates on validation loss over training epochs; b) prediction performances on different TopK ratios; c) prediction performances using different proportions of data as training set.

4.5 Summary

We used graphs to represent symptomatic human behaviors acquired by multimodal mobile sensors and illustrated the potential for graph neural networks to identify and forecast human states in this research. This is the first effort that we are aware of that utilizes graph neural networks with multi-modal mobile sensing to describe graph-based behavior. Rather than developing fine-grained behavior markers manually, we proposed using graph representations to encode the dynamics of human behavior state transitions, node embedding techniques to extract topological node attributes, and finally GNNs to learn automatic deep features for influenza-like symptom recognition. We demonstrated that GNNs with GraphSAGE convolutional layers beat baseline models with handmade features by a large margin. Additionally, as seen in Figure Figure 4.1, the graph-based behavior modeling and mobile sensing framework may be applied to applications in mobile sensing, such as those in mental health [114].

Apart from generalizing this paradigm to the realm of mobile sensing, this study has public health implications in terms of circumventing the restrictions of standard public health reporting systems and syndromic monitoring. On an individual level, this approach enables the passive and unobtrusive detection of influenza-like symptoms and the delivery of treatments aimed at changing people' habits such as self-isolation. On a neighborhood and population level, early identification of influenza-like symptoms may alert residents to the possibility of contracting the disease from individuals who are ill and in close proximity, as well as assist in forecasting countrywide influenza levels for public health policy choices.

Our current work is subject to the following limitations. To begin, we constructed global models utilizing data from all participants, assuming that their behaviors and personalities were similar among individuals in our research. A more effective strategy would be to develop customized models by retraining the global model using individual data. Unfortunately, our dataset was insufficient for developing a tailored model owing to missing data and the study's brief duration. Second, utilizing Bluetooth encounters to approximate the closeness of genuine face-to-face social interactions may be skewed [115]. This estimate is based on the premise that a greater number of detected Bluetooth contacts results in a greater likelihood of face-to-face social interactions, which is not necessarily the case. Thirdly, no laboratory-confirmed influenza diagnosis is presented. Influenza symptoms are generic and comparable to those of a variety of other illnesses at an early stage, limiting the use of our study. Finally, to address privacy issues and minimize the possibility of identity disclosure, we anonymized and encrypted GPS coordinates and Bluetooth encounter device IDs. As a result, critical semantic and contextual information about the behavior states represented by our graphs is lost, significantly impairing the decision process's interpretability.

We want to expand our existing work in two ways in the future. To begin, by extending the sequence of mobile sensing data observations to several days, dynamic graphs through time lines can be used to capture temporal variations in behavior states, allowing for the extraction of spatiotemporal representations of human behaviors for the analysis of dynamic interactions between disease symptoms and human behaviors over time. Second, we want to use semi-supervised machine learning approaches to combine labeled and unlabeled data in order to generate more broad models.

Chapter 5

FedMobile: Incremental Federated Learning for Mobile Health

5.1 Background and Motivation

By leveraging the embedded sensors, mobile devices (e.g., smartphones, smartwatches, and wearable devices) pave the pathways to collect human behavioral and physiological signatures passively and continuously [83]. Human-centered data collected by mobile devices have attracted large attention in both industry and academia, revolutionizing modern healthcare systems. More and more healthcare applications , such as physical state inference, mental health monitoring, and mobile interventions, have been investigated by using mobile sensing techniques and machine learning [1]. However, privacy concerns of mobile device users constrained the wide adoption of mobile sensing applications. Data storage location, access authority, and usage of knowledge extraction are among the typical aspects of users' concerns. For instance, Global Positioning System (GPS) trajectories can expose mobile users' traveling history, which can be exploited by adversarial agents to retrieve personal information, such as race, gender, physical activities, social relationship, and health status [116]. Bluetooth devices can be used for contact tracing, as users who contract contagious viruses are generally required to release all their personal information [117].

Privacy preservation constitute a problem in both legal and technical domain, and cannot be simplified by a voluntary consent form that are expected to be agreed by the users [116]. Rigours and robust technical and organizational measures should be carefully designed through engineering processes to protect user privacy in mobile sensing applications. Numerous privacy-preserving approaches and algorithms have been proposed according to different application scenarios. K-anonymity algorithm and its variants have been proposed to hide identities in the source of the data items [118]. Differential privacy aims to reduce exposure of authentic information by adding perturbations that follow certain noise distribution while maintain data utility [119]. However, anonymization techniques can make sensitive attributes vulnerable to inference attacks; and differential privacy can downgrade the data utility and lower the power of learned models [120]. Different from anonymization techniques, Federated Learning (FL) is a decentralized learning algorithm in which users can collaboratively contribute knowledge to the central server without data sharing between the users [42]. When we apply FL in moible sensing apps, each mobile device can contain their own individual data set in their smart devices, and use the local data to train local models, reducing the probability of data leakage; a central server trains a global model by learning from local weights or gradients, rather than learning directly from local data.

Despite the theoretical success of FL algorithm, there are still multiple challenges in FL implementation in mobile sensing applications. First and foremost, mobile sensing techniques collect data continuously and incrementally, and deposit data cumulatively on device. This can cause memory shortage in local mobile devices. What's more, new behavioral patterns can evolve among people. This evolving human behavior/habit can make previous learned knowledge less representative for recognising the trajectory of symptom development. Furthermore, there exists spatial and temporal dependencies in human behavior dynamics (i.e. people's activities can vary over time and locations). Multi-modal raw sensor data without efficient representation engineering can be limited and may fail to capture the complex relationships embedded in people's behaviors. Last but not the least, there are usually a significant amount of unannotated data in mobile sensing data collection, because of prohibitively high cost in annotating ground truth symptom labels. In this chapter, we propose an incremental semi-supervised federated learning algorithm, FedMobile, and a spatio-temporal graph neural network, GCN-LSTM, in an attempt to address these challenges. Our contributions are summarized as follows:

- We propose an incremental federated learning (IFL) framework, FedMobile, to train neural network models decentralizedly by using Knowledge distillation that can transfer expert knowledge to apprentices. This incremental framework is designed to update existing models only using newly available data in an online fashion.
- We integrate the proposed imcremental FL with a semi-supervised method using Consistency Regularization (CR) [121], which can utilize massive amount of unlabeled data to enhance model robustness.
- We apply graph representations to encode complex human behaviors by transforming multi-modal raw sensor data to graphs, and propose a spatio-temporal graph neural network, GCN-LSMT, to fit multi-channel dynamic graph inputs.
- Our proposed FedMobile is a general privacy-preserving method that is applicable for multitudinous mobile sensing and machine learning applications.

5.2 Method

In this section, we firstly introduce our proposed incremental federated learning method, FedMobile, for streaming data in Section subsubsection 5.2. Then, we describe graph rep-

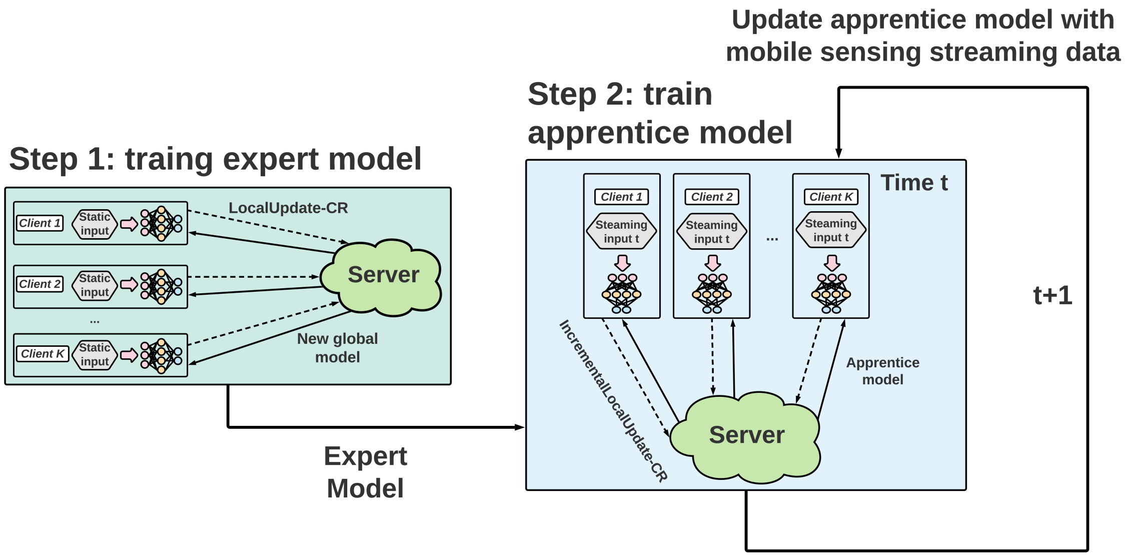
Update apprentice model with mobile sensing streaming data
Step 1: traing expert model
Step 2: train apprentice model
Client 1
Static input
Client 2
Client K
LocalUpdate-CR
Server
New global model
Expert Model
Time t
Steaming input t
IncrementalLocalUpdate-CR
Apprentice model
t+1

Figure 5.1: The general framework of our Incremental Federated Learning Approach.

resentation of mobile sensing data and demonstrate a spatio-temporal graph neural network modeling framework for multi-modal mobile sensing data in Section subsubsection 5.2.

Incremental Federated Learning with Consistency Regularization

For the data streaming in mobile sensing, incremental domain and class problem for the newly collected data can downgrade the prediction power of neural network models that are trained by using historical data [122]. People can change their behaviors over time because of seasonal changes or external events, causing domain shift of mobile sensing data and unclassified outcomes. Modeling retraining that uses all available data can be a default solution. However, retraining models can increase computational burden due to the large volume of data. This computational burden can make on-device FL infeasible. In mobile sensing data collection, it's expensive and challenging to collect sufficient labeled data to train machine learning models supervisedly. Our proposed incremental federated learning (IFL) approach with consistency regularization can overcome these challenges: IFL learns from incoming data incrementally and semi-supervisedly by integrating consistency regularization. The proposed framework consists of two major steps as shown in Figure 5.1:

training 1) an expert model and 2) an apprentice model.

Step 1: training an expert model decentralizedly using consistency regularized local update. For semi-supervised learning prediction challenges, consistency regularization (CR) has showed state-of-the-art performance. [123, 124]. The main concept behind CR is to punish models that are sensitive to disturbed inputs under the premise that the perturbations have no effect on the label semantics. As the global model passes through more rounds of communications in a FL scenario, especially in practice with non-IID data, the FL algorithm might run into a weight divergence problem [125]. For each round of communications, we assume that the micro batches of unlabeled data inside each client stay stable despite minor perturbations. We introduce a new loss component, Kullback–Leibler divergence, $KL(\bullet||\bullet)$, to calculate the divergence between the prediction output distributions of current and previous round of communications. If local models strive to update their model weights in divergent and locally overfitting ways, they will be penalized. The loss function for local update with inter-communication CR is expressed as

$$\mathcal{L}(\mathbf{W}) = \mathcal{L}_{CE}(\hat{\mathbf{y}}, \mathbf{y}) + \mathcal{L}_{CR}(\mathcal{F}(\sigma^{cr}), \mathcal{F}(\sigma^{u})) + \mathcal{R}(\mathbf{W}), \tag{5.1}$$

where $\mathcal{L}_{CE}(\bullet)$ is a cross entropy loss function, σ^{cr} and σ^{u} are logit outputs for unlabeled inputs from previous and current round DL models respectively. $\mathcal{L}_{CR}(\bullet)$ is the consistency regularization loss, where $\mathcal{L}_{CR} = \lambda KL(\mathcal{F}(\sigma^{cr})||\mathcal{F}(\sigma^{u}))$ and $\mathcal{F}(\bullet)$ is a softmax function, $\lambda \in (0, 1)$ is the CR coefficient. $\mathcal{R}(\bullet)$ is the regularization function (e.g. L2-norm). We present **LocalUpdate-CR** in Algorithm algorithm 1. During Step 1 of training expert model distributedly in Figure 5.1, **LocalUpdate-CR** uses semi-supervised FL to update the weights of client models, using both labeled data and a large quantity of unlabeled data, and then uploads the modified weights to the server to update the global model. Finally, an expert model is fed into the next stage of incremental FL to train an apprentice model utilizing streaming data without having to retrain the model from scratch at the conclusion of step 1.

Algorithm 1: Local update with consistency regularization. During each local update for client k, we calculate the inter-communication consistency regularized loss, where overfitting weight updates can be penalized and the weight divergence problem can be mitigated. $\mathbf{W}$ is the DL model weight, $\mathcal{M}$ is the DL model, $\mathbf{X}$ and $\mathbf{X^u}$ are labeled and unlabeled input data, and η is the learning rate.

```
**LocalUpdate-CR** $(k, \mathbf{W}, \mathcal{M}, \mathbf{X}, \mathbf{X}^u, \mathbf{y}, \sigma^{cr})$ :
    for each epoch $i = 1, ..., E$ do
        for each batch $b = 1, ..., B$ do
            $\hat{\mathbf{y}}_b \leftarrow \mathcal{M}(\mathbf{X}_b)$;
            $\sigma_b^u \leftarrow \mathcal{M}(\mathbf{X}_b^u)$;
            $\mathcal{L}(\mathbf{W}) \leftarrow \mathcal{L}_{CE}(\hat{\mathbf{y}}, \mathbf{y}) + \mathcal{L}_{CR}(\mathcal{F}(\sigma^{cr}), \mathcal{F}(\sigma^u)) + \mathcal{R}(\mathbf{W})$;
            $\sigma_b^{cr} \leftarrow \sigma_b^u$;
            $\mathbf{W} \leftarrow \mathbf{W} - \eta \nabla \mathcal{L}(\mathbf{W})$;
        end
    end
    return $\mathbf{W}, \sigma^{cr}$;
```

Step 2: training apprentice model using incremental local update. The expert model and the apprentice model are the two structures that make up knowledge distillation (KD). An expert model is a model that has been trained using representative data from the domain population or a model that has already been trained. The expert model's knowledge is transmitted to the apprentice model by reducing the distance between their prediction distributions given the identical inputs in the distillation phase [126]. In incremental FL, for the incoming batch of streaming data at time t, which includes both labeled and unlabeled samples, the loss function for the client local update is expressed as:

$$\begin{aligned}\mathcal{L}(\mathbf{W}^\rho) = (1-\alpha)\mathcal{L}_{CE}(\hat{\mathbf{y}}_\mathbf{t}, \mathbf{y}_\mathbf{t}) + \alpha\mathcal{L}_{KD}(\mathcal{F}(\sigma_t^\tau), \mathcal{F}(\sigma_t^\rho)) \\ + \mathcal{L}_{CR}(\mathcal{F}(\sigma^{cr}), \mathcal{F}(\sigma_t^u)) + \mathcal{R}(\mathbf{W}^\rho),\end{aligned} \tag{5.2}$$

where W^ρ is the apprentice model weights, $\alpha \in (0, 1)$ is a balancing weight, σ_t^τ and σ_t^ρ are the logit outputs from the expert and apprentice model respectively. $\mathcal{L}_{KD}$ is the knowledge distillation loss, and $\mathcal{L}_{KD}(\mathcal{F}(\sigma_t^\tau), \mathcal{F}(\sigma_t^\rho)) = KL(\mathcal{F}(\sigma_t^\tau)||\mathcal{F}(\sigma_t^\rho))$ [126, 127]. **IncrementalLocalUpdate-CR** is presented in Algorithm algorithm 2. In Step 2 shown in

Figure 5.1, given streaming data input at time t, incremental FL uses **IncrementalLocalUpdate-CR** to update client local models and then sends the updated weights to the server, which further aggregates the client model weights and updates the global model weights.

