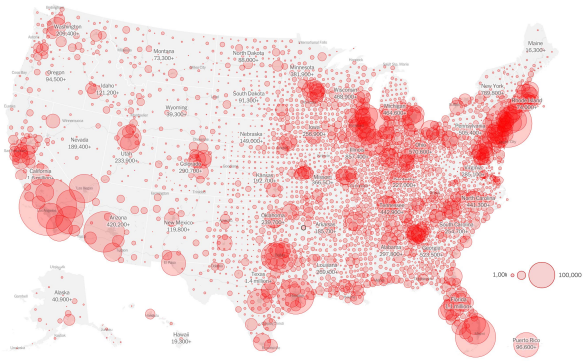
Toward Combatting COVID-19: A Risk Assessment System

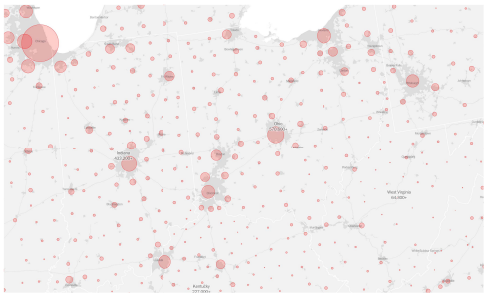
Qianlong Wang, *Graduate Student Member, IEEE*, Yifan Guo, *Graduate Student Member, IEEE*, Tianxi Ji, *Graduate Student Member, IEEE*, Xufei Wang, *Graduate Student Member, IEEE*, Bingfang Hu, and Pan Li, *Member, IEEE*

***Abstract*—The coronavirus disease 2019 (COVID-19) has rapidly become a significant public health emergency all over the world since it was first identified in Wuhan, China, in December 2019. Until today, massive disease-related data have been collected, both manually and through the Internet of Medical Things (IoMT), which can be potentially used to analyze the spread of the disease. On the other hand, with the help of IoMT, the analysis results of the current status of COVID-19 can be delivered to people in real time to enable situational awareness, which may help mitigate the disease spread in communities. However, current accessible data on COVID-19 are mostly at a macrolevel, such as for each state, county, or metropolitan area. For fine-grained areas, such as for each city, community, or geographical coordinate, COVID-19 data are usually not available, which prevents us from obtaining information on the disease spread in closer neighborhoods around us. To address this problem, in this article, we propose a two-level risk assessment system. In particular, we define a "risk index." Then, we develop a risk assessment model, called MK-DNN, by taking advantage of the multikernel density estimation (MKDE) and deep neural network (DNN). We train MK-DNN at the macrolevel (for each metro area), which subsequently enables us to obtain the risk indices at the microlevel (for each geographic coordinate). Moreover, a heuristic validation method is further designed to help validate the obtained microlevel risk indices. Simulations conducted on real-world data demonstrate the accuracy and validity of our proposed risk assessment system.**

***Index Terms*—Coronavirus disease 2019 (COVID-19), deep neural network (DNN), Internet of Medical Things (IoMT), kernel density estimation (KDE), risk assessment.**



(a)



(b)

Fig. 1. Confirmed cases (a) in the U.S. and (b) in Ohio State (by January 8, 2021) [9]. Size of the circle represents the number of cases in the area.

I. Introduction

A CORONAVIRUS disease that was first outspread in Wuhan city of China in December 2019, named coronavirus disease 2019 (COVID-19), raised intense attention not only within China but also internationally. As an infectious disease, COVID-19 has been regarded as a significant health crisis in the U.S. and worldwide due to its rapid outbreak [1], [2]. As of January 8, 2021, the disease has resulted in over 22 million confirmed cases and 370 700 reported deaths in the U.S. and over 89 million confirmed cases and over 1.9 million reported deaths worldwide. Fig. 1(a) shows the distribution of confirmed cases over states and counties in the U.S. During the COVID-19 outspread, the Internet of Medical Things (IoMT) as an extension and specialization of the Internet of Things (IoT) has been applied to combat COVID-19. On the one hand, IoMT helps collect informative disease-related data by using smart sensors [3], [4]. Combined with other related sources, e.g., demographic and geographical data, these disease-related data can be potentially used to analyze the spread of the disease. On the other hand, IoMT can utilize these analysis results and enable smart medical/personal devices to track and monitor the progression of COVID-19, improve people's situational awareness, and hence, may help mitigate the disease spread in communities [5]–[8].