Algorithm 2: Incremental local update with consistency regularization. In each incremental update for each batch of incoming streaming data at time t, the apprentice model is updated locally by leveraging the knowledge generalized from the expert model by using knowledge distillation from incremental learning.$\mathbf{W}^{\rho}$ is the model weight of the apprentice model, $\mathcal{M}^{\rho}$ and $\mathcal{M}^{\tau}$ are apprentice DL model and expert DL model respectively.

```
IncrementalLocalUpdate-CR ($k, \mathbf{W}^{\rho}, \mathcal{M}^{\rho}, \mathcal{M}^{\tau}, \mathbf{X}_t, \mathbf{X}^u_t, \mathbf{y}_t, \sigma^{cr}$):
    for each epoch $i = 1, ..., E$ do
        for each batch $b = 1, ..., B$ do
            $\hat{\mathbf{y}}_{t,b}, \sigma^{\rho}_{t,b} \leftarrow \mathcal{M}^{\rho}(\mathbf{X}_{t,b})$;
            $\sigma^{u}_{t,b} \leftarrow \mathcal{M}^{\rho}(\mathbf{X}^{u}_{t,b})$;
            $\sigma^{\tau}_{t,b} \leftarrow \mathcal{M}^{\tau}(\mathbf{X}_{t,b})$;
            $\mathcal{L}(\mathbf{W}^{\rho}) \leftarrow (1-\alpha)\mathcal{L}_{CE}(\hat{\mathbf{y}}_{\mathbf{t}}, \mathbf{y}_{\mathbf{t}}) + \alpha\mathcal{L}_{KD}(\mathcal{F}(\sigma^{\tau}_{t}), \mathcal{F}(\sigma^{\rho}_{t})) +$
              $\mathcal{L}_{CR}(\mathcal{F}(\sigma^{cr}), \mathcal{F}(\sigma^{u}_{t})) + \mathcal{R}(\mathbf{W}^{\rho})$;
            $\sigma^{cr}_{b} \leftarrow \sigma^{u}_{t,b}$;
            $\mathbf{W}^{\rho} \leftarrow \mathbf{W}^{\rho} - \eta\nabla\mathcal{L}(\mathbf{W}^{\rho})$;
        end
    end
    return $\mathbf{W}^{\rho}, \sigma^{cr}$;
```

End-to-End Incremental FL. Based on **LocalUpdate-CR** and **IncrementalLocalUpdate-CR**, we present the end-to-end incremental FL framework, as shown in Algorithm algorithm 3. Given static labeled input X^l, the corresponding target y^l and unlabeled input X^u from each client k, the goal of step 1 is to train a global expert model using semi-supervised FL with both labeled and unlabeled local data. First, the server initializes W_0 for the global model $\mathcal{M}_{\prime}$, and the logit output $\sigma^{cr}_{0,\bullet} \sim Uniform(0,1)$ for the inter-communication CR. Then for each round of communications, we run **LocalUpdate-CR** for each client k to update the local models, and send the updated weights to the server. The server aggregates the new local weights using FedAvg [128]. After local weight aggregation, the global model weights are updated in the server. Finally, after C rounds of communications, the first stage of incremental FL outputs the trained expert model denoted as $\mathcal{M}^{\tau}$. In step 2, given the expert model $\mathcal{M}^{\tau}$, streaming inputs $X^{l}_{t,k}$, $X^{u}_{t,k}$ for each client k at time t, we train an apprentice model $\mathcal{M}^{\rho}$ in a decentralized manner using streaming data and knowledge learned from the expert model to approximate the performance of a centralized-trained model. At each incremental iteration t, we run C rounds of communications to train the apprentice model. In each round of communications, we run **IncrementalLocalUpdate-CR** to update local model weights and send the updated weights to the server. We then aggregate the local model weights and update the apprentice model weights incrementally.

Algorithm 3: End-to-end incremental federated learning. As shown in Figure 5.1, this incremental FL method first trains a global expert model using FedAvg with **LocalUpdate-CR**, and then feeds the trained expert model into the second step of incremental FL. In the second step, a global apprentice model is trained using FedAvg with **IncrementalLocalUpdate-CR**.

```
Step 1: train expert model
Server executes:
    initialize $\mathbf{W}_0$ for $\mathcal{M}_0$;
    initialize $\sigma^{cr}_{0,\bullet} \sim Uniform(0,1)$;
    for each round of communications $c = 0, 1, 2..., C$ do
        for each client $k \in K$ in parallel do
            $\mathbf{W}_{c+1,k}, \sigma^{cr}_{c+1,k} \leftarrow$ LocalUpdate-CR($k, \mathbf{W}_{c,k}, \mathcal{M}_c, \mathbf{X}^l_{c,k}, \mathbf{X}^u_{c,k}, y_{c,k}, \sigma^{cr}_{c,k}$);
        end
        $\mathbf{W}_{c+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} \mathbf{W}_{c+1,k}$;
        $\mathcal{M}_{c+1} \leftarrow$ update $\mathcal{M}_c$ by $\mathbf{W}_{c+1}$;
    end
    return expert model $\mathcal{M}^\tau$;
Step 2: train the apprentice model using streaming data and knowledge distilled
        from the expert model
Server executes:
    initialize $\mathbf{W}^\rho_0$ for $\mathcal{M}^\rho_0$;
    for streaming data at time $t = 0, 1, 2...$ do
        $\mathbf{W}^\rho_t \leftarrow \mathbf{W}(M^\rho_t)$;
        for each round of communications $c = 0, 1, 2..., C$ do
            for each client $k \in K$ in parallel do
                $\mathbf{W}^\rho_{t,k}, \sigma^{cr}_{t,k} \leftarrow$ IncrementalLocalUpdate-
                 CR($k, \mathbf{W}^\rho_{t,k}, \mathcal{M}^\rho_t, \mathcal{M}^\tau, \mathbf{X}^l_{t,k}, \mathbf{X}^u_{t,k}, \mathbf{y}_{t,k}, \sigma^{cr}_{t,k}$);
            end
            $\mathbf{W}^\rho_t \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} \mathbf{W}^\rho_{t,k}$;
            $\mathcal{M}^\rho_t \leftarrow$ update $\mathcal{M}^\rho_t$ by $\mathbf{W}^\rho_t$;
        end
        $\mathcal{M}^\rho_{t+1} \leftarrow \mathcal{M}^\rho_t$
    end
    return apprentice model $\mathcal{M}^\rho$;
```

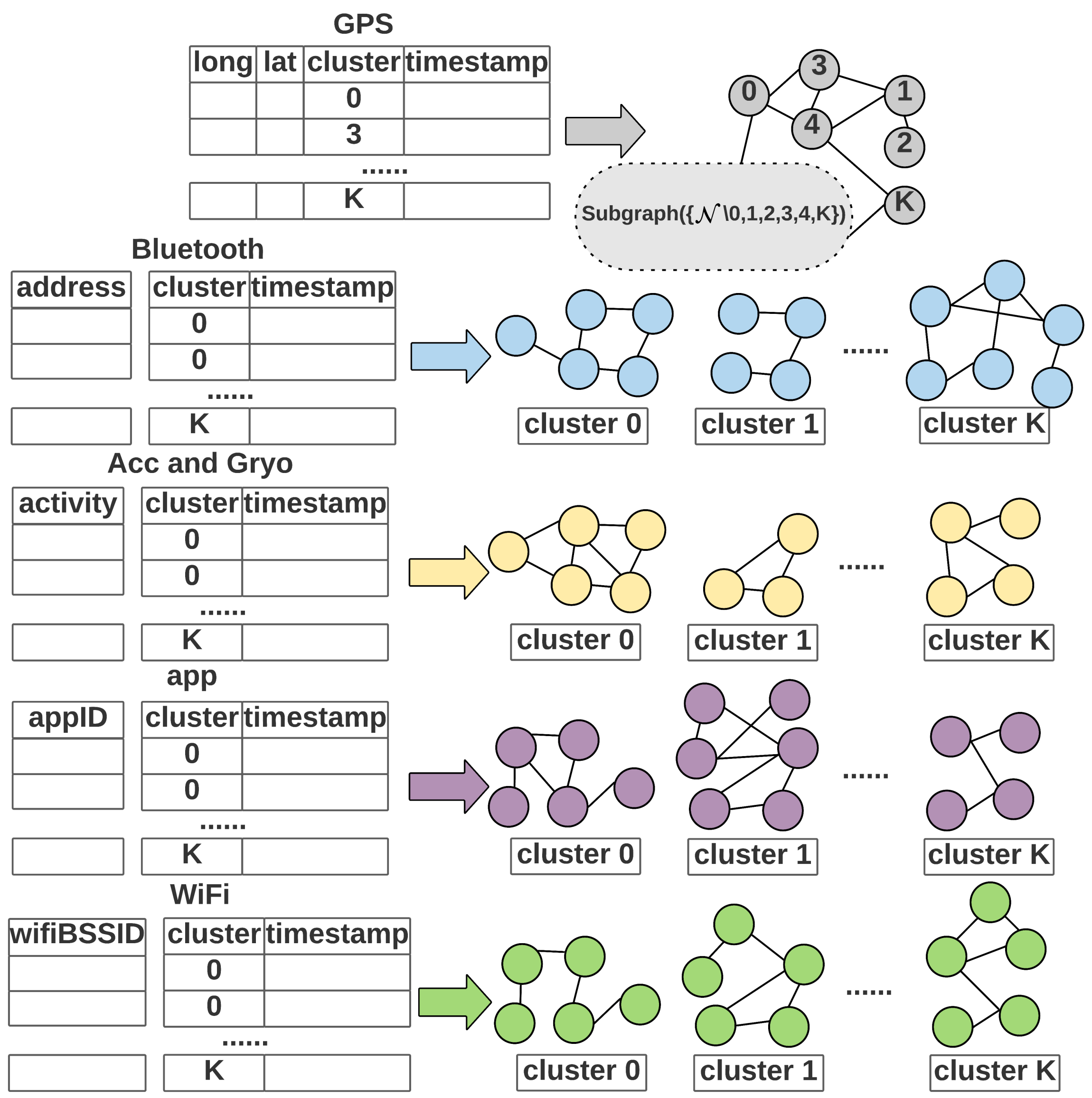

Figure 5.2: Graph Representations of Mobile Sensing Data. People can have varying behaviors from place to place. Thus, we generate graphs to represent the behaviors at each corresponding geospatial cluster.

Spatio-temporal Graph Neural Networks in Mobile Sensing

Graph Representations of Mobile Sensing Data: Machine learning studies of mobile sensing have investigated at handcrafted feature engineering extensively. Complex connections and high-level dependency among human state fluctuations may be underrepresented in manually derived features, despite fine-grained featurization of raw mobile sensing data and adequate interpretability. Furthermore, since structural topology occurs in non-euclidean domains of raw mobile sensing data (e.g., GPS, Bluetooth Encounter), capturing such topological information through handcrafted feature engineering can be difficult [129]. We employ graphs to describe human behaviors in this paper, and we leverage mobile sensing to illustrate the feasibility and promise of FedMobile-trained GCN-LSTM on an influenza-like symptom identification challenge. We believe that persons who are in the early stages of influenza will become less active unconsciously, avoiding needless travel and interpersonal contacts. We use a variety of embedded sensors to collect data on human behavior: Human mobility behaviors are captured by GPS sensors; Bluetooth encounters are used to evaluate social behaviors; accelerometer and gyroscope can monitor physical activity; web-virtual behaviors are recorded by recording app usages; and environmental contexts may be inferred by WiFi signals.

In graph construction process as shown in Fig Figure 5.2, for mobility behavior, we consider each cluster label as a node and create edges between every pair of successive visited sites given a series of cluster labels obtained from clustering of GPS coordinates that capture people's daily moving route. We divide the multi-modal sequential discrete sensor signals into numerous sub-sequences for the other behaviors and surrounding environment, with each sub-sequence representing the behaviors or contexts at the corresponding site individuals visit. We express each identified human activity as a node in graphs for accelerometer and gyroscope, for example, and add edges between every two successive activity states in the table. Then, for each cluster partition, we use this graph creation technique to produce a succession of graphs. Each sample in the GCN-LSTM input comprises of a static graph that encodes mobility behavior and multi-channel dynamic graphs that capture social, activity, web-virtual behaviors, and ambient surroundings, as well as a static graph that encodes mobility behavior. Finally, the constructed graph structured data is used to identify influenza-like symptoms early on.

Spatio-temporal Graph Neural Networks: During the course of a day, people may visit various sites and act differently in each of them. When individuals are at work or on campus during the day, for example, they engage in more diversified activities, but when they are at home after work, they engage in less activities. Topology changes in graphs are triggered by various actions, showing the presence of spatial-hierarchical and temporal differences in human activity. We propose spatio-temporal graph neural network, GCN-LSTM, to model the dynamic graphs with spatial hierarchy. The overall architecture of GCN-LSTM is shown in Figure 5.3. In general, we employ GCN to generate graph embeddings that capture behavior dynamics, and then feed the series of multi-channel graph embeddings into an LSTM to infer health outcomes. In this study, we denote the adjacency matrix of G_t as A_t, and generate node attribute matrix, denoted as X_t, using graph embedding methods [130]. To capture the topological structure and transition dynamics of people's traveling patterns, different graph convolutions including GCN [93] and GAT [95] can be applied to extract the high-level representations of the graph inputs. We use GCN in this study, and the activated layer-wise propagation in a GCN can be expressed as

$$Z = ReLU(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}\mathbf{W}), \tag{5.3}$$

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, $\mathbf{I}$ is the identity matrix, $\mathbf{D}$ is the diagonal matrix, $\tilde{\mathbf{D}} = \mathbf{D} + \mathbf{I}$, and $\mathbf{W}$ is the learnable weight matrix. To capture the temporal variations and inter-transition dependence between visited clusters, we feed the output from GCN, denoted as $\mathbf{Z}^p = [z_1^p, ..., z_T^p]$, to a LSTM layer [13]. Each cell in LSTM can be expressed as

$$i_t = \sigma(\mathbf{W}_{zi}z_t^p + \mathbf{W}_{hi}h_{t-1} + \mathbf{W}_{ci}c_{t-1} + b_i) \tag{5.4}$$

$$f_t = \sigma(\mathbf{W}_{zf}z_t^p + \mathbf{W}_{hf}h_{t-1} + \mathbf{W}_{cf}c_{t-1} + b_f) \tag{5.5}$$

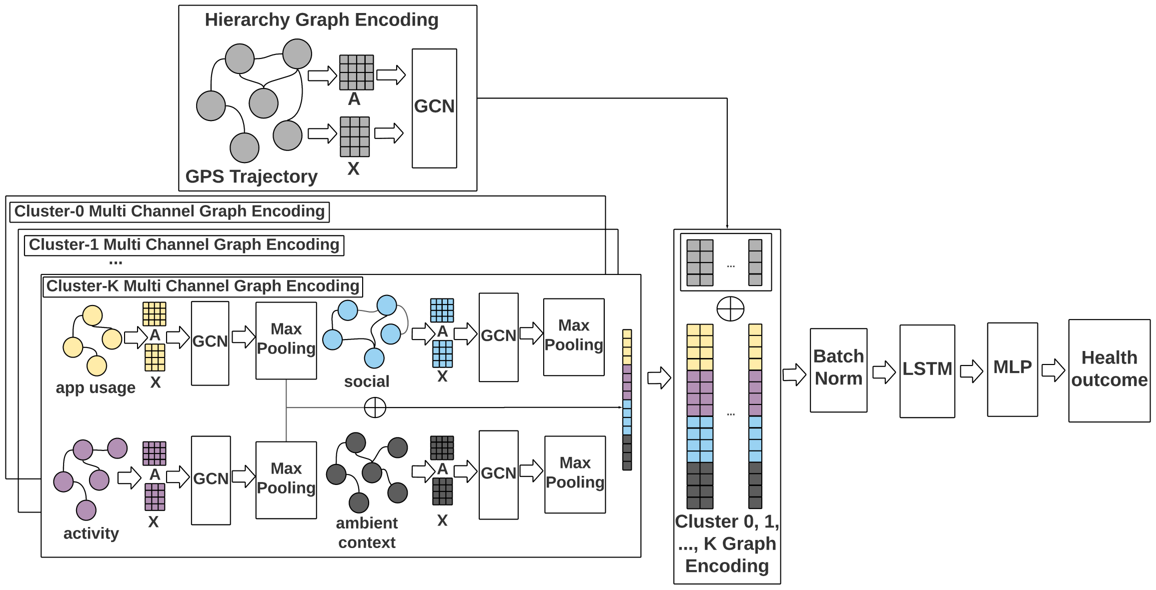
Hierarchy Graph Encoding
GPS Trajectory
A
X
GCN
Cluster-0 Multi Channel Graph Encoding
Cluster-1 Multi Channel Graph Encoding
Cluster-K Multi Channel Graph Encoding
app usage
social
activity
ambient context
Max Pooling
Cluster 0, 1, ..., K Graph Encoding
Batch Norm
LSTM
MLP
Health outcome

Figure 5.3: GCN-LSTM architecture. **A** and **X** represent the adjacency matrix and feature matrix respectively for each graph.

$$c_t = i_t tanh(\mathbf{W}_{zc} z_t^p + \mathbf{W}_{hc} h_{t-1} + \mathbf{W}_{cc} c_{t-1} + b_c) + f_t c_{t-1} \tag{5.6}$$

$$o_t = \sigma(\mathbf{W}_{zo} z_t^p + \mathbf{W}_{ho} h_{t-1} + \mathbf{W}_{co} c_{t-1} + b_o) \tag{5.7}$$

$$h_t = o_t tanh(c_t), \tag{5.8}$$

where i_t, f_t, c_t, o_t denote the states of the input gate, forget gate, memory cell, and the output gate respectively, and h_t denotes the output of LSTM at time t.