Manuscript received January 10, 2021; revised February 28, 2021; accepted March 16, 2021. Date of publication March 31, 2021; date of current version October 22, 2021. *(Corresponding author: Pan Li.)*

Qianlong Wang is with the Department of Computer and Information Sciences, Towson University, Towson, MD 21252 USA (e-mail: qwang@towson.edu).

Yifan Guo, Tianxi Ji, Xufei Wang, and Pan Li are with the Department of Electrical, Computer, and Systems Engineering, Case Western Reserve University, Cleveland, OH 44106 USA (e-mail: yxg383@case.edu; txj116@case.edu; xxw512@case.edu; pxl288@case.edu).

Bingfang Hu is with the School of Medicine, Case Western Reserve University, Cleveland, OH 44106 USA (e-mail: bxh430@case.edu).

Digital Object Identifier 10.1109/JIOT.2021.3070042

However, current accessible data on COVID-19 are mostly at the macrolevel [10], [11]. As shown in Fig. 1(b), considering the example of Ohio, only confirmed case information for counties and metropolitan areas is accessible. Nevertheless, disease-related data at the microlevel, e.g., for cities and communities, are unavailable to the public. Therefore, we are faced with the following question: can we model the status of the virus outspread and estimate the risk not only at the macrolevel, e.g., for counties and metros, but also at the microlevel, e.g., cities, communities, and even geographical coordinates on the map?

Existing works related to disease risk assessment are limited. Some of them adopt various statistic or learning tools, e.g., autoregressive integrated moving average (ARIMA), logistic regression, and stacked autoencoder, to estimate the growing cases of certain areas in the long or short term [12]–[14]. Others focus on combining disease-related data with multisource data, e.g., Twitter comments and Web news, to estimate the severity of the disease in certain areas [42], [43]. To the best of our knowledge, how to estimate the risk of COVID-19 at the microlevel remains an open and challenging research problem.

In this article, we develop a two-level risk assessment system. We first define a "risk index." Then, we design a risk assessment model called MK-DNN by taking advantage of the multikernel density estimation (MKDE) and deep neural network (DNN). We train MK-DNN at the macrolevel, and subsequently use the trained model to obtain risk indices at the microlevel, particularly at each geographic coordinate. The flowchart of our system is shown in Fig. 2. Specifically, our system mainly consists of four components, which are: 1) data collection; 2) risk definition; 3) two-level modeling; and 4) result validation. In the first part, we adopt data from multisources, including both disease-related and demographic data, for better analyzing the disease spread. Second, we define a risk index based on the attributes collected from multisourced data in order to quantify the risk of COVID-19. Third, we design a two-level modeling process. At the macrolevel, the collected multisourced data are used to generate a kernel density estimation (KDE) map. Then, the features extracted from these KDE maps and the calculated risk indices by definition are used to train a DNN at the macrolevel. At the microlevel, the trained MK-DNN is applied to assess the risk indices at each geographic coordinate. Since there is no ground-truth data of risk indices at the microlevel, in the fourth part, we further design a heuristic scheme to validate the risk indices at the microlevel estimated by MK-DNN, which is also a two-level procedure. At the macrolevel, we train a validation network (an independent DNN) to infer the confirmed case of a metro area by using MK-DNN assessed risk index of that metro and the demographic data as the input. At the microlevel, a similar validation network is applied at the microlevel within that metro and used to estimate the confirmed case number at the microlevel. Then, the confirmed case number at the macrolevel can be estimated by summing up all the estimated confirmed case numbers at the microlevel. The difference between the macro and microlevel validation scores can effectively help validate the estimated risk indices at the microlevel.