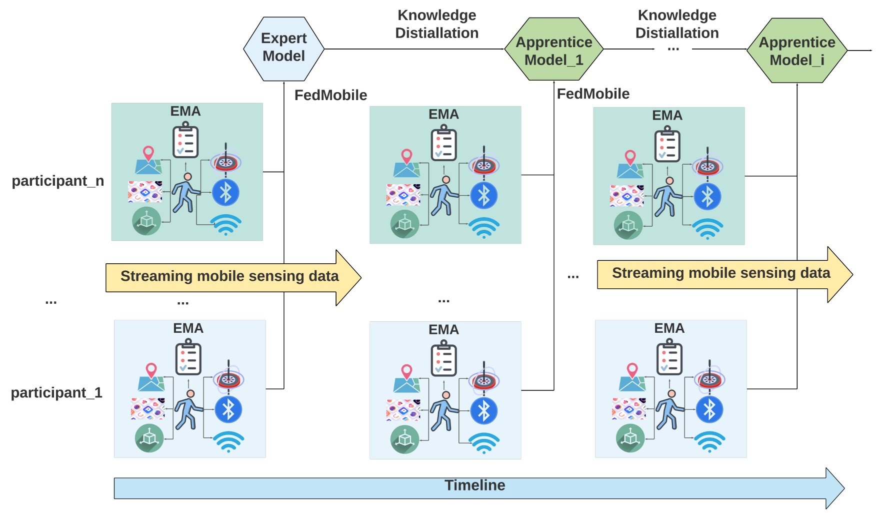

Figure 5.4: FedMobile is a mobile sensing data streaming platform. FedMobile may update global models gradually in mobile sensing data streaming by solely utilizing the freshly arriving batch of mobile sensing data. We establish an apprentice model based on the previously trained expert model, and then update the apprentice model using just the incoming batch of streaming data in a federated online approach..

5.3 Data and Experiments

Data collection and description

The data for this influenza-like symptom identification experiment came from a multi-cohort mobile sensing study we did to help with illness detection early. In all, 2,700 people were recruited from 24 states throughout the United States to take part in the data collecting. This research lasted one year in total. GPS, Bluetooth encounter, Wifi signal, app use, accelerometer, gyroscope, and other sensor data were recorded. We requested participants to download and run ReadiSens, a mobile sensing app. Fever, cough, trouble breathing, exhaustion, muscular pains, headache, sore throat, runny nose, nausea, and diarrhea were among the self-reported influenza symptoms collected by ecological momentary assessments (EMAs) provided at 8 p.m. everyday. Only individuals with at least 14 days of data were chosen to assure the validity of our studies, resulting in data from about 800 people for this research. From February 15th, 2019 through April 30th, 2020, active data collecting was undertaken on a rolling basis. The overall number of annotated and unannotated daily observations was roughly 6,100 and 22,000, respectively, with 18 percent reporting influenza symptoms. The average age of the included participants in this chapter was $42.78(sd = 10.16)$. 64.36% of the participants were female, with white being 66.3%, African American 20.8%, Asian 6.5%, multiple races 3.2%, Hispanic 3.1%, and others 0.1%. Among the participants, 61% were full time workers, 14% were full time students, 12% were working part time, 8% had retired, 3% were care takers, while 2% were temporarily unemployed.

Implementation details

We split the labeled data set into 70%, 20%, and 10% subsets for training, validation, and testing, respectively, in the centralized training process and baseline Federated Learning approaches. For the centralized training procedure, we apply semi-supervised incremental learning. We replicate mobile sensing streaming data and execute FedMobile utilizing the data in chronological sequence throughout the incremental FL training procedure. Step 1 of FedMobile is used to create an expert model using the first 40% of training data. as described in Section subsubsection 5.2. In the step 2 of FedMobile, as described in Section algorithm 5.2, the number of incremental updates was set to 20, and the simulated streaming data was divided into 20 batches based on their chronological order for incremental training. The overall training process of FedMobile for streaming mobile sensing data is shown in Fig Figure 5.4. To assign daily mobile sensing observations, 40 pseudo dispersed clients are developed. Each pseudo distributed client collects local data from participants who are part of the same G-means-generated demographic profile cluster [64]. Without regard for privacy, we train the models using Adam [131] on a central server to see if federated (privacy-protected) techniques can achieve equivalent performance to the centralized method.

Baselines

Federated Learning: a) FederatedAveraging (FedAvg) [128] is a well-known supervised federated learning method that is often used as the baseline model in federated learning research. FedAvg's shortcoming is that it is limited to supervised federated learning tasks and is unable to learn from vast amounts of unlabeled data.

b) FedSem [132] is a semi-supervised federated learning approach that makes use of pseudo-labeled data to enhance the performance of federated learning. Pseudo labeling is a semi-supervised approach for predicting the labels of unannotated data using pre-trained models and combining them with labeled data to train new models [133].

c) FedFixMatch is a naive combination of FedAvg and FixMatch [134] that incorporates pseudo-labeling and consistency regularization for semi-supervised learning.

d) Federated Matching (FedMatch) [135] performs semi-supervised federated learning by using a novel inter-client consistency loss and disjoint learning for labeled and unlabeled data.

Neural Networks: a) MLP: multi-layer perceptron [109] is widely regarded as the standard model for connecting deep learning approaches to the area of mobile sensing. We fit the node attribute matrices (extracted using a graph embedding approach called NodeSkech [100]) using MLP without incorporating the graph's topological structures (adjacency matrices) in the input.

b) GCN [93] is a baseline deep learning model for graph structured data. By concatenating the outputs of a succession of GCNs, we may utilize GCNs to represent a sequence of graph inputs. We have a matching GCN for each graph from a single sensor channel in the input of the graph sequence, which generates the high-level features and feeds them into a dense layer for prediction.

c) GAT: graph attention network [95] makes use of attention processes to generate node representations by assigning more weight to significant nodes and (or) characteristics. GAT is used in the same way as GCN to extract high-level embeddings of graph inputs without

Model	F1-Score				PR-AUC			
	MLP	GCN	GAT	GCN-LSTM	MLP	GCN	GAT	GCN-LSTM
Centralized	0.73	0.75	0.76	0.81	0.76	0.79	0.84	0.88
FedAvg	0.65	0.66	0.68	0.72	0.68	0.71	0.73	0.75
FedSem	0.67	0.69	0.70	0.73	0.70	0.72	0.76	0.81
FedFixMatch	0.69	0.71	0.72	0.75	0.71	0.75	0.80	0.83
FedMatch	0.70	0.72	0.74	0.78	0.71	0.77	0.81	0.83
FedMobile(ours)	0.72	0.73	0.75	0.80	0.74	0.78	0.83	0.86

Table 5.1: Performance comparison from two perspective: one is within federated learning methods, and between FedMobile and the centralized training paradigm; another one is between the proposed GCN-LSTM model and the baseline neural networks.

taking into account the graph's temporal connection.

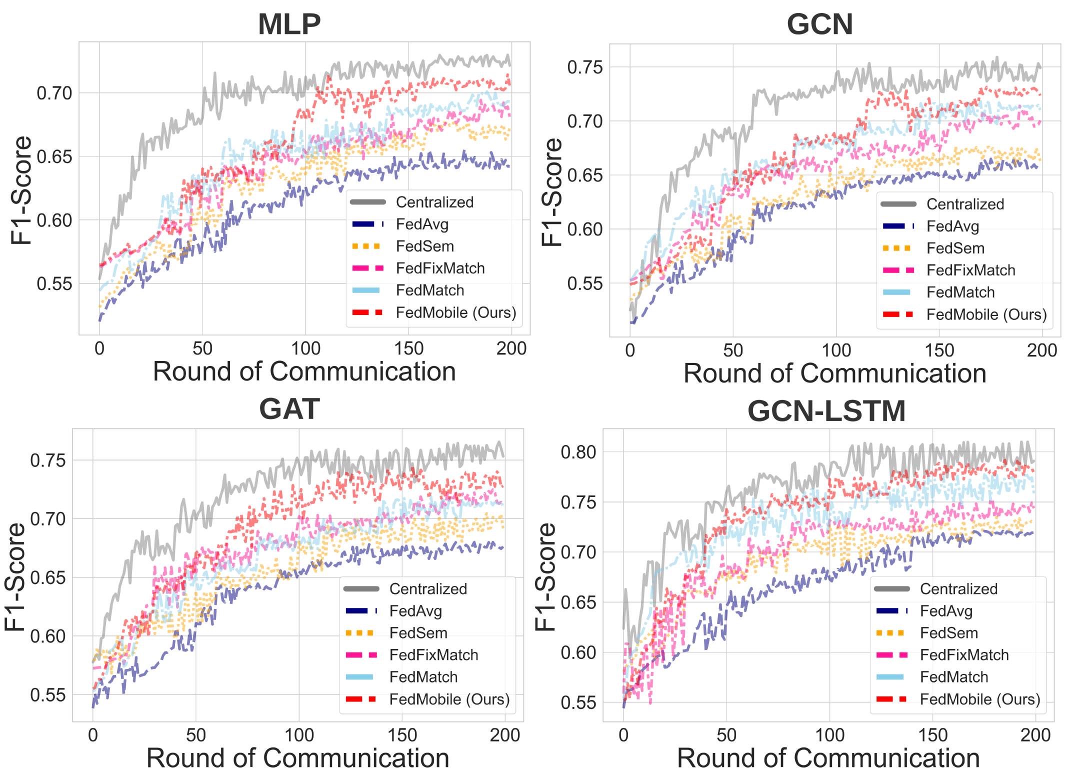

Figure 5.5: Testing F1-Score comparison FedMobile and the baseline models for 200 communication rounds (epochs for centralized training). In the FedMobile, we train the expert models in the first 80 communication rounds and perform incremental update through 80-200 communication rounds.

5.4 Results and Analysis

We compare our proposed FedMobile and GCN-LSTM methods from two perspectives in this study: 1) we compare FedMobile to other FL baseline models and to a centralizedly trained model that incorporates incremental learning and consistency regularization; and 2) we compare baseline model performance to the proposed GCN-LSTM model performance in various training scenarios. We use F1-Score and PR-AUC as evaluation metrics to compare model performance. The evaluation performances are shown in Table Table 6.1. We observe that models trained using FedMobile provide outcomes equivalent to those trained centrally. Models trained on FedMobile outperform models trained on other FL baselines. As shown in Fig Figure 5.5, we observe that, generally, in the first half of the communication rounds, FedMobile exhibits a similar training trajectory to FedMatch or FedFixMatch, as FedMobile trains an expert model without incremental learning in the first step; however, FedMobile exhibits an increasing F1-Score in the second step (after 80 communication rounds), implying incremental knowledge gains. FedMobile's higher performance may be ascribed to numerous components of the incremental federated learning architecture's design. To begin, the knowledge distillation (KD) property, which penalizes new knowledge in order to mitigate overfitting and adapt to it after several iterations of incremental learning, can perform fine-tuning in the second step of training an apprentice model, such that the final few layers (LSTM+MLP) in the GCN-LSTM model can learn better fitted weights [127]. Second, FedMobile's Consistency Regularization (CR) technique leverages a large number of unannotated samples to improve the robustness of GNNs by punishing significant changes in the inferred label semantics across communication rounds. Additionally, CR minimizes the divergence of local weight updates across dispersed clients, which helps stabilize the global model parameter optimization. In terms of neural networks, the

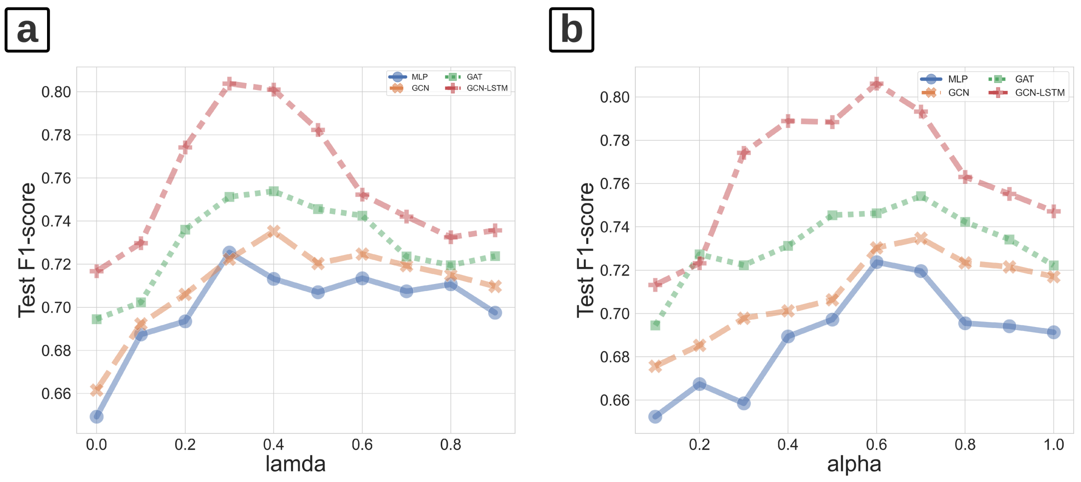

Figure 5.6: Impact of (a) Consistency Regularization coefficient λ and (b) KD loss coefficient α to the performance of FedMobile-trained model.

suggested GCN-LSTM model outperforms baseline neural network models. Due to the fact that individuals exhibit a variety of behaviors when they visit various locations, the topological structures of graph representations of sensing signals may exhibit spatial and temporal changes through people's traveling trajectories. GCN-LSTM employs GCN to capture high-level spatial and topological properties and LSTM to encode temporal information in a graph sequence.

Hyperparameter Analysis We analyze the impact of Consistency Regularization (CR) coefficient λ and the Knowledge Distillation (KD) coefficient α on the performance of FedMobile-trained models. As shown in Figure 5.6 (a), with a CR coefficient of 0.3/0.4, the highest performing models can be achieved. This discovery suggests that utilizing unlabeled data to impose an excessive/insufficient penalty (large/small $lambda$) on the prediction outcomes cannot alleviate within-client inconsistency, hence making the models less resilient. When

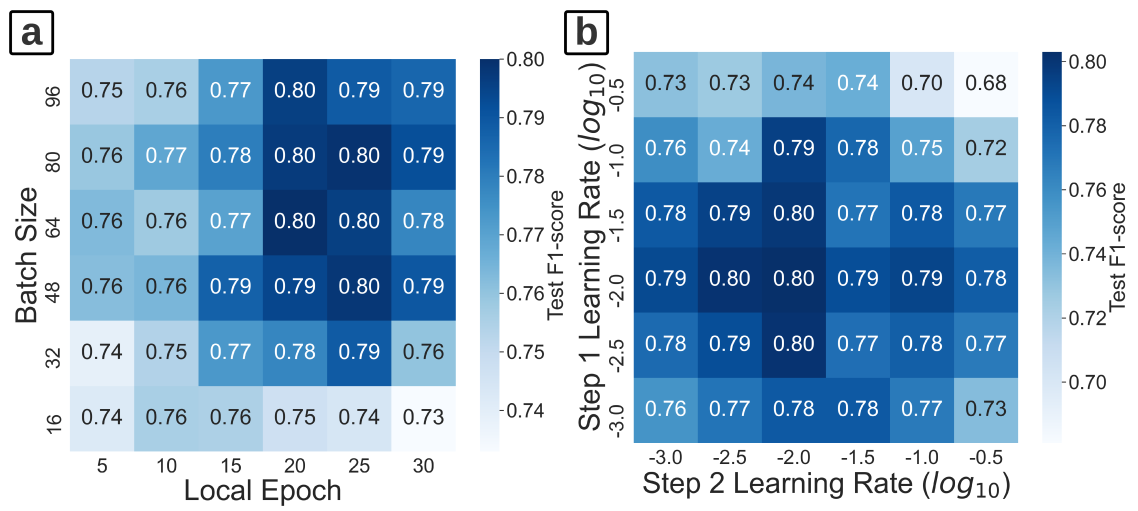

Figure 5.7: The impact of batch size and number of local epochs, and learning rate in step 1 and 2 of FedMobile to the performance of GCN-LSTM.

the CR coefficient is equal to zero (FedMobile without CR), performance degradation may also suggest the regularization impact of CR. In Figure 5.6 (b), when $alpha$ is close to 0 or 1, the models perform poorly, showing that entire dependence on either the expert or apprentice models might have a detrimental effect on model generalization. And a value of $alpha$ of 0.6/0.7 may provide the highest results, which can be explained by the fact that previously taught information can act as a type of regularization, preventing the trained models from overfitting in response to fresh incoming data with possible domain shifts.

In Figure 5.7 (a), we demonstrate the impact of varying the size of the local batch and the number of local epochs on a FedMobile-trained GCN-LSTM. We notice that a small batch size and a limited number of local epochs impair the prediction performance of GCN-LSTM. This might be because a limited batch size results in less representative anticipated soft labels that do not accurately reflect robust label distribution, making the CR less effective.. In Figure 5.7 (b), We demonstrate that the performance of FedMobile-trained GCN-LSTM is also reliant on the expert and apprentice model learning rates. In general, we can train the apprentice model at a lower learning rate than the expert model. Additionally, we note that when the apprentice model is built up with [LSTM:MLP] (fine-

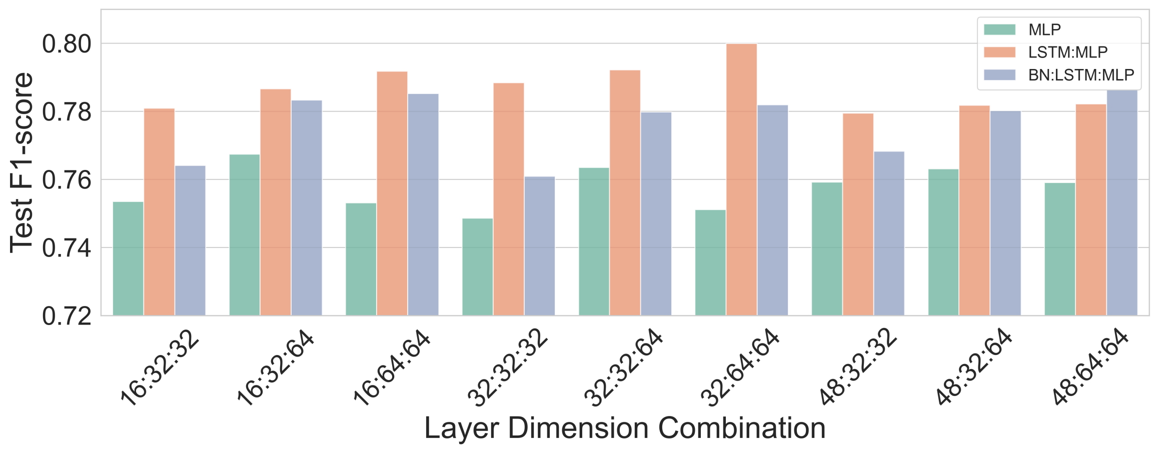

Figure 5.8: The impact of incremental updating (fine-tuning) layers and layer dimension of GCN:LSTM:MLP to the performance of GCN-LSTM.

tuning) layers, it performs better than when simply the MLP layer and the BN:LSTM:MLP layer are used, as shown in Figure 5.8. The GCN-LSTM with layer dimensions 32:64:64 for GCN:LSTM:MLP shows the best performance.

5.5 Summary

We develop FedMobile, an incremental semi-supervised Federated Learning framework, and propose a spatio-temporal graph neural network, the GCN-LSTM, for modeling graph representations of mobile sensing data in this study. FedMobile has proved its ability to train neural network models with continuous streaming data collection. It incorporates Knowledge Distillation with Consistency Regularization, allowing FL models to be updated online, alleviating compute and memory load, and allowing for the use of vast amounts of unlabeled data to eliminate within-client inconsistency. GCN-LSTM is a spatiotemporal graph neural network model that is aware of its spatial hierarchy. It is capable of capturing the geographic hierarchy that exists between spatial settings and the desired human actions. GCN-LSTM may also be used to represent the dynamics of topological alterations in human actions for health inference. The experiment we did with 800 participants' mobile sensing data reveals that GCN-LSTM trained with FedMobile can identify the presence of influenza-like symptoms in a privacy-preserving manner without impairing prediction capacity.

This approach has implications for privacy-preserving machine learning applications in mobile health. To begin, FedMobile demonstrates a viable technique for deploying Federated Learning on-device in order to address the memory scarcity problem in mobile devices. Additionally, the assessment findings indicate that FedMobile may be utilized for privacy-preserving individual-level health monitoring and inference without jeopardizing the predictive ability of neural network models. Additionally, the Centers for Disease Control and Prevention (CDC) now depends on collaborative efforts ranging from local clinics to state agencies to report instances of influenza virus infection [77]. However, this case report method is notorious for being unreliable and prone to underestimation. By implementing our suggested mobile sensing technology in practice, we can overcome the limits of standard public health reporting and symptom monitoring. Finally, if influenza-like symptoms are recognized, treatments may be offered to alter an individual's behavior (e.g., self-isolation). The actions may be scaled up to a community level to fight influenza virus spread.

Additionally, this approach has multiple drawbacks. To begin, while the mobile sensing data is obtained in the field, the experiments are run using simulations in an incremental learning environment utilizing aggregated data from each client. Because a client's data is derived from several individuals, it may result in increased heterogeneity in the data. However, in practice, every individual user may be viewed as a client, and so the data from each client will have a greater degree of homogeneity, which may have been overlooked in our present suggested technique. Second, we assume that clients update their local models simultaneously in our research. However, in the actual world, each mobile device has a unique processing capability, making synchronous federated learning impossible. Thirdly, we excluded medical diagnosis of influenza from the sample of individuals who self-reported experiencing similar symptoms. Due to the similarity of influenza symptoms to those of other illnesses, particularly in the early stages of sickness, the clinical importance of the prediction results may be diminished.

FedMobile will be available on smartphones in the future, and we will establish a digital platform for decentralized human health monitoring. Additionally, to enable individual-level health monitoring, it is important to design a strong Federated Learning algorithm capable of operating on non-identical and dispersed (non-IID) data with a stationary global update. Additionally, we want to expand the current sensor network to incorporate audio data, physiological data (heart rate, for example), and text messages. Additionally, tailored FL for lifelong learning is an area worth researching.

Chapter 6

Semi-supervised Graph Instance Transformer for Mental Health Inference

6.1 Background and Motivation

Not the same as physical health that is easy to recognize, mental health acts as an essential role in human systems. Anxiety and depression are widely spread around modern lives because of demanding financial stress and suffocative social pressure [136]. It's necessary to develop effective and efficient approaches to detect and recognize mental health symptoms as early as possible and prevent people from negative consequence of being suffering from mental illness. Personal sensing techniques provide tremendous opportunities for mental health inference: people's daily behaviors can be impacted by their mental health status, and the daily activity trajectories can be profiled by mobile and wearable devices.

By leveraging mobile sensing techniques, we can apply and develop machine learning algorithms for early detection of the symptoms of anxiety disorder and depression. Boukhechba et al. created predictive models for social anxiety recognition by using Global

Positioning System (GPS) traces [137]. Digital behavioral markers extracted from mobile sensing data, which are collected by using smartphones from college students, are used to predict depression in some existing researches [138, 139]. Desipte the success of using machine learning models to predict mental health symptoms, there are still challenges in mobile sensing studies for mental health inference. First and foremost, collecting large amounts of labeled data involving with substantial number of participants to train robust machine learning models is challenging in mobile sensing studies. Secondly, self-reported results from ecological momentary assessments (EMAs) that are used to generate ground truth information requires heavy user burdens, resulting in sparsely annotated data. What's more, traditional machine learning algorithms that are used to fit mobile sensing data are largely relying on handcrafted feature engineering, which makes the trained models with low level of generalizability. Additionally, in multi-modal sensory environment, handcrafted features cannot serve as comprehensive representations of the raw sensing data.

In this section, we propose a semi-supervised Graph Instance Transformer (SS-GIT) that is able to allievate the above mentioned limitation from previous mobile sensing studies for mental health inference. We transform the mental health inference problem as a multiple instance learning problem, and utilize the unlabeled data to enhance the robustness of GIT by using a semi-supervised method referred as Contrastive Self-supervised Learning (CSSL). In SS-GIT, raw mobile sensing data collected from multiple sensing resource are represented by graphs and transformed to multi-channel graph instance sets. Each set of graph instances consists of multiday human behavior trajectories, and is fed to GIT to generate instance embeddings and finally fed into multiple instance learning pooling layer to generate bag representations.

Our contributions are summarized as follows:

- We propose a permutation invariant graph neural network model for early detection of mental illness symptoms under weak supervision.

- We integrate graph neural network (GNN), set transformer, and attention-based mul-

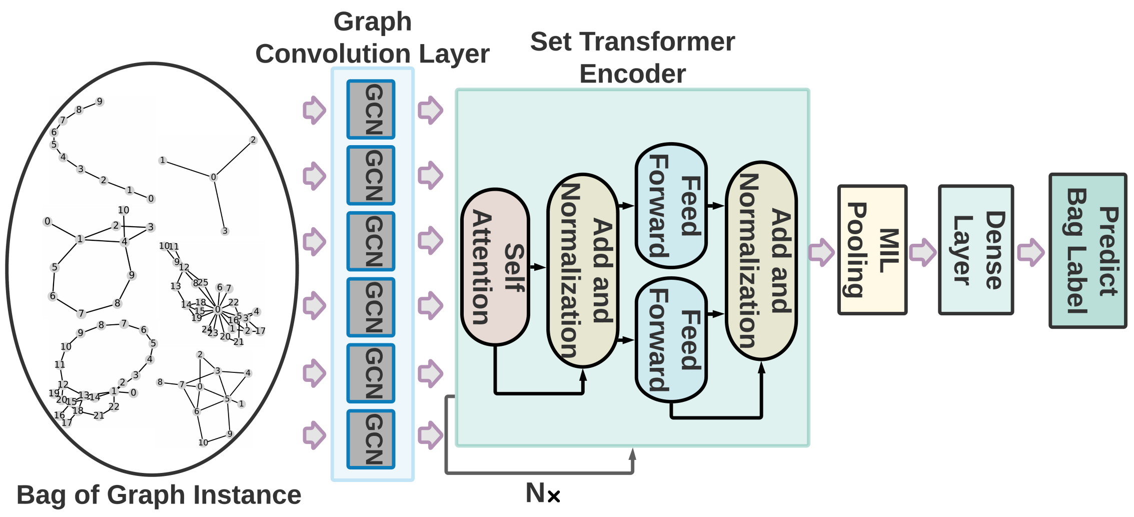

Figure 6.1: The architecture of Graph Instance Transformer (GIT).

tiple instance learning (MIL) pooling into Graph Instance Transformer (GIT), such that our model can process the bag of graph instances as input.

- By leveraging large amount of unlabeled bags of graph instances, we propose a semi-supervised multiple graph instance learning framework using contrastive self-supervised learning (CSSL), and integrate GIT with CSSL to form our final presentation of the proposed model, referred as Semi-supervised Graph Instance Transformer (SS-GIT)

6.2 Method

In this section, we illustrate our proposed Graph Instance Transformer (GIT), and the semi-supervised multiple graph instance learning framework. In generally, our proposed SS-GIT processes the input of bags of graph instances without having the knowledge of instance-level labels; the goal of our proposed method is to predict binary (positive/negative) bag-level labels. In SS-GIT, we integrate contrastive self-supervised learning with GIT, making we can train bag classifier semi-supervisedly.

Problem Formulation

Retrospective surveys (e.g., EMA) collected from patients are usually used as ground truth labels to provide supervision in the training process of machine learning models for early detection of mental illness. The retrospective surveys describe the existence of mental illness related symptoms for a past period of time (e.g., two weeks), and the mental health results can be linked to the digital behavioral markers that generated from mobile sensing data by using multi-modal sensors (e.g., accelerometer, gyroscope). We formalize mental health inference task as a Multiple Instance Learning (MIL) problem, predicting labels of unseen bags that contain multiple instances, assuming that positive bags contain at least one positive instance and negative bags contain only negative instances [140]. In the setting of mobile sensing data that have sparse labeled daily observations, this MIL formulation can predict the bag labels, without knowing the labels of single instances in bags. Due to the advantages of using graph representation learning for mobile sensing data [19], we transform mobile sensing signals to graphs for each daily mobile sensing observation; we form the bags of graph instances by aggregating multiday mobile sensing observations, since each EMA describe mental health status for past 12 days.

Graph Instance Transformer

Here we briefly introduce GIT. In general, there are three components in GIT: graph convolution layer, set transformer encoder, and MIL pooling, as shown in Fig. Figure 6.1. GIT can be summarized to conduct the following four steps when it is used to predict bag labels: 1) graph convolution layer produce graph embeddings based on the bag of graph instances as input for GIT; 2) the set transformer encoder learns graph instance embeddings from the bag of graph instances; 3) the permutation-invariant MIL pooling generate bag embeddings given the graph instance embeddings; 4) the dense layer predicts the bag labels based on the bag embeddings. The detailed design of GIT is demonstrated below.

Graph Convolution Layer, such as Graph Convolution Network (GCN) [93] and Graph

Isomorphic Network (GIN)[96], can generate numerical representations of graph structured inputs by encoding the geometric and complex interconnections of local sub-graph structures [6]. In this work, we use graph representations of mobile sensing data to capture diverse and complex human behavior dynamics [19]. In GIT, GCN is implemented to produce instance level graph embeddings based the information that are shared around node neighborhoods. Given a graph $\mathcal{G}(V, E, \mathbf{X})$, the adjacency matrix and degree matrix of $\mathcal{G}$ can be denoted as $\mathbf{A}$ and $\mathbf{D}$ respectively. And $\mathbf{X}$ is a node feature matrix. Then layer-wise propagation in a GCN can be expressed as

$$e = \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}\mathbf{W}). \tag{6.1}$$

We set each node in $\mathcal{G}$ as self-connected, and define $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. $\tilde{\mathbf{D}} = \mathbf{D} + \mathbf{I}$, and $\mathbf{W}$ is the weight matrix which will be learned in the training process. $\sigma(\bullet)$ is the activation function, such as $ReLU(x) = max(0, x)$.

Set Transformer Encoder: In Multiple Instance Learning (MIL) problems, we train a classifiers to recognize whether the input bags of instances as positive or negative, and the the order of the graph instances in each set does not impact the bag-level classification. In our situation, prediction results of mental illness recognition should not be influenced by the chronological order of graph instances, as not matter which day the patients show mental illness symptoms, the EMA can have the record that there exist some mental illness symptoms. Set Transformer is a permutation-invariant neural network model, and can capture interactions among instances in the input bags [51]. The permutation invariance property of set transformer enable us to generate the attention based instance embeddings from the outputs of graph convolution layer. Without considering the order of the elements in the bag input, Set Transformer encoders utilize attention mechanism [141] to assign higher attention score for the sub-graph structures that have higher contribution to the classification tasks. Multi-Head Self-Attention, by aggregating local representations, can enhance global representation power across graph instances. Specifically, the multi-head attention embedding consists of the concatenated output of multiple self-attention results from each GCN-graph embedding, and can be expressed as:

$$\begin{aligned} MultiHead(E) &= [H_1, H_2, ..., H_N]\mathbf{W^O}; \\ where\ H_i &= SelfAttention(E) \\ &= softmax(\frac{(E\mathbf{W_i^Q})(E\mathbf{W_i^K})^T}{\sqrt{d_k}}E\mathbf{W_i^V}). \end{aligned} \tag{6.2}$$

$E = \{e_1, e_2, ..., e_N\}$ is the set input of graph embeddings, and N is the size of the graph instance bag. W_i^Q, W_i^K, W_i^V are the learnable weight matrices for the ith attention head, and d_k is the dimension of the column space of W_i^K. $\mathbf{W^O}$ is another learnable linear projection matrix.

MIL Pooling can map the graph instance embeddings to another numerical spaces that contains bag embeddings. Based on the fundamental theorem of symmetric functions, any MIL pooling should satisfy the requirement of permutation-invariant symmetric function [50]. Popular MIL pooling includes max pooling, mean pooling, and attention based MIL pooling, such as MIL attention pooling [140]. In this study, we use gated attention MIL pooling, which applies gated attention mechanism to capture non-linearity and complex relations among the instances. The gated attention MIL pooling is expressed below:

$$a_n = \frac{exp\{\mathbf{W}_{atn}^T(tanh(\mathbf{V}z_n^T)) \bigodot sigm(\mathbf{U}z_n^T)\}}{\sum_j^N exp\{\mathbf{W}_{atn}^T(tanh(\mathbf{V}z_j^T)) \bigodot sigm(\mathbf{U}z_j^T)\}}, \tag{6.3}$$

where $\mathbf{W}_{atn}, \mathbf{U}, \mathbf{V}$ are learnable parameters, $\bigodot$ is an elementwise multiplication and $sigm(\bullet)$ is the sigmoid function.