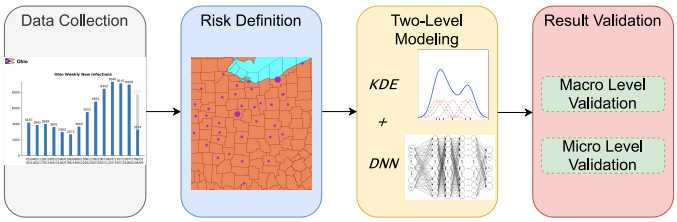


Fig. 2. Flowchart of the proposed COVID-19 assessment system.

In addition, our microlevel validation also helps optimize the hyperparameter in MKDE.

The main contributions of this article are summarized as follows.

1) We define a risk index for quantifying the risk of COVID-19.
2) We develop a risk assessment model based on MKDE and DNN, called MK-DNN. We train MK-DNN at the macrolevel and then apply it at the microlevel to obtain risk indices.
3) We further design a heuristic validation method to validate the risk estimated at the microlevel.
4) Simulations based on the up-to-date disease-related data and demographic data demonstrate the effectiveness of our proposed COVID-19 risk assessment system.

The remainder of this article is organized as follows. Section II introduces the most related work for combatting COVID-19. Section III details the proposed risk assessment system. In Section IV, we conduct simulations to evaluate the performance of the proposed COVID-19 assessment model and further validate the results. Finally, we conclude this article in Section V.

II. Related Work

In this section, we introduce the works that utilize various mathematical and learning tools to combat COVID-19. These works can be generally classified into two categories, the first of which focuses on disease diagnose and treatment in the biomedical domain, and the second of which focuses on the outbreak pattern and trend analysis for the disease.

To diagnose the disease, Chen *et al.* [17] proposed a prospective study on applying the deep learning model to detect coronavirus pneumonia through computed tomography (CT) images. Song *et al.* [18] proposed the details relation extraction neural network (DRE-Net) to extract the top-K details in the CT images for identifying patients with COVID-19. Wang *et al.* [19] proposed a deep learning-based detection method, where the authors randomly selected regions of interest in CT images and used the Inception network to extract features. Randhawa *et al.* [20] identified an intrinsic COVID-19 virus genomic signature and introduced a machine learning-based approach for the classification of virus genomes. Xu *et al.* [21] compared multiple convolutional neural network (CNN) models to classify CT samples with COVID-19, influenza viral pneumonia, or no-infection. Rao and Vazquez [22] proposed a machine learning-based COVID-19 case identification method by using a mobile

phone-based Web survey. Moreover, to help the treatment process, Shi *et al.* [23] proposed a deep learning-based quantitative CT Model in predicting the severity of COVID-19 patients. Yan *et al.* [24] developed a machine learning-based prognostic model to predict the survival rate for individual severe patients by using three key clinical features, i.e., lactic dehydrogenase (LDH), lymphocyte, and high-sensitivity C-reactive protein (hsCRP).

Other works focus on learning the disease outbreak patterns, aiming to analyze the severity for certain areas and further predict the trend of outbreak. Wang *et al.* [25] adopted a logistic model to predict the trend of the epidemic. Similarly, Hermanowicz [13] used a logistic growth model to estimate the growing cases in near-real time. Hu *et al.* [14] developed a modified stacked autoencoder for real-time forecasting the confirmed cases of COVID-19 across China. Majumder and Mandl [26] combined public-available cumulative cases from the ongoing outbreak with phenomenological modeling methods to conduct a preliminary transmissibility assessment. Song *et al.* [27] developed a health informatics toolbox that enables public health workers to timely analyze and evaluate the time-course dynamics of COVID-19 through a Markov SIR infectious disease process. Zhu *et al.* [28] introduced the virus host prediction (VHP) to predict the potential hosts of viruses using a deep learning algorithm. Chakraborty and Ghosh [12] presented a hybrid approach based on the ARIMA model and Wavelet-based forecasting model generating short-term (real-time) forecasts of the future COVID-19 cases for multiple countries. Roda *et al.* [29] proposed a study, indicating that a simpler model may be more reliable on learning the trend of the epidemic, and further modeled the potential of a second outbreak after the return-to-work in the city. Sajadi *et al.* [30] presented an analysis of temperature and latitude for predicting the spreading trend of the COVID-19.