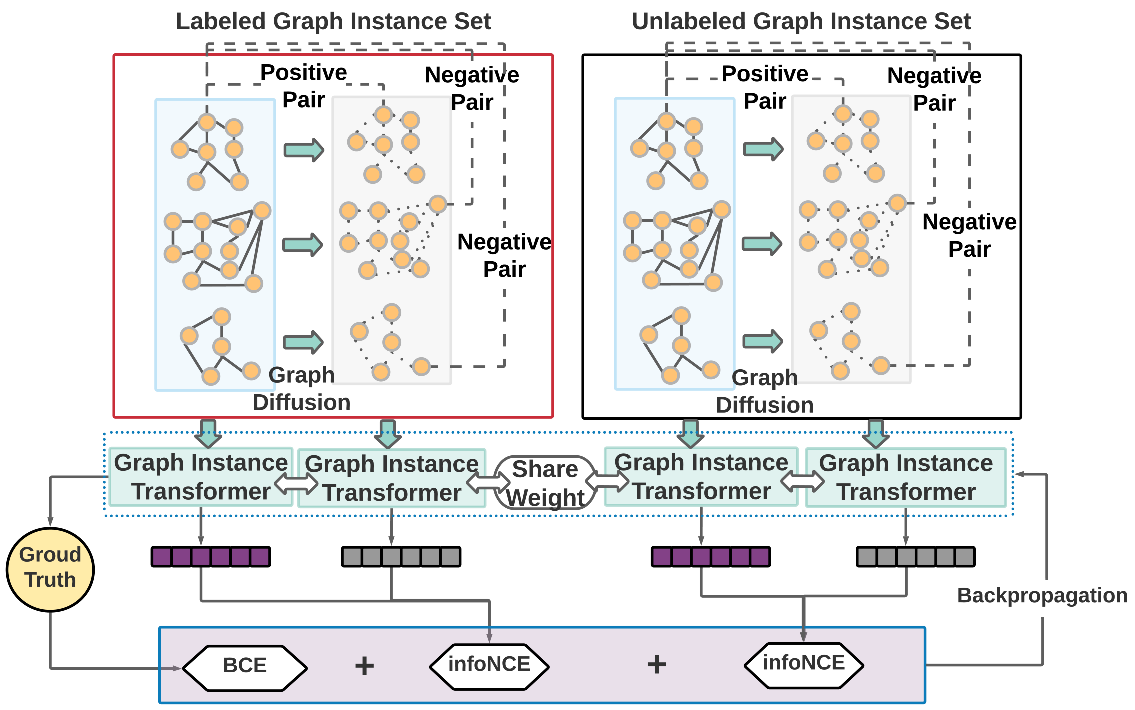

Figure 6.2: Framework of Semi-supervised Multiple Graph Instance Learning using Contrastive Self-supervised Learning.

Semi-supervised Multiple Graph Instance Learning

For the sparsity of annotated bags of graph instances, semi-supervised multiple graph instance learning (semi-MGIL) is designed to calculate contrastive loss by using contrastive self-supervised learning [142] and use the contrastive loss as regularization to alleviate overfitting in graph-instance bag classification. As shown in Fig. Figure 6.2, our proposed semi-MGIL firstly generates augmented graphs as contrastive counterparts to the original bag of graph instances for both labeled and unlabeled data. As the contrastive pairs are formed by the original and augmented graphs, we feed the contrastive pairs into GIT to produce predicted bag labels and bag embeddings for the supervised part and the self-supervised part respectively. We calculate Binary Cross Entropy (BCE) loss by comparing predicted bag labels and ground truth labels, and calculate contrastive losses, infoNCEs, between the bag embeddings generated from original graphs and augmented graphs. Finally, we use backpropagation to minimize the loss function (BCE+infoNCEs) and optimize the parameters in GIT. The detailed implementation of graph-based contrastive self-supervised learning and graph augmentation (graph diffusion[143]) are illustrated below:

In contrastive self-supervised learning (CSSL), by using data augmentation techniques, we generate similar (positive) counterpart for an original sample, and dissimilar (negative) counterpart from the contrastive sample, which is assumed to be different from the original one. Given the positive and negative pairs, CSSL targets to minimize the discrimination between the embedding of positive pairs and maximize discrimination between the embedding of negative pairs [142]. In our study, as we have input of bags of graph instances, we pair the original and augmented graph instance as positive, and the original and other augmented graph instances within the bag as negative, as shown in Fig. Figure 6.2, under the assumption that people can have different behavioral patterns from day to day.

There are two steps in the overall process of SS-GIT, the first is supervised learning, and the second is contrastive self-supervised learning. In supervised learning, we calculate BCE loss to generate supervision to train GIT. In the contrastive self-supervised learning part, we maximize the similarity between positive pairs of graph instance inputs, which are usually measured by using mutual information $\mathcal{I}(\mathbf{h}_i, \mathbf{h}_j)$[144], where $\mathbf{h}_i$ and $\mathbf{h}_j$ is a pair of embeddings. In this study, rather than maximizing the mutual information directly $\mathcal{I}(\bullet, \bullet)$, we maximize the lower bound of mutual information called infoNCE [144]. Thus, in our semi-MGIL, given a bag of graph instances $\mathcal{B}$ with size N, we minimize the infoNCE loss that can maximize the mutual information,

$$\mathcal{L}_{infoNCE} = -\frac{1}{N}\sum_{g \in \mathcal{B}}[log\frac{exp(\mathcal{D}(h_i, h_i^{'}))}{\sum_{g^{'} \in \mathcal{B} \setminus \{g_i\}} exp(\mathcal{D}(h_i, h_j^{'}))}], \tag{6.4}$$

where (h_i and $h_i^{'}$) is a positive pair of embeddings and (h_i, $h_j^{'}$) is a negative pair of embeddings for graph instance g_i. The final loss function for semi-MGIL can be expressed as

$$\mathcal{L}_{semi} = \mathcal{L}_{BCE}(y, \hat{y}) + \mathcal{L}_{infoNCE}(\mathcal{H}_l) + \mathcal{L}_{infoNCE}(\mathcal{H}_u) \tag{6.5}$$

, where $\mathcal{H}_l$ and $\mathcal{H}_u$ are labeled and unlabeled bags of graph instance representations respectively.

Graph Augmentation: the goal of graph augmentation in semi-MGIL given the graph strucutred data is to generate congruent views of the original graph input without supervision [145]. In general, there exist three different categories of graph augmentation techniques: first is Edge Perturbation, which can perturb topological connectivity by adding or deleting edges within graphs. The second type is Node Manipulation, including node insertion and deletion, changes local sub-graph structures of the original graphs. The third type of graph augmentation is Graph Diffusion, which can smooth out adjacency matrices over graphs, making the information can pass through higher-order neighborhood [146]. Intuitively, similar as Gaussian filter that are used in image processing to make images blurred, graph diffusion can make discrete adjacency matrices be continuously approximated to. Graph diffusion has been demonstrated to provide state-of-the-art results in graph classification and other graph related tasks. Since graphs generated from mobile sensing data can be sensitive to edge perturbation and node manipulation that can downgrade prediction performance [147], in this study, we apply graph diffusion to generate augmented graphs for semi-MGIL. The graph diffusion is formulated as below:

$$\mathbf{S} = \sum_{k=0}^{\infty} \theta_k \mathbf{T}^k, \tag{6.6}$$

where $\mathbf{T} = \mathbf{A}\mathbf{D}^{-1}$. And $\theta_k = \alpha(1-\alpha)^k$, where $\alpha \in (0, 1)$ is a customized hyperparameter.

6.3 Data and Experiments

Data Description, Collection and Preprocessing

The data that is used for evaluation of the performance of the proposed method is generated from a research project about mobile sensing for earlier illness diagnosis including influenza, mental disorders (anxiety and depression) and brain concussion. In totall, about 2,700 participants from 24 states in the U.S. were recruited and asked to install a mobile sensing app, ReadiSens. The time of the research project ranges from Feb 15th 2019 to Apr 30th 2020. In this study, our data include about 1,300 participants who submitted EMAs related their mental health status and who had at least 14 days worth of mobile sensing data records. The average age of the participants is $39.65(sd = 10.71)$. 66.12% of the participants are female, with White being 63.7%, African American 18.9%, Asian 9.6%, multiple races 3.8%, Hispanic 3.1%, and others 0.9%. Among the participants, 53% are full time workers, 19% are full time students, 15% are working part time, 6% have retired, 4% are care takers, and 3% are temporarily unemployed.

When people use ReadiSens, their mobility behavior, socil interactions physical activities and app usage are recorded by the embedded sensors, including GPS sensor, Bluetooth encounter, Wifi signal, app usage, accelerometer, gyroscope etc. GPS location data (e.g., latitude and longitude coordinates) are uploaded to AWS cloud every 30 mins; Bluetooth Encounters data every 15 mins; activity data when activities change; Wifi signal data every 15 mins. Ecological momentary assessments (EMAs) were delivered every 12 days to collect self-reported evaluation of participants' mental health status for the past 12 days. The mental health surveys used for the self-evaluation included GAD-2 for anxiety disorder [148] and PHQ-2 for depression [149]. Data were periodically uploaded to Amazon Web Services (AWS) Simple Storage Service (S3). Participants' data security and privacy were protected by data encryption and anonymization.

In the data preprocessing, as shown in Fig. Figure 6.3, people's daily behavior signatures are collected by the multimodal sensor. By using graph representation learning, we transform the multi-channel mobile sensory dat into multi-channel graphs, where each graph is regarded as on graph instance in the bag input. Since our EMAs summarize people's mental health status for past 12 days, we construct the bag of graph instance using

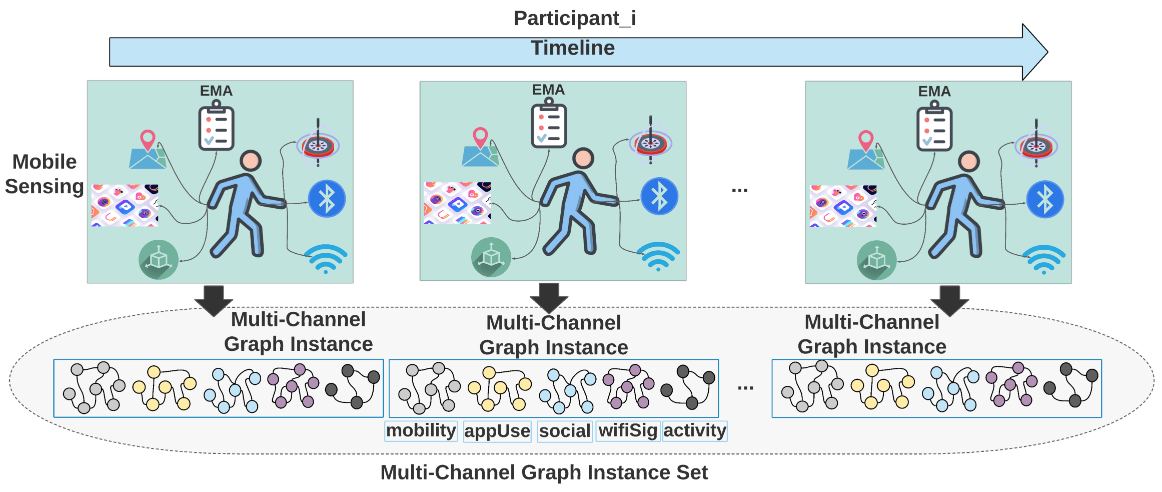
Participant_i
Timeline
Mobile Sensing
EMA
EMA
EMA
...
Multi-Channel Graph Instance
Multi-Channel Graph Instance
Multi-Channel Graph Instance
...
mobility
appUse
social
wifiSig
activity
Multi-Channel Graph Instance Set

Figure 6.3: Transform Multi-modal Mobile Sensing Data to Multi-Channel Graph Instance Set.

every 12 days mobile sensing data. For the ground truth information, self-reported GAD-2 and PHQ-2 results are used to provide quantitative evaluation about participants' mental health status. We use score of 3 points as cut-off, which are widely used threshold in mental health studies [150, 151, 152] for identifying possible positive cases in mental illness diagnosis. The positive label (GAD-2/PHQ-2 score ≥ 3) indicates that participants have mental disorders. Finally, we have 5,600 annotated bag of graph instance and 12,000 unannotated bag of graph instance for anxiety disorder. And we have 4,900 annotated bag of graph instance and 10,700 unannotated bag of graph instance for depression. The class distribution for the bag of graph instance is illustrated here: proportion of positive cases for anxiety was 19% and proportion of positive cases for depression was 15%. Since we have imbalanced data set, we conduct oversampling for the minor class to re-balance the training data. For the training and validation process, we separated the whole data set into 70% training data and 20% validation data. 10% of the data set are used for testing the models.

Baseline Model

We compare our proposed model to the following state-of-the-art baseline models:

GCN: graph convolutional network [93] is the classical graph neural network model and commonly serve as baseline model in graph classification problem [153].

GIN: graph isomorphic network [96] is the state-of-the-art GNN model which can alleviate graph isomorphism challenge in graph classification.

GRU: gated recurrent neural network [32] aims to solve the long-term dependency problem given sequential input, and has been widely studied in mobile sensing application to achieve competitive performance.

GCN-GRU: graph convolution-gated recurrent units is a temporal graph convolutional neural network, and has been applied in intelligent transportation which can outperform traditional deep learning models [154].

Deep Set: In the research area of permutation invariant neural network, Deep Set [50] is a benchmark deep architecture which can deal with sets as inputs.

Set Transformer: Set Transformer [51] is a attention-based permutation-invariant Transformer neural network.

Evaluation Metrics

Since the problem that we are solve fundamentally is a supervised graph classification problem in the bag-level. in this study, we use synthetic classification evaluation metrics including F1-score, ROC-AUC, and PR-AUC to compare the performance among our proposed method and the baseline models.

6.4 Results and Analysis

There are three different types of input format, aggregated, sequential, and instance set, in the evaluation part of comparing the proposed models with the baselines, as shown in TABLE Table 6.1. Here is the difference between these three different input type: the aggregated input refer to the embeddings of graph instances that are concatenated in each

Category	Input	Model	Anxiety			Depression		
			F1-score	ROC-AUC	PR-AUC	F1-score	ROC-AUC	PR-AUC
Baseline	Aggregated	GCN	0.714±0.036	0.808±0.045	0.721±0.043	0.729±0.036	0.796±0.047	0.765±0.031
		GIN	0.706±0.032	0.792±0.047	0.755±0.039	0.715±0.041	0.783±0.038	0.743±0.042
	Sequential	GRU	0.731±0.048	0.826±0.034	0.763±0.038	0.726±0.045	0.794±0.035	0.758±0.039
		GCN-GRU	0.753±0.044	0.817±0.037	0.784±0.039	0.738±0.033	0.809±0.032	0.779±0.040
	Instance Set	Deep Set	0.780±0.025	0.863±0.019	0.792±0.023	0.746±0.016	0.811±0.021	0.786±0.017
		Set Tfmer	0.801±0.016	0.879±0.011	0.813±0.018	0.752±0.027	0.818±0.015	0.795±0.024
Proposed	Instance Set	SS-GIT	**0.867±0.023**	**0.912±0.018**	**0.890±0.025**	**0.823±0.022**	**0.897±0.019**	**0.834±0.026**

Table 6.1: Result comparison between baseline models and our proposed model GIT using the Semi-supervised Deep Multiple Graph Instance Learning framework shown in Figure 6.2, referred as semi-supervised graph instance transformer (SS-GIT)

set input. The sequential input refer to the chronologically ordered sequence of graph instances. The instance set input refer to the unordered set of graph instances.

As shown in TABLE Table 6.1, we present the results of quantitative evaluation. From TABLE Table 6.1, we can observe the outperformance of the proposed SS-GIT over the selected baseline models both the anxiety recognition and depression recognition tasks. Specifically, our proposed model shows 8.8% higher F1-score, 6.7% higher ROC-AUC, and 7.2% higher PR-AUC than the best baseline model, Set Transformer. The superior performance demonstrated by SS-GIT can be attributed to the synthesises of graph neural network, set transformer and gated attention MIL pooling. In SS-GIT, the graph convolution layer has the advantage of producing representative embeddings of graph structured input that are transformed from raw mobile sensing data. However, in traditional Neural Network methods (Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN)) and handcrafted feature engineering, the complex human dynamics cannot be comprehensively represented. Additionally, set transformer encoder plays can essential role of capturing complex interactions between local sub-graph structures in neighboring graph instances from the same bag of graphs. What's more the gated attention MIL pooling can assign higher attention weight to the graph instances which have higher contributions to the prediction task than other graph instances, implying that this channel of human behavior is more indicative to the final mental illness detection. Another interesting observation is that the models with instance set input format present better performance than the other selected baseline models that use aggregated and sequential graph input. We can explain this observation from this perspective: the problem of MIL formulation enables the permutation-invariant models (e.g., Graph Instance Transformer) to generate most salient representations of positive instances which is helpful for recognition of positive bags. In mental health inference, given the EMAs that summarize patients' mental health status for a past 12 days, the order and the position of the days of having anxiety/depression symptoms should not impact the status of mental illness. Furthermore, no matter which day that patients feel anxiety/depression, the corresponding EMAs should show positive in their mental health illness diagnosis. For the aggregated graph instance input, which we do not consider about the temporal relationship between daily mobile sensing observation, using one single graph to describe 12 days human behaviors can make the positive mental illness behavior be overridden. This can be one of the reasons that Graph Neural Network model (e.g., GCN) by using aggregated show downgraded performance. For the sequential graph input, despite the existence of temporal correlation between human behaviors of continuous days, as we mentioned before, the order of the graph instance should not impact the reported results, and chronologizing graph instances could introduce long-term dependence issue which can make GRU less robust.