Note that very few works study the risk assessment at the microlevel. Among them, Jahanbin and Rahmanian [15] developed a fuzzy rule-based evolutionary algorithm, called Eclass1-MIMO, which can mine twitter and Web news to predict morbidity rates in a certain region. Ye *et al.* [16] proposed an AI-driven system, called α-Satellite, to extract features from heterogeneous sources and provide community-level risk assessment under COVID-19. These works mainly focus on feature extraction for predicting the risk level for areas with accessible disease-related data. In this work, we develop a two-level risk assessment system for estimating the risk index at the microlevel without available disease-related data.

III. Two-Level Risk Assessment System

The structure of our two-level risk assessment system has been shown in Fig. 2. In the following, we elaborate on the four system components, i.e., multisourced data collection, risk definition, two-level modeling, and validation, respectively.

A. Multisourced Data Collection

The severity of disease spread depends on many factors, such as population density, age distribution, and the increasing speed of confirmed cases. Properly identifying data sources helps us better define the risk level and learn a risk prediction model by mining intrinsic disease spread patterns. Relying on a single-source data often leads to the learned model mediocre, because single-source data are not informative for the model to extract the complex patterns in the data. In our system, large-scale multisourced data, including both disease-related and demographic data, are collected for risk defining and model learning.

1) Disease Related Data: We collect up-to-date public disease status data from authoritative organizations, e.g., Centers for Disease Control and Prevention (CDC), and the World Health Organization (WHO) [31], [32]. The status data include the accumulated number of confirmed cases, death cases, recovered cases, and weekly hospitalizations regarding each state, county, and metros. Based on the collected data, we can calculate daily cases and increasing rates for each category, which are also essential features for risk defining and model learning.

2) Demographic Data: The demographic data are collected from United States Census Bureau and county government websites, which include population, population per square mile (density), area (square miles), age distribution, and housing units regarding every state, county, metro, and city. Some attributes are essential indices that reflect the risk level for corresponding areas. For example, high population density intuitively leads to a high spread rate of the disease. A large ratio of senior adults in a particular area may cause more spread of disease and more severe complications after being infected due to their lower immunity. To better utilize these data, we further calculated the median age for each area as an extra attribute. Moreover, for a better model learning performance, the model should be applied to more fine-grained data. Thus, we mainly adopt demographic data over the city level for model learning.

B. Risk Index Definition

We define a risk index to quantify the risk under COVID-19. Ideally, the risk level should well reflect the severity of the epidemic in a certain area, including the ratio and increasing speed of the infection. Thus, both disease-related data, e.g., the confirmed case number and demographic data are critical elements for risk definition, which are detailed as follows.

1) Disease Related Attributes $\mathbf{a}_1$*:* For a given area, the disease-related attribute includes the confirmed case number, death case number, increasing speed (new cases), and acceleration (increasing speed of new cases), which is represented by the vector $\mathbf{a}_1$. For example, as of May 11, 2020, the Cleveland metropolitan area in Ohio State had 2861 confirmed cases, 147 death cases, 66 new cases, and 26 cases on new cases compared with the day before. Thus, the disease-related attribute $\boldsymbol{a}_1 \in \mathbb{R}^4$ for the area is denoted as $\mathbf{a}_1 = [2861, 147, 66, 26]$.

2) Demographic Attributes $\mathbf{a}_2$*:* To generate the risk level, we mainly adopt population and age-related information, which are the most critical indices related to the risk level. Particularly, $\mathbf{a}_2$ contains the population, population

density, and median age. For example, as of the latest data, the Cleveland metropolitan population has a population of 396 815, a density of 5107 (population per square mile), and the median age of 35.7. Thus, the demographic attribute $\boldsymbol{a}_2 \in \mathbb{R}^3$ for the area is denoted as $\mathbf{a}_2 = [396\,815, 5107, 35.7]$.

By aggregating the fetched attributes through linear combination, we define the risk index R for a specific area as follows:

$$R = \sum_{i=1}^{\kappa} \alpha_i \boldsymbol{A}_R(i), i \in [1, \kappa]$$
$$\boldsymbol{A}_R = \boldsymbol{a}_1 \oplus \boldsymbol{a}_2. \tag{1}$$

α_i is a weight factor that indicates the importance of each attribute. $\boldsymbol{A}_R$ is the set of attributes used for obtaining risk. $\boldsymbol{A}_R(i)$ represents the ith element in $\boldsymbol{A}_R$, where κ is the length of vector $\boldsymbol{A}_R$. $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$ are the aforementioned disease-related and demographic attributes. Note that $\sum_{i=1}^{\kappa} \alpha_i = 1$. $\oplus$ is the vector concatenation operator.

C. Two-Level Modeling of MK-DNN

We employ a widely used deep learning method, i.e., DNN [33], [34], to predict the risk index for a particular area as defined above [35]. However, the COVID-19 data are not available in the microlevel areas. Here, we apply a nonparametric statistic tool, i.e., KDE [36], to extract features for certain areas, which can be used as the input of the DNN model and, hence, estimate the risk index for the corresponding areas. Specifically, we train and test our DNN model at the macrolevel using the risk indices that can be directly calculated by the definition as the true label, which is called macrolevel modeling. The trained DNN can then be applied to the microlevel to obtain the risk indices, which is called microlevel modeling. The framework is elaborated as follows.

1) Macrolevel Modeling: In particular, there are four main procedures, which include coordinate sampling, feature map building, feature extraction, and training and testing. As mentioned above, we consider a metro area at the macrolevel and a single coordinate at the microlevel. The flowchart of macrolevel modeling is shown in Fig. 3. Specifically, an independent KDE function is applied for each attribute, e.g., the confirmed case number or population. We adopt a total of K attributes for building KDEs. Thus, there are a total of K KDE feature density maps. Note that K is the number of attributes for building KDEs, while κ is the number of attributes used for defining the risk index R. We adopt more attributes to build more feature maps for better model learning performance and, hence, $K > \kappa$. In feature extraction, K learned feature maps are used to generate features used as the input of the following DNN. In particular, considering a metro area, we first sample a certain amount of coordinates within the metro area. The reason for sampling coordinates is to incorporate more information for model learning. Using a single coordinate to extract features may be improper and insufficient. By sampling more coordinates, the metro's status can be better represented and the extracted features become more informative. After sampling, for each feature map, the learned KDE first outputs the estimation, called feature density, for each sampled

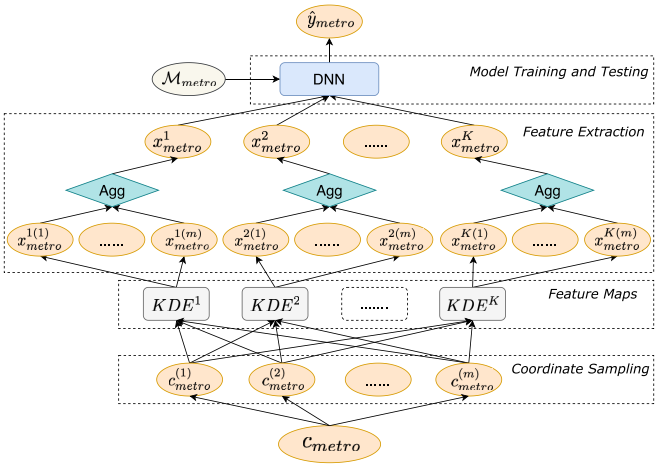


Fig. 3. Flowchart of the macrolevel prediction. Input a metro coordinate as example. The output is the predicted risk level for the metro. "Agg" represents the aggregation function. $\mathcal{M}_{\text{metro}}$ is the metadata of the metro.