Interpretability Analysis

Since our optimal goal is to develop real machine learning systems that can detect mental illness symptoms and then use mobile sensing apps to realize the machine learning model that can be used in clinics, we provide interpretability analysis of the proposed SS-GIT to enhance the trustworthy and transparency of this model. In this interpretability analysis, we applied GNN based gradient-weighted CAM (GNN-Grad-CAM) [113] to visualize the

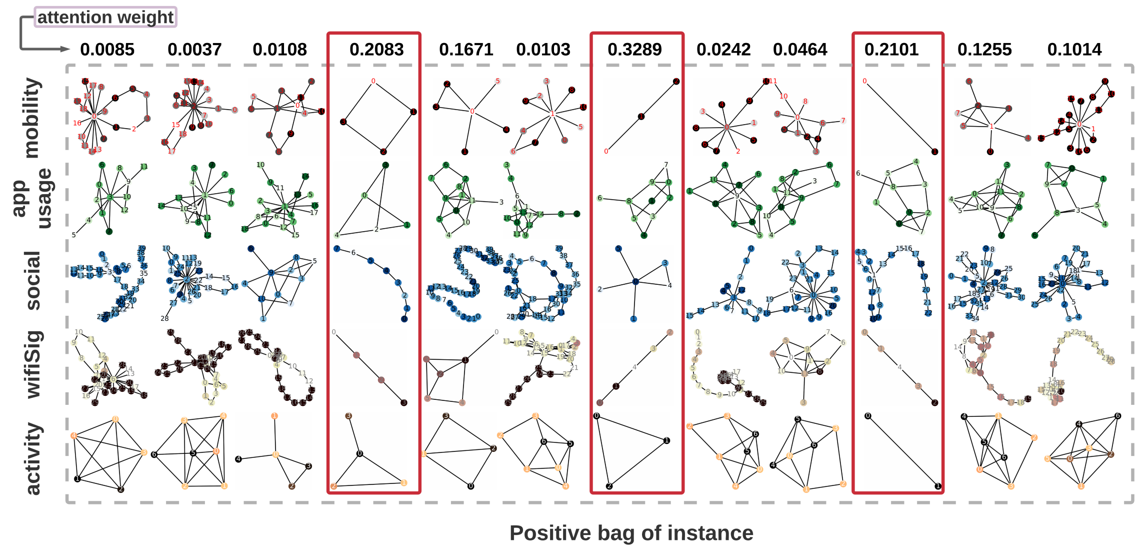

Figure 6.4: Case Study: node importance plot with attention weight for a positive bag of multi-channel graph instances.

node importance of nodes and global/local graph structures to illustrate how SS-GIT makes decision. In addition, since we apply attention mechanism in SS-GIT by assigning higher attention weight for the graph instances that can have higher contribution for mental illness recognition, we compare the contributions of graph instances to the bag embeddings by demonstrating the attention weights that are generated by MIL pooling layers. As shown in Fig Figure 6.4, we present the interpretability analysis of a positive bag of graph instance that are randomly selected from the data set. The multi-channel graph instances with red circles are assigned with higher attention weights than other bag of graph instances, implying that these bags of graph instances have higher contribution to SS-GIT and people can have mental illness symtoms for these days. We can observe that the multi-channel bag of graph instances that have higher attention-scores generally show less topological complexity (e.g., less cardinality and node degree) than the multi-channel graph instances that have lower attention-score. Complex bag of graph instances are indicative that people show diverse human behavior state changes and further imply that the people are physically and mentally well, vice versa. As our task is binary classification to detect the existence of mental illness symptoms, and at least one day of positive symptomatic behavior can make

Ablation	Anxiety	Depression
	F1-score (↑)	F1-score (↑)
GIT-without MIL Pooling	0.801 (-)	0.776 (-)
GIT	0.819 (0.018)	0.784 (0.019)
GIT+CSSL	0.852 (0.033)	0.811 (0.027)
GIT+semi-CSSL	0.867 (0.015)	0.823 (0.012)

Table 6.2: Result of ablation analysis.

the whole bag as positive, the multi-channel graph instances with higher attention weight can be potential positive instances that represent the human behaviors when people feel less engagement, depression or social anxious.

Ablation Study

In this section, we present the results of ablation study that is to investigate the impact of the components in SS-GIT. In this ablation study, we aim to evaluate the impact of multiple instance learning (MIL) pooling, Contrastive Self-supervised Learning (CSSL) and Semi-CSSL. As shown in TABLE Table 6.2, we present the results of ablation study. By comparing the performance of the proposed SS-GIT and the ablated models, we can observe that the module of CSSL can improve the prediction power by using contrastive learning to automatically generate self-supervision as regularization, as the GIT with CSSL shows the highest F1-score increase. What's more, the contribution of semi-supervised learning on the unlabeled data is less important than the contribution of MIL pooling. We would like to argue that the MIL pooling can contribute to generate more robust bag embeddings. Furthermore, semi-supervised learning for the unlabeled data demonstrate a relative small contribution to the prediction performance of the model. We can explain that this is because using CSSL for the labeled data can generate sufficient regularization in the training process of SS-GIT.

6.5 Summary

In this work, we propose a semi-supervised Graph Instance Transformer (SS-GIT) model for early mental illness recognition by using passively collected mobile sensing data. Instead of feeding all mobile sensing observations into a neural network model to predict mental illness, we transform the prediction task as a multiple instance learning (MIL) problem. Based on the MIL problem formulation, we generate bags of graph instances as input to the proposed model, which the chronological order of the original mobile sensing observations cannot impact the prediction results. Our proposed method can process the input that are sets of instance, implying that we don't have to label every element in the set input. In addition, given the challenges of label sparsity, GIT is integrated with contrastive self-supervised learning, such that the large amount of unlabeled data can be utilized to generate contrastive regularization to improve the robustness of the proposed model. As demonstrated in the results, our proposed method can outperform over the selected baseline models in mobile sensing tasks about mental illness recognition. As our proposed model is also generalizable for other different mobile sensing collection, implying that we can propagate our model for other machine learning tasks in mobile sensing, such as early identification of other chronic mental illness. In real world situation, there can be low engagement with mobile sensing apps, making sparsity in ground truth collection, as most of mental illness early-stage diagnosis are replying on self-reported retrospective surveys. The proposed method enable us to train fairly competitive models semi-supervisedly under weak supervision.

Although our proposed method shows superior performance, our approach is also subject to the following limitations. First of all, ground truth labels can be biased, weakening the faithfulness of the mental health study and potentially downgrading the reliability of the predictive modeling. Different people can have different standards about their daily mental health status, thus self-reported measure usually cannot be consistent and standardized as the annotations in other research field, such as image recognition. The self-reported retrospective surveys can introduce recall biases and avoidance coping (e.g. behaviors that avoid addressing psychiatric problems), which can degrade the data quality. Secondly, heterogeneity in human behaviors can negative impact the generalizability of our proposed model. Additionally, behavior nonstationarity and habit evolution over time also limit the capacity of our model.

In the future, it's worthy to investigate the personalized machine learning by leveraging transfer learning. We also plan to expand our research to the study of behavioral symptomatology for psychiatric disorder, and explore graph theory to enhance the graph representation of nonstationary and evolving human behaviors. Furthermore, we will work with psychiatrists and clinicians to improve the quality of survey responses and design new algorithm of recalibration to counter the inconsistency and biases in ground truth collection.

Chapter 7

Discussion

7.1 Broader Impact

First and foremost, we propose a general predictive model of salivary cortisol levels by using mobile sensors. We use Graph Representation Learning (GRL) methods to automatically generate human state embeddings to make cortisol level predictions. To our best knowledge, this is the first work that applies GRL and mobile sensing to predict the biomarker. Secondly, we propose an end-to-end multi-channel graph neural network framework to model human behaviors by leveraging multi-modal mobile sensing data. The goal is to automatically extract high-level features representing the dynamic interactions between human states. To our best knowledge, this is the first work that applies graph neural networks to infer human health states using mobile sensing data. What's more, we propose a semi-supervised permutation invariant neural network model, referred to as the Semi-supervised Graph Instance Transformer (SS-GIT), for early stage mental illness diagnosis under weak supervision. Last but not least, we propose an incremental semi-supervised federated learning (IFL) framework, FedMobile, to train neural network models decentralizedly and semi-supervisedly by integrating Knowledge Distillation and consistency regularization. Our incremental framework enables us to update existing models only using newly available data in an online fashion. We apply graph representations to encode complex human behaviors by transforming multi-modal raw sensor data to graphs and propose a spatio-temporal graph neural network, referred to as the Hierarchical GCN-LSMT, to fit the multi-channel dynamic graph inputs with spatial-hierarchy in mobile sensing.

By using mobile sensing techniques, the end-to-end multi-channel GNN framework can be easily generalized and applied to predict other health-related outcomes, such as mental disorders. GIT is a general-permutation invariant graph neural network that can solve the problem of graph multiple instance learning. For example, given bags of graph representations of social media information (e.g., tweets), we can use GIT to predict the risk of suicide. And in security applications, given a bag of graph representations of human skeletons in a crowd of people, we can use GIT for crowd abnormal behavior detection. FedMobile serves as a general federated learning algorithm, contributing to the field of privacy-preserving machine learning. FedMobile has the potential to be extended to other application scenarios, such as distributed learning of image recognition. Overall, our proposed deep graph learning methods show a state-of-the-art method to model human behaviors from a topological perspective. Our proposed models also have numerous practical implications, as illustrated in the following.

Clinical Implication: Currently, the primary methods for measuring cortisol are through serum and saliva samples, which are inconvenient to collect and costly to analyze. The inconvenience of having biospecimens collected is pronounced in adults with pancreatic cancer given the impact of their disease on their health, daily functioning, and overall quality of life. Biospecimen collection is also impractical as a means of monitoring cortisol in real-time given the time needed to complete assays. Tracking cortisol levels through our proposed predictive modeling has the ability to advance our understanding of the trajectory of tumor growth and the potential role it plays in response to anticancer treatment.

Our proposed incremental federated learning (IFL) can alleviate the financial burdens from continuously purchasing and restoring resources to stock newly collected data since

IFL can update, transfer, and adapt new knowledge by updating the apprentice model without retraining all historical data. And our proposed semi-supervised learning framework in IFL and GIT can be an economical training strategy in mobile sensing studies since we don't have to pay extra money to attract participants to actively annotate observations.

Implications for Public Health: At present, the Centers for Disease Control and Prevention (CDC) relies on collaborative efforts from local clinics and state agencies to report influenza virus infection cases [77]. However, this case report system can be unpunctual and prone to underestimate. The limitations of the traditional public health reporting systems and syndrome surveillance can be overcome by deploying our proposed mobile sensing solution for early-stage influenza-like symptom recognition in practice. Our proposed mobile sensing solution can be used to develop intelligent influenza surveillance systems to continuously monitor influenza activity, automatically detect early ILI among the population, and accurately predict influenza outbreaks. Furthermore, interventions can be delivered to mobile users to change individuals' behaviors, such as self-isolation, if influenza-like symptoms are detected. The interventions can be scaled up to a population level to combat the transmission of the influenza virus.

7.2 Limitation

First and foremost, in this DGL framework, we transfer mobile sensing data into undirectional and unweighted graphs. However, edges in each graph can also have rich behavioral information. For example, in the GPS trajectory graph, the edges can be assigned features such as distance between two places. This edge information cannot be handled with the consolidated GNN models. Secondly, using the DGL framework requires the setup of a large number of hyperparameters and is involved with many parameters to build a model. However, when modeling human behaviors, there is usually a small number of samples. The DGL learned model cannot be reliable with small training samples, making this framework impractical to realize. What's more, missing value and missing sensor problems are hard to alleviate in the DGL framework. In handcrafted feature engineering, the missing value can be imputed by using other features. However, in the DGL framework, missing values are ignored, resulting in an inaccurate representation of human behaviors. And the missing sensor problem, which is also a challenging problem in handcrafted feature engineering, can lead to missing graph input in the GNN models. Last but not least, this DGL framework assumes that each type of behavior has the same impact on the health-related problem. However, different health issues can cause different levels of change in human behavior. For example, depression can cause diminished social interactions but may not lead to a large decrease in mobility because people may still go to grocery stores or workplaces to fulfill their daily responsibilities. Some topologically similar graphs can not be differentiated by using GNN. [155]. Even though some graphs have different structures with different nodes and edge connections, these graphs can be isomorphic. When we apply GRL to human behavior representation, the prediction result of GNN will be the same even though people have different travel trajectories but visit the same places. GNNs are sensitive to noise in graph signals.[156]. Small changes in node features can make the prediction different. To overcome this challenge, GNN needs a large number of samples to train a more generalized model. In mobile sensing, data is hard and expensive to collect. It's changeling to train GNN with a small number of samples with standard data augmentation methods. The interpretability analysis of GNNs in mobile health can be biased. Even though GNN-CAM are widely used to provide interpretation of input graph structure in GNNs, GNN-CAM can only provide post-hoc node importance plot. In mobile sensing study, users can have various behaviors even if they report that they have the same health states. Thus, when we analyze the interpretability of graph inputs of mobile sensing data, generalised knowledge of connecting human behaviors with relevant health outcomes can be less persuasive.

7.3 Future Works

In incorporating edge features in DGL, we can apply EGNN [157] which aggregates node feature and edge feature together to feed into Graph convolutions operation. Standard data augmentation method [158, 159] for image and text data are not applicable in GNN. Since the input of GNN is generated from non-euclidean spaces [9]. Few methods have been proposed for data augmentation in graph data [160]; however, all of the methods have not been consolidated. There are several works in the general IoT field that discuss missing sensor data imputation for the missing sensor problem.Liu et al. proposed a graph-based sensor imputation method by using a message passing mechanism in a graph neural network to construct a spatial sensor topological graph. [161]. The correlation between physically connected sensors is exploited to reconstruct the features from missing sensors. In our case, we have not only the missing node attribute but also the missing graphs. It is a challenging task to predict a graph-structured output given related graph inputs. To leverage potentially different contributions from different human behaviors to predict health outcomes, we can apply attention mechanisms [162] to let our model focus on the important behaviors. Attention methods are applied originally in Natural language processing by assigning heavier weight to the significant words to predict performance better. Thus we can adapt this attention method in our graph-based human behavior modeling by assigning more weights to the latent representations from more significant behaviors.

One of the critical requirements for the deployment of neural network models in mHealth is reliability, which aims to guarantee their functionality in real-world situations. Unsuccessful mobile communication, which can be caused by sensor malfunction, Internet interruption, and data corruption, can negatively impact the effectiveness of mHealth applications with inaccurate or incorrect decision-making [163]. Sensor and network failures, for example, could cause healthcare professionals to miss the best time to treat patients, which could cause them to have severe physical or mental injuries. Thus, in spite of the superior performance that has been achieved by using GNNs, how failures can be prevented and detected and how the adverse exceptions can be addressed is what people are concerned about in mHealth implementation. Additionally, in real-world deployment of machine learning for health, there is a trade-off between using handcrafted features and deep features generated from GNNs. Handcrafted features have its advantage in easy interpretation but cannot provide granular interpretation. GNNs have advantage in providing detailed interpretation, but GNNs are still limited in providing generalisedable knowledge for clinical application. Developing reliable neural network models is a promising research direction to realize large-scale mHealth deployments. Liu et al. [161] propose a novel algorithm trying to alleviate the sensor failure issue: by leveraging the message passing mechanism in GNNs, the proposed algorithm can use the information passed from available sensors to reconstruct the features of missing sensors. However, since mobile sensing systems always have a large number of sensors, transmitting all sensor readings to the cloud for centralized data processing, prediction, and control results in significant delays due to communication network congestion and computational overload. [164]. Therefore, efficient and effective edge computing can be investigated to process the data at the individual sensor level or local sensor network areas. How to implement GNNs on edge computing systems is an exciting area of study to enhance the reliability of GNNs in healthcare applications. Last but not least, we will continue to work on to develop clinical-level interpretable GNNs for mobile health, such that we can better interpret human behaviors and link them with related health outcomes.

Mobile sensing systems are laying the groundwork to link every object in our daily life together, transforming the world. Things will be able to connect and interact with one another and with their surrounding environment, aiming to improve people's quality of life. In the process of sensing the environment through the connected sensors, information can be communicated with other mobile devices. Current strategies to store the sensory data include uploading the data to a central server (cloud serving) for future use. Despite the benefits of cloud-based data storage, centralized mobile sensing systems may face significant challenges. For instance, unencrypted server data is vulnerable to hacking and may result in the disclosure of critical information. For instance, during the data collection in mobile sensing systems, GPS trajectories may expose the location of mobile users, which can be used to infer additional sensitive personal information such as race, gender, physical activity, social relationships, and health conditions [116]. There are several privacy preservation techniques that can be utilized to enhance privacy protection in mobile sensing systems, such as anonymization, federated learning, and differential privacy. For example, the k-anonymity method and variations have been developed to anonymize the identity of data items' sources [118]. And differential privacy can minimize the disclosure of sensitive information by introducing perturbations that follow a specified noise distribution while preserving the value of the data [119]. Federated learning (FL) is another prospective technique for privacy preservation where sensor nodes can collaboratively contribute knowledge to the global learning objective without transmitting data samples to a central server or exchanging data across sensor nodes. However, there are disadvantages to anonymization techniques and differential privacy: Anonymization exposes sensitive characteristics to inference attacks, while differential privacy reduces the usefulness of data and diminishes the strength of learnt models [120]. Federated GNNs, which retain the predictive potential of GNNs while also protecting sensitive data, is an additional research path worth investigating.