coordinate. Then, the output feature densities are aggregated to a single value, which is considered as the corresponding extracted feature. By sampling multiple coordinates in a metro area, more KDE estimated feature densities are considered; thus, the status of the metro can be well represented. As there are K feature maps, K features are extracted and input to the DNN along with the metadata $\mathcal{M}_{\text{metro}}$ for training and testing. In the following, each main procedure is elaborated.

a) Coordinate sampling: As mentioned above, instead of using a single coordinate, we sample m coordinates within the metro area to better represent the metro's status. In particular, given a metro with the coordinate c_{metro}, we draw $l \times l$ grids with a certain interval between nodes, where the center of the grids is located on c_{metro}. Note that $m = l \times l$. l and the interval between nodes are predefined parameters. Thus, we obtain the sample coordinates for c_{metro}, that is, $\tilde{\mathbf{c}}_{\text{metro}} = [c^{(1)}_{\text{metro}}, \ldots, c^{(m)}_{\text{metro}}]$.

b) Feature map building: For each attribute, a feature map is built based on KDE. Taking the attribute of the confirmed case number, and the granularity level of metro as an example, the feature map is built as follows. Considering a total of n confirmed cases, each case i is assigned with a coordinate $c_i = [\text{lat}_i, \text{lon}_i]$ of the belonged metro. lat_i and lon_i are the corresponding latitude and longitude. Hence, the confirmed cases can be represented by $\mathbf{c}_{\text{confirmed}} = [c_1, \ldots, c_n]$. The coordinates of each metro can be obtained from the United States Census Bureau. Thus, the corresponding KDE function can be represented as

$$f(c) = \frac{1}{nh} \sum_{i=1}^{n} \phi\left(\frac{||c - c_i||}{h}\right). \tag{2}$$

c is the coordinate of the interest. c_i is the element in $\mathbf{c}_{\text{confirmed}}$. ϕ is the nonnegative kernel function. $h > 0$ is the bandwidth, which is a smoothing parameter. With (2), the estimated feature density for any coordinate of interest c can be calculated. Thus, the corresponding feature map is generated. Note that KDEs of different attributes may be built based on different

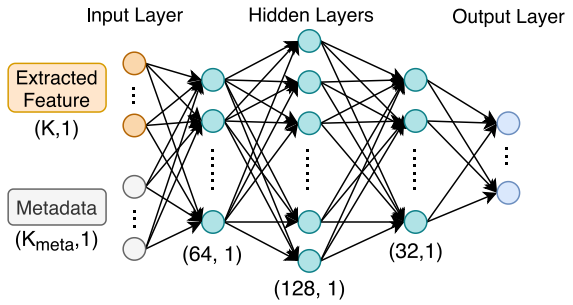


Fig. 4. Structure of our DNN. K and K_{meta} are the lengths of extracted feature and metadata, respectively. The size of the output layer depends on the type of the model task. Section IV elaborates both input layer and output layer sizes.

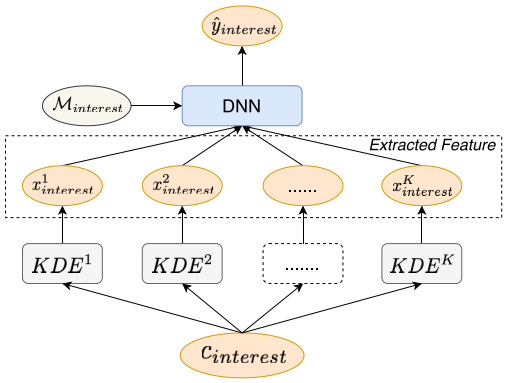


Fig. 5. Flowchart of the microlevel prediction. c_{interest} is the coordinate of interest, and $\mathcal{M}_{\text{interest}}$ is the corresponding metadata.

granularity levels, which means that different KDEs have different resolutions. For instance, for confirmed case and death case attributes, the most fine-grained granularity is metro level, so each case is assigned a coordinate of the belonged metro. For population density attribute, the granularity level can be finer, i.e., communities in the city. Thus, each node of the population density attribute will be assigned with a coordinate of the belonged community. Moreover, we remain a consistent bandwidth of h for each KDE, where different choices of h are evaluated in the simulation.