The interconnected things in sensing rich environment introduce multiple security challenges, as the vast majority of Internet technologies and communication protocols were not created in mind [165]. With the advancement of sensor technology and widespread use of mHealth, consumers are concerned about their personal information being exposed to hidden cyber threats and technological crime, as well as other security problems that might constrain wider applications. Massive amounts of data are collected and managed by mobile sensing systems to support the services in healthcare. Attackers and adversaries, such as cybercriminals, hackers, and hacktivists, want to profit from this priceless information. When smart devices are hacked, credit card numbers, bank account PINs, and other personally identifiable information may be exposed to the public, resulting in financial loss for users of mHealth services. Sensor nodes and other sensing components that are connected through the Internet are also vulnerable to intelligent malware, which is designed to spread from device to device and intrude and interfere in users' everyday lives. What's more, hacktivists or illegitimate political agitators can exploit and manipulate connected smart devices to spark protests against sovereign governments [165]. There are already established security protection mechanisms in use, such as encryption, authentication, and access control. However these pred-defined mechanisms are incapable of adapting to new types of attacks and intelligent adversaries adequately safeguard IoT networks [164]. Machine learning models (e.g., neural networks) have been extensively investigated to learn normal and abnormal patterns in the interactions of smart devices. From a data-driven approach, the information relating to the robust operations of mobile sensing systems can be collected and utilized to profile normal interaction behaviors of sensor nodes; hence, malicious activities can be detected if the collected data shows an anomalous distribution. Existing methods based on traditional machine learning and neural network models, however, are incapable of capturing topological information and multi-hop interactions between sensor nodes.There is an emerging trend of applying GNNs to adversarial intrusion detection: for example, Lo et al. [166] propose GNN-based network intrusion detection system, which can capture both the edge features of a IoT network and the topological features. Originating game theory, adversarial learning and GNNs have potential to be integrated in the future to learn malicious activities with economic consideration.

Massive data production rates in the era of Big Data and the Internet of Things (IoT) continuously increase the demand for massive data processing, storage, and transmission [167]. However, smart devices are resource-constrained devices, with limited memory, computation and energy [168]. mobile sensing systems that are limited in terms of energy and power must not only provide high performance capabilities, but also serve as a platform for automation and intelligence. From handwritten digit recognition to autonomous driving, deep learning techniques (e.g., CNNs, RNNs) demonstrate groundbreaking success in achieving human-level recognition capability [169]. This superior performance has resulted in a huge increase in the use of DNN models in various applications. However, the massive computational, energy, and storage requirements of DNN models make their implementation on resource-constrained smart devices prohibitively expensive [170]. Current solutions to alleviate the issues of high demanding computational and energy resource to implement DNNs in smart devices include: 1) cloud computing; 2) developing edge computing GPUs. However, cloud computing suffers from high wireless energy overhead and has unattainable performance in weak network connectivity. GPUs in mobile devices may deplete significant amounts of mobile battery capacity [171]. Different from the infrastructure improvement solution, DNN compression techniques have been proposed for efficient storage and computation consumption with minimal accuracy compromise [172]. For example, knowledge distillation enables one to train a student model by learning from a teacher model, and the student model acts as a compressed version of the teacher model with similar prediction power [173]. The research of using knowledge distillation on graph neural networks have been investigated in these studies [174, 175, 176], while not extensively studied in the filed of smart health. In the future, there are opportunities to research how to efficiently integrate GNNs in smart devices by using knowledge distillation techniques.

Chapter 8

Conclusion

The use of mobile sensing systems has exploded in popularity over the past decade, owing to their ability to monitor human behaviors passively and unobtrusively across a plethora of domains of people-centered applications. In the sensor-rich environment, conventional deep learning techniques have demonstrated their capacity to process complex multi-modal sensory data and perform sophisticated detection and recognition tasks. By considering human behavior dynamics and sensor topology, deep graph learning methods can capture the complex relationships and interdependency within human behavior dynamics and demonstrate state-of-the-art results in multiple mHealth tasks. In this thesis, we present a set of designs for graph-based learning systems to address the limitations of conventional neural network models in the mobile sensing field. By using the mobile sensing data collected in the wild, we demonstrate that graph-based learning systems can uniformly outperform the selected baseline models, including traditional machine learning models and general neural network models, in different mHealth problems. We also provide a discussion about future research directions and real-world implementation of the proposed learning systems.

REFERENCES

[1] M. Boukhechba and L. E. Barnes, "Swear: Sensing using wearables. generalized human crowdsensing on smartwatches," in *International Conference on Applied Human Factors and Ergonomics*, Springer, 2020, pp. 510–516.

[2] L. Cai *et al.*, "An integrated framework for using mobile sensing to understand response to mobile interventions among breast cancer patients," *Smart Health*, vol. 15, p. 100 086, 2020.

[3] M. Boukhechba, A. R. Daros, K. Fua, P. I. Chow, B. A. Teachman, and L. E. Barnes, "Demonicsalmon: Monitoring mental health and social interactions of college students using smartphones," *Smart Health*, vol. 9, pp. 192–203, 2018.

[4] M. Boukhechba *et al.*, "Contextual analysis to understand compliance with smartphone-based ecological momentary assessment," in *Proceedings of the 12th EAI International Conference on Pervasive Computing Technologies for Healthcare*, 2018, pp. 232–238.

[5] M. Boukhechba, L. Cai, C. Wu, and L. E. Barnes, "Actippg: Using deep neural networks for activity recognition from wrist-worn photoplethysmography (ppg) sensors," *Smart Health*, vol. 14, p. 100 082, 2019.

[6] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 4–24, 2020.

[7] Y. Rong *et al.*, "Deep graph learning: Foundations, advances and applications," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 3555–3556.

[8] F. Chen, Y.-C. Wang, B. Wang, and C.-C. J. Kuo, "Graph representation learning: A survey," *APSIPA Transactions on Signal and Information Processing*, vol. 9, 2020.

[9] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[10] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *arXiv preprint arXiv:1709.05584*, 2017.

[11] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.

[12] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," *arXiv preprint arXiv:1801.07455*, 2018.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] Z. Li *et al.*, "A hybrid deep learning approach with gcn and lstm for traffic flow prediction," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 1929–1933.

[15] Y. Yu, T. Xia, H. Wang, J. Feng, and Y. Li, "Semantic-aware spatio-temporal app usage representation via graph convolutional network," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4, no. 3, pp. 1–24, 2020.

[16] D. Wang *et al.*, "Calendar graph neural networks for modeling time structures in spatiotemporal user behaviors," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 2581–2589.

[17] X. Chen, Y. Wang, J. He, S. Pan, Y. Li, and P. Zhang, "Cap: Context-aware app usage prediction with heterogeneous graph embedding," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 1, pp. 1–25, 2019.

[18] G. Dong *et al.*, "Using graph representation learning to predict salivary cortisol levels in pancreatic cancer patients," *Journal of Healthcare Informatics Research*, pp. 1–19, 2021.

[19] G. Dong, L. Cai, D. Datta, S. Kumar, L. E. Barnes, and M. Boukhechba, "Influenza-like symptom recognition using mobile sensing and graph neural networks," in *Proceedings of the Conference on Health, Inference, and Learning*, 2021, pp. 291–300.

[20] G. Dong, M. Tang, L. Cai, L. E. Barnes, and M. Boukhechba, "Semi-supervised graph instance transformer for mental health inference," in *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, IEEE, 2021, pp. 1221–1228.

[21] K. Liu, Y. Li, N. Xu, and P. Natarajan, "Learn to combine modalities in multimodal deep learning," *arXiv preprint arXiv:1805.11730*, 2018.

[22] S. Chakraborty, S. Aich, M.-i. Joo, M. Sain, and H.-C. Kim, "A multichannel convolutional neural network architecture for the detection of the state of mind using physiological signals from wearable devices," *Journal of healthcare engineering*, vol. 2019, 2019.

[23] O. Kampman, E. J. Barezi, D. Bertero, and P. Fung, "Investigating audio, visual, and text fusion methods for end-to-end automatic personality prediction," *arXiv preprint arXiv:1805.00705*, 2018.

[24] M. S. Salekin, G. Zamzmi, D. Goldgof, R. Kasturi, T. Ho, and Y. Sun, "Multi-channel neural network for assessing neonatal pain from videos," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, IEEE, 2019, pp. 1551–1556.

[25] W. Fan *et al.*, "Graph neural networks for social recommendation," in *The World Wide Web Conference*, ser. WWW '19, San Francisco, CA, USA: Association for Computing Machinery, 2019, pp. 417–426, ISBN: 9781450366748.

[26] W. Torng and R. B. Altman, "Graph convolutional neural networks for predicting drug-target interactions," *Journal of Chemical Information and Modeling*, vol. 59, no. 10, pp. 4131–4149, 2019.

[27] D. Neil, J. Briody, A. Lacoste, A. Sim, P. Creed, and A. Saffari, "Interpretable graph convolutional neural networks for inference on noisy knowledge graphs," *arXiv preprint arXiv:1812.00279*, 2018.

[28] X. Wang *et al.*, "Traffic flow prediction via spatial temporal graph neural network," in *Proceedings of The Web Conference 2020*, ser. WWW '20, Taipei, Taiwan: Association for Computing Machinery, 2020, pp. 1082–1092, ISBN: 9781450370233.

[29] N. Wu, X. W. Zhao, J. Wang, and D. Pan, "Learning effective road network representation with hierarchical graph neural networks," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '20, Virtual Event, CA, USA: Association for Computing Machinery, 2020, pp. 6–14, ISBN: 9781450379984.

[30] J. Zhou *et al.*, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[31] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," *arXiv preprint arXiv:1511.05493*, 2015.

[32] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[33] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model cnns," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5115–5124.

[34] A. Micheli, "Neural network for graphs: A contextual constructive approach," *IEEE Transactions on Neural Networks*, vol. 20, no. 3, pp. 498–511, 2009.

[35] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," *arXiv preprint arXiv:1709.04875*, 2017.

[36] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv preprint arXiv:1611.07308*, 2016.

[37] C. Yang, P. Zhuang, W. Shi, A. Luu, and P. Li, "Conditional structure generation through graph variational generative adversarial nets," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[38] M. Tang, C. Yang, and P. Li, "Graph auto-encoder via neighborhood wasserstein reconstruction," in *International Conference on Learning Representations*, 2022.

[39] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang, "Adversarially regularized graph autoencoder for graph embedding," *arXiv preprint arXiv:1802.04407*, 2018.

[40] H. Shi, H. Fan, and J. T. Kwok, "Effective decoding in graph auto-encoder using triadic closure," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 906–913.

[41] B. Malle, P. Kieseberg, E. Weippl, and A. Holzinger, "The right to be forgotten: Towards machine learning on perturbed knowledge bases," in *International Conference on Availability, Reliability, and Security*, Springer, 2016, pp. 251–266.

[42] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[43] A. Hard *et al.*, "Federated learning for mobile keyboard prediction," *arXiv preprint arXiv:1811.03604*, 2018.

[44] X. Hao, W. Ren, R. Xiong, X. Zheng, T. Zhu, and N. N. Xiong, "Fair and autonomous sharing of federate learning models in mobile internet of things," *arXiv preprint arXiv:2007.10650*, 2020.

[45] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "Fedhealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, 2020.

[46] A. Vaid *et al.*, "Federated learning of electronic health records to improve mortality prediction in hospitalized patients with covid-19: Machine learning approach," *JMIR medical informatics*, vol. 9, no. 1, e24207, 2021.

[47] Y. Chen, J. Bi, and J. Z. Wang, "Miles: Multiple-instance learning via embedded instance selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 1931–1947, 2006.

[48] Z.-H. Zhou, Y.-Y. Sun, and Y.-F. Li, "Multi-instance learning by treating instances as non-iid samples," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 1249–1256.

[49] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 8, pp. 1619–1632, 2010.

[50] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. Salakhutdinov, and A. Smola, "Deep sets," *arXiv preprint arXiv:1703.06114*, 2017.

[51] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh, "Set transformer: A framework for attention-based permutation-invariant neural networks," in *International Conference on Machine Learning*, PMLR, 2019, pp. 3744–3753.

[52] P. Rawla, T. Sunkara, and V. Gaduputi, "Epidemiology of pancreatic cancer: Global trends, etiology and risk factors," *World journal of oncology*, vol. 10, no. 1, p. 10, 2019.

[53] W. Wu *et al.*, "Rising trends in pancreatic cancer incidence and mortality in 2000–2014," *Clinical epidemiology*, vol. 10, p. 789, 2018.

[54] S. E. Sephton, R. M. Sapolsky, H. C. Kraemer, and D. Spiegel, "Diurnal cortisol rhythm as a predictor of breast cancer survival," *Journal of the National Cancer Institute*, vol. 92, no. 12, pp. 994–1000, 2000.

[55] E. K. Adam, M. E. Quinn, R. Tavernier, M. T. McQuillan, K. A. Dahlke, and K. E. Gilbert, "Diurnal cortisol slopes and mental and physical health outcomes: A systematic review and meta-analysis," *Psychoneuroendocrinology*, vol. 83, pp. 25–41, 2017.

[56] B. Martens and Z. Drebert, "Glucocorticoid-mediated effects on angiogenesis in solid tumors," *The Journal of Steroid Biochemistry and Molecular Biology*, vol. 188, pp. 147–155, 2019.

[57] D. C. Mohr, M. Zhang, and S. M. Schueller, "Personal sensing: Understanding mental health using ubiquitous sensors and machine learning," *Annual review of clinical psychology*, vol. 13, pp. 23–47, 2017.

[58] W. Schlotz, K. Hammerfald, U. Ehlert, and J. Gaab, "Individual differences in the cortisol response to stress in young healthy men: Testing the roles of perceived stress reactivity and threat appraisal using multiphase latent growth curve modeling," *Biological psychology*, vol. 87, no. 2, pp. 257–264, 2011.

[59] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, 2014, pp. 1188–1196.

[60] B. Rozemberczki and R. Sarkar, "Characteristic functions on graphs: Birds of a feather, from statistical descriptors to parametric models," *arXiv preprint arXiv:2005.07959*, 2020.

[61] F. Gao, G. Wolf, and M. Hirn, "Geometric scattering for graph data analysis," in *International Conference on Machine Learning*, 2019, pp. 2122–2131.

[62] A. Tsitsulin, D. Mottin, P. Karras, A. Bronstein, and E. Müller, "Netlsd: Hearing the shape of a graph," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2347–2356.

[63] B. Li and D. Pi, "Network representation learning: A systematic literature review," *Neural Computing and Applications*, pp. 1–33, 2020.

[64] G. Hamerly and C. Elkan, "Learning the k in k-means," in *Advances in neural information processing systems*, 2004, pp. 281–288.

[65] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "Graph2vec: Learning distributed representations of graphs," in *Proceedings of the 13th International Workshop on Mining and Learning with Graphs (MLG)*, 2017.

[66] C. H. Bastien, A. Vallières, and C. M. Morin, "Validation of the insomnia severity index as an outcome measure for insomnia research," *Sleep medicine*, vol. 2, no. 4, pp. 297–307, 2001.

[67] J. A. Steeves *et al.*, "Ability of thigh-worn actigraph and activpal monitors to classify posture and motion," *Medicine and science in sports and exercise*, vol. 47, no. 5, p. 952, 2015.

[68] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[69] A. E. Hoerl and R. W. Kennard, “Ridge regression: Biased estimation for nonorthogonal problems,” *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[70] H. Drucker, C. J. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, “Support vector regression machines,” in *Advances in neural information processing systems*, 1997, pp. 155–161.

[71] A. Liaw, M. Wiener, *et al.*, “Classification and regression by randomforest,” *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[72] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system,” in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[73] F. Murtagh, “Multilayer perceptrons for classification and regression,” *Neurocomputing*, vol. 2, no. 5-6, pp. 183–197, 1991.

[74] M. Yamin, “Counting the cost of covid-19,” *International Journal of Information Technology*, pp. 1–7, 2020.

[75] K. van Barneveld *et al.*, “The covid-19 pandemic: Lessons on building more equal and sustainable societies,” *The Economic and Labour Relations Review*, vol. 31, no. 2, pp. 133–157, 2020.

[76] D. Blumenthal, E. J. Fowler, M. Abrams, and S. R. Collins, *Covid-19—implications for the health care system*, 2020.

[77] F. Ma, S. Zhong, J. Gao, and L. Bian, “Influenza-like symptom prediction by analyzing self-reported health status and human mobility behaviors,” in *Proceedings of the 10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, 2019, pp. 233–242.

[78] H. Hu, H. Wang, F. Wang, D. Langley, A. Avram, and M. Liu, “Prediction of influenza-like illness based on the improved artificial tree algorithm and artificial neural network,” *Scientific reports*, vol. 8, no. 1, pp. 1–8, 2018.

[79] F. Al Hossain, A. A. Lover, G. A. Corey, N. G. Reich, and T. Rahman, “Flusense: A contactless syndromic surveillance platform for influenza-like illness in hospital waiting areas,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4, no. 1, pp. 1–28, 2020.

[80] S. Volkova, E. Ayton, K. Porterfield, and C. D. Corley, “Forecasting influenza-like illness dynamics for military populations using neural networks and social media,” *PloS one*, vol. 12, no. 12, e0188941, 2017.

[81] C.-T. Yang *et al.*, “Influenza-like illness prediction using a long short-term memory deep learning model with multiple open data sources,” *The Journal of Supercomputing*, pp. 1–27, 2020.

[82] D. R. Olson, K. J. Konty, M. Paladini, C. Viboud, and L. Simonsen, “Reassessing google flu trends data for detection of seasonal and pandemic influenza: A comparative epidemiological study at three geographic scales,” *PLoS Comput Biol*, vol. 9, no. 10, e1003256, 2013.

[83] M. Boukhechba, A. N. Baglione, and L. E. Barnes, “Leveraging mobile sensing and machine learning for personalized mental health care,” *Ergonomics in Design*, vol. 28, no. 4, pp. 18–23, 2020.

[84] J. Gong *et al.*, “Understanding behavioral dynamics of social anxiety among college students through smartphone sensors,” *Information Fusion*, vol. 49, pp. 57–68, 2019.

[85] M. Boukhechba, Y. Huang, P. Chow, K. Fua, B. A. Teachman, and L. E. Barnes, “Monitoring social anxiety from mobility and communication patterns,” in *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers*, 2017, pp. 749–753.

[86] M. K. Ameko *et al.*, “Cluster-based approach to improve affect recognition from passively sensed data,” in *2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, IEEE, 2018, pp. 434–437.

[87] L. Cai *et al.*, “State affect recognition using smartphone sensing data,” in *Proceedings of the 2018 IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies*, 2018, pp. 120–125.

[88] P. Poletti, R. Visintainer, B. Lepri, and S. Merler, “The interplay between individual social behavior and clinical symptoms in small clustered groups,” *BMC infectious diseases*, vol. 17, no. 1, p. 521, 2017.

[89] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, “A density-based algorithm for discovering clusters in large spatial databases with noise.,” in *Kdd*, vol. 96, 1996, pp. 226–231.

[90] Google, *Google activity recognition api*, 2020.

[91] Apple, *Cmmotionactivity*, 2020.

[92] A. Zheng and A. Casari, *Feature engineering for machine learning: principles and techniques for data scientists.* ” O’Reilly Media, Inc.”, 2018.

[93] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *arXiv preprint arXiv:1609.02907*, 2016.

[94] W. Hamilton, Z. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” *Advances in neural information processing systems*, vol. 30, 2017.

[95] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, “Graph attention networks,” *arXiv preprint arXiv:1710.10903*, 2017.

[96] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, “How powerful are graph neural networks?” *arXiv preprint arXiv:1810.00826*, 2018.

[97] B. Perozzi, R. Al-Rfou, and S. Skiena, “Deepwalk: Online learning of social representations,” *CoRR*, vol. abs/1403.6652, 2014. arXiv: 1403.6652.

[98] X. Liu, T. Murata, K.-S. Kim, C. Kotarasu, and C. Zhuang, “A general view for network embedding as matrix factorization,” in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019, pp. 375–383.

[99] B. Perozzi, V. Kulkarni, and S. Skiena, “Walklets: Multiscale graph embeddings for interpretable network classification,” *arXiv preprint arXiv:1605.02115*, 2016.

[100] D. Yang, P. Rosso, B. Li, and P. Cudre-Mauroux, “Nodesketch: Highly-efficient graph embeddings via recursive sketching,” in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1162–1172.

[101] J. Li, L. Wu, R. Guo, C. Liu, and H. Liu, “Multi-level network embedding with boosted low-rank matrix approximation,” in *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2019, pp. 49–56.

[102] H. Xiong, Y. Huang, L. E. Barnes, and M. S. Gerber, “Sensus: A cross-platform, general-purpose system for mobile crowdsensing in human-subject studies,” in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ser. UbiComp ’16, Heidelberg, Germany: Association for Computing Machinery, 2016, pp. 415–426, ISBN: 9781450344616.

[103] F. Errica, M. Podda, D. Bacciu, and A. Micheli, “A fair comparison of graph neural networks for graph classification,” *arXiv preprint arXiv:1912.09893*, 2019.

[104] C. Cangea, P. Veličković, N. Jovanović, T. Kipf, and P. Liò, “Towards sparse hierarchical graph classifiers,” *arXiv preprint arXiv:1811.01287*, 2018.

[105] J. Lee, I. Lee, and J. Kang, "Self-attention graph pooling," *arXiv preprint arXiv:1904.08082*, 2019.

[106] F. C. Pampel, *Logistic regression: A primer*. SAGE Publications, Incorporated, 2020, vol. 132.

[107] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

[108] M. W. Gardner and S. Dorling, "Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences," *Atmospheric environment*, vol. 32, no. 14-15, pp. 2627–2636, 1998.

[109] N. D. Lane and P. Georgiev, "Can deep learning revolutionize mobile sensing?" In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, 2015, pp. 117–122.

[110] F. Laport-López, E. Serrano, J. Bajo, and A. T. Campbell, "A review of mobile sensing systems, applications, and opportunities," *Knowledge and Information Systems*, vol. 62, no. 1, pp. 145–174, 2020.

[111] Z. Lou, L. Wang, and G. Shen, "Recent advances in smart wearable sensing systems," *Advanced Materials Technologies*, vol. 3, no. 12, p. 1 800 444, 2018.

[112] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2921–2929.

[113] P. E. Pope, S. Kolouri, M. Rostami, C. E. Martin, and H. Hoffmann, "Explainability methods for graph convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 772–10 781.

[114] C. Wu, L. Cai, M. S. Gerber, M. Boukhechba, and L. E. Barnes, "Vector space representation of bluetooth encounters for mental health inference," in *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*, 2018, pp. 1691–1699.

[115] M. Girolami, F. Mavilia, and F. Delmastro, "Sensing social interactions through ble beacons and commercial mobile devices," *Pervasive and Mobile Computing*, vol. 67, p. 101 198, 2020.

[116] F. Kargl, R. W. van der Heijden, B. Erb, and C. Bösch, "Privacy in mobile sensing," in *Digital Phenotyping and Mobile Sensing*, Springer, 2019, pp. 3–12.

[117] J. Bell, D. Butler, C. Hicks, and J. Crowcroft, *Tracesecure: Towards privacy preserving contact tracing*, 2020. arXiv: 2004.04059 `[cs.CR]`.

[118] K. L. Huang, S. S. Kanhere, and W. Hu, "Preserving privacy in participatory sensing systems," *Computer Communications*, vol. 33, no. 11, pp. 1266–1280, 2010.

[119] C. Liu, S. Chakraborty, and P. Mittal, "Deeprotect: Enabling inference-based access control on mobile sensing applications," *arXiv preprint arXiv:1702.06159*, 2017.

[120] X. Yin, Y. Zhu, and J. Hu, "A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, pp. 1–36, 2021.

[121] H. Zhang, Z. Zhang, A. Odena, and H. Lee, "Consistency regularization for generative adversarial networks," *arXiv preprint arXiv:1910.12027*, 2019.

[122] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

[123] A. Mustafa and R. K. Mantiuk, "Transformation consistency regularization-a semi-supervised paradigm for image-to-image translation," *arXiv preprint arXiv:2007.07867*, 2020.

[124] V. Verma, K. Kawaguchi, A. Lamb, J. Kannala, Y. Bengio, and D. Lopez-Paz, "Interpolation consistency training for semi-supervised learning," *arXiv preprint arXiv:1903.03825*, 2019.

[125] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[126] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[127] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1789–1819, 2021.

[128] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, PMLR, 2017, pp. 1273–1282.

[129] G. Dong, L. Cai, D. Datta, S. Kumar, L. E. Barnes, and M. Boukhechba, "Influenza-like symptom recognition using mobile sensing and graph neural networks," in *Proceedings of the Conference on Health, Inference, and Learning*, ser. CHIL '21,

Virtual Event, USA: Association for Computing Machinery, 2021, pp. 291–300, ISBN: 9781450383592.

[130] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowledge-Based Systems*, vol. 151, pp. 78–94, 2018.

[131] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[132] A. Albaseer, B. S. Ciftler, M. Abdallah, and A. Al-Fuqaha, "Exploiting unlabeled data in smart cities using federated edge learning," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*, IEEE, 2020, pp. 1666–1671.

[133] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, "Pseudo-labeling and confirmation bias in deep semi-supervised learning," in *2020 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2020, pp. 1–8.

[134] K. Sohn *et al.*, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," *arXiv preprint arXiv:2001.07685*, 2020.

[135] W. Jeong, J. Yoon, E. Yang, and S. J. Hwang, "Federated semi-supervised learning with inter-client consistency & disjoint learning," in *International Conference on Learning Representations*, 2021.

[136] Anxiety and D. A. of America. "Facts statistics." ().

[137] M. Boukhechba, P. Chow, K. Fua, B. A. Teachman, L. E. Barnes, *et al.*, "Predicting social anxiety from global positioning system traces of college students: Feasibility study," *JMIR mental health*, vol. 5, no. 3, e10101, 2018.

[138] R. Wang *et al.*, "Tracking depression dynamics in college students using mobile phone and wearable sensing," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 2, no. 1, pp. 1–26, 2018.

[139] K. O. Asare, Y. Terhorst, J. Vega, E. Peltonen, E. Lagerspetz, D. Ferreira, *et al.*, "Predicting depression from smartphone behavioral markers using machine learning methods, hyperparameter optimization, and feature importance analysis: Exploratory study," *JMIR mHealth and uHealth*, vol. 9, no. 7, e26540, 2021.

[140] M. Ilse, J. Tomczak, and M. Welling, "Attention-based deep multiple instance learning," in *International conference on machine learning*, PMLR, 2018, pp. 2127–2136.

[141] D. Hu, "An introductory survey on attention mechanisms in nlp problems," in *Proceedings of SAI Intelligent Systems Conference*, Springer, 2019, pp. 432–448.

[142] W. Falcon and K. Cho, “A framework for contrastive self-supervised learning and designing a new approach,” *arXiv preprint arXiv:2009.00104*, 2020.

[143] J. Klicpera, S. Weißenberger, and S. Günnemann, “Diffusion improves graph learning,” *arXiv preprint arXiv:1911.05485*, 2019.

[144] Y. Xie, Z. Xu, J. Zhang, Z. Wang, and S. Ji, “Self-supervised learning of graph neural networks: A unified review,” *arXiv preprint arXiv:2102.10757*, 2021.

[145] J. Zeng and P. Xie, “Contrastive self-supervised learning for graph classification,” *arXiv preprint arXiv:2009.05923*, 2020.

[146] K. Hassani and A. H. Khasahmadi, “Contrastive multi-view representation learning on graphs,” in *International Conference on Machine Learning*, PMLR, 2020, pp. 4116–4126.

[147] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, “Graph contrastive learning with augmentations,” *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[148] F. Plummer, L. Manea, D. Trepel, and D. McMillan, “Screening for anxiety disorders with the gad-7 and gad-2: A systematic review and diagnostic metaanalysis,” *General hospital psychiatry*, vol. 39, pp. 24–31, 2016.

[149] L. Manea *et al.*, “Identifying depression with the phq-2: A diagnostic meta-analysis,” *Journal of Affective Disorders*, vol. 203, pp. 382–395, 2016.

[150] L. G. Staples *et al.*, “Psychometric properties and clinical utility of brief measures of depression, anxiety, and general distress: The phq-2, gad-2, and k-6,” *General hospital psychiatry*, vol. 56, pp. 13–18, 2019.

[151] M. Giuliani, A. Gorini, S. Barbieri, F. Veglia, and E. Tremoli, “Examination of the best cut-off points of phq-2 and gad-2 for detecting depression and anxiety in italian cardiovascular inpatients,” *Psychology & Health*, pp. 1–14, 2020.

[152] A. J. Hughes, K. M. Dunn, T. Chaffee, J. J. Bhattarai, and M. Beier, “Diagnostic and clinical utility of the gad-2 for screening anxiety symptoms in individuals with multiple sclerosis,” *Archives of physical medicine and rehabilitation*, vol. 99, no. 10, pp. 2045–2049, 2018.

[153] J. B. Lee, R. Rossi, and X. Kong, “Graph classification using structural attention,” in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1666–1674.

[154] L. Zhao *et al.*, “T-gcn: A temporal graph convolutional network for traffic prediction,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848–3858, 2019.

[155] V. K. Garg, S. Jegelka, and T. Jaakkola, “Generalization and representational limits of graph neural networks,” *arXiv preprint arXiv:2002.06157*, 2020.

[156] D. Zügner and S. Günnemann, “Certifiable robustness of graph convolutional networks under structure perturbations,” in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1656–1665.

[157] L. Gong and Q. Cheng, “Exploiting edge features for graph neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9211–9219.

[158] C. Shorten and T. M. Khoshgoftaar, “A survey on image data augmentation for deep learning,” *Journal of Big Data*, vol. 6, no. 1, p. 60, 2019.

[159] K. Kong *et al.*, “Flag: Adversarial data augmentation for graph neural networks,” *arXiv preprint arXiv:2010.09891*, 2020.

[160] T. Zhao, Y. Liu, L. Neves, O. Woodford, M. Jiang, and N. Shah, “Data augmentation for graph neural networks,” *arXiv preprint arXiv:2006.06830*, 2020.

[161] S. Liu *et al.*, “Handling missing sensors in topology-aware iot applications with gated graph neural network,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4, no. 3, pp. 1–31, 2020.

[162] Y. Wang, M. Huang, X. Zhu, and L. Zhao, “Attention-based lstm for aspect-level sentiment classification,” in *Proceedings of the 2016 conference on empirical methods in natural language processing*, 2016, pp. 606–615.

[163] S. J. Moore, C. D. Nugent, S. Zhang, and I. Cleland, “Iot reliability: A review leading to 5 key research directions,” *CCF Transactions on Pervasive Computing and Interaction*, vol. 2, no. 3, pp. 147–163, 2020.

[164] M. A. Al-Garadi, A. Mohamed, A. K. Al-Ali, X. Du, I. Ali, and M. Guizani, “A survey of machine and deep learning methods for internet of things (iot) security,” *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1646–1685, 2020.

[165] A. Mosenia and N. K. Jha, “A comprehensive study of security of internet-of-things,” *IEEE Transactions on emerging topics in computing*, vol. 5, no. 4, pp. 586–602, 2016.

[166] W. W. Lo, S. Layeghy, M. Sarhan, M. Gallagher, and M. Portmann, "E-graphsage: A graph neural network based intrusion detection system for iot," *arXiv preprint arXiv:2103.16329*, 2021.

[167] M. Shafique *et al.*, "Adaptive and energy-efficient architectures for machine learning: Challenges, opportunities, and research roadmap," in *2017 IEEE Computer society annual symposium on VLSI (ISVLSI)*, IEEE, 2017, pp. 627–632.

[168] X. Wang, Y. Han, V. C. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 869–904, 2020.

[169] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, "A survey of deep learning applications to autonomous vehicle control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 712–733, 2020.

[170] G. Li, Z. Liu, F. Li, and J. Cheng, "Block convolution: Towards memory-efficient inference of large-scale cnns on fpga," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2021.

[171] N. D. Lane *et al.*, "Deepx: A software accelerator for low-power deep learning inference on mobile devices," in *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, IEEE, 2016, pp. 1–12.

[172] R. Mishra, H. P. Gupta, and T. Dutta, "A survey on deep neural network compression: Challenges, overview, and solutions," *arXiv preprint arXiv:2010.03954*, 2020.

[173] S. Sun, Y. Cheng, Z. Gan, and J. Liu, "Patient knowledge distillation for bert model compression," *arXiv preprint arXiv:1908.09355*, 2019.

[174] Y. Liu *et al.*, "Knowledge distillation via instance relationship graph," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7096–7104.

[175] S. Lee and B. C. Song, "Graph-based knowledge distillation by multi-head attention network," *arXiv preprint arXiv:1907.02226*, 2019.

[176] C. Lassance, M. Bontonou, G. B. Hacene, V. Gripon, J. Tang, and A. Ortega, "Deep geometric knowledge distillation with graphs," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2020, pp. 8484–8488.