c) Feature extraction: Next, we extract features for each attribute based on the generated feature maps. As there are a total of K attributes, the extracted feature for a particular metro area is represented by $\mathbf{x}_{\text{metro}} = [x^1_{\text{metro}}, \ldots, x^K_{\text{metro}}]$. As it is shown in Fig. 3, to obtain each element $x^j_{\text{metro}} \in \mathbf{x}_{\text{metro}}$, m coordinates within the metro have been sampled. We calculate the KDE estimated feature density for each sampled coordinate and apply an aggregation method to generate x^j_{metro}. Specifically, considering $x^j_{\text{metro}} \in \mathbf{x}_{\text{metro}}$, we have

$$x^j_{\text{metro}} = A\left(f_j\left(c^{(1)}_{\text{metro}}\right), \ldots, f_j\left(c^{(m)}_{\text{metro}}\right)\right). \tag{3}$$

$A(\cdot)$ is an aggregation function. $j \in [1, K]$ is the index of a certain attribute, e.g., the confirmed case, population density, etc. $f_j(\cdot)$ is the corresponding KDE function as shown in (2), where $c_i \in \mathbf{c}_j$. Specifically, we adopt mean aggregation. Thus, (3) is rewritten as

$$x^j_{\text{metro}} = \frac{1}{m}\sum_{i=1}^{m} f_j\left(c^{(i)}_{\text{metro}}\right), i \in [1, m], j \in [1, K]. \tag{4}$$

$c^{(i)}_{\text{metro}}$ is the element in $\tilde{\mathbf{c}}_{\text{metro}}$. By calculating (4) over K attributes, we obtain the extracted feature vector $\mathbf{x}_{\text{metro}}$ for the certain metro.

d) Training and testing: So far, we have defined the risk and extracted feature vector for each macrolevel area, i.e., a metro. Using extracted feature vector and defined risk as input and true label, respectively, a learning model can be trained and tested for predicting the risk index. To design the structure of our learning model, we mainly consider the following factors. First, as we adopt data from both spatial and time domains, e.g., confirmed cases, new cases, and acceleration (detailed in the simulation), the model capacity is a major concern. Second, the model needs to be trained efficiently for better applicability in real-world applications in IoMT. Considering these, DNN is adopted in our system due to its strong ability to learn the nonlinear complex correlations among data and its high training efficiency compared with other well-known models, e.g., convolutional or recurrent neural networks [37]. In particular, the adopted DNN structure is shown in Fig. 4, where there are a total of three hidden layers. The size of each hidden layer is displayed in the figure. The input of the DNN is a vector that concatenates both extracted feature and the metadata, with the length of $K+K_{\text{meta}}$. The output layer size is not fixed because we have developed different types of model tasks for the performance test. In particular, we develop both regression and classification tasks to test the DNN performance, where the mean square error (MSE) and cross-entropy are adopted for the loss function, respectively.

In the regression task, y and $\hat{y}$ (subscript metro is omitted for simplicity) are the defined risk and the predicted risk, respectively. The corresponding MSE loss can be calculated by

$$\mathcal{L}_{\text{reg}} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{5}$$

where N is the total number of data samples.

In the classification task, the defined risk y is first categorized into a p-length one-hot vector $\mathbf{y} \in \Delta^p$, where $\Delta^p = \{\mathbf{y} \in \{0, 1\}^p, \mathbf{1}^T\mathbf{y} = 1\}$. p is the number of classes, where each of them represents different risk level. Correspondingly, the output of the DNN model becomes $\hat{\mathbf{y}} \in \Delta'^p$, where $\Delta'^p = \{\hat{\mathbf{y}} \geq 0, \mathbf{1}^T\hat{\mathbf{y}} = 1\}$ is predicted score over p classes. Hence, the corresponding cross-entropy loss is calculated by

$$\mathcal{L}_{\text{class}} = -\frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i^T \log \hat{\mathbf{y}}_i. \tag{6}$$

By developing both regression and classification tasks, the performance of the MK-DNN model is extensively evaluated.

2) Microlevel Modeling: By taking advantage of KDE, the prediction model can be further extended to a microlevel. That is, for any coordinate of interest, the model is able to output a risk level assessment. In particular, the flowchart of the microlevel modeling is shown in Fig. 5. Considering a

coordinate of interest c_{interest}, we first apply the KDEs generated from the macrolevel model to calculate the feature density for K attributes. Thus, the output of KDEs $\mathbf{x}_{\text{interest}} = [x^1_{\text{interest}}, \ldots, x^K_{\text{interest}}]$ is considered as the extracted feature and inputted to the DNN model trained in the macrolevel modeling. For $\mathcal{M}_{\text{interest}}$, we directly adopt the corresponding macrolevel metadata as the metadata of c_{interest}.

D. Validation

Our model can now be trained over macrolevel and obtain the risk indices for microlevel areas. Since we have ground-truth data available at the macrolevel, it is straightforward to train our risk assessment system and evaluate its performance at the macrolevel. Since ground-truth data are unavailable at the microlevel, we evaluate the performance of our risk assessment system by comparing the sum of the microlevel confirmed case number and the macrolevel confirmed case number obtained based on risk indices. Specifically, we design validation score functions at both macrolevel and microlevel. The reason for designing validation score functions is to make evaluation metrics consistent at both macrolevel and microlevel validations. The consistent validation score functions at both levels can demonstrate the degradation of the model performance when it is applied at the microlevel. Intuitively, a small validation score can demonstrate the effectiveness of our model at the microlevel. The details of macrolevel and microlevel validation processes are presented as follows.

1) Macrolevel Validation: In macrolevel validation, we verify the validity of the MK-DNN's macrolevel-predicted risk indices. The structure of macrolevel validation is shown in Fig. 6, where we take a certain metro of interest as the input to validation. Specifically, the trained MK-DNN is first applied to generate the corresponding predicted risk level. Here, we adopt the continuous result from the regression task for validation. Then, the aforementioned demographic attribute $\mathbf{a}_2$ is concatenated to the MK-DNN output, which is used as the input of the validation network. Based on the true confirmed case number of the metro $\mathbf{a}^{\text{confirmed}}_{1,\text{metro}}$ and the predicted result $\hat{\mathbf{a}}^{\text{confirmed}}_{1,\text{metro}}$, the network generates a validation score $s^{\text{macro}}_{\text{metro}}$, which is calculated by

$$s^{\text{macro}}_{\text{metro}} = \exp\left(-\left(\frac{\hat{\mathbf{a}}^{\text{confirmed}}_{1,\text{metro}} - \mathbf{a}^{\text{confirmed}}_{1,\text{metro}}}{\mathbf{a}^{\text{confirmed}}_{1,\text{metro}}}\right)^2\right). \tag{7}$$

Moreover, we adopt MSE as the loss function to train the validation network. By minimizing the loss during the training process, it is obvious to see the validation score $s^{\text{macro}}_{\text{metro}}$ will increase.

As discussed previously, the defined risk is an effective metric that reflects the ratio of infections in a certain area. Therefore, by inputting the obtained risk indices and demographic attributes, the validation score effectively shows the risk's validity. A higher score indicates that the obtained risk index effectively depicts the relation between the demographic feature and the confirmed case, and can be easily learned by the network. Conversely, a lower score indicates the ineffectiveness of the obtained risk index. Thus, the macrolevel obtained risk index is validated.

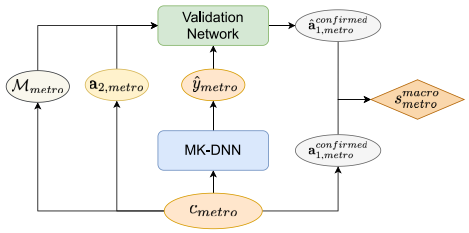


Fig. 6. Process of the macrolevel validation. $\mathbf{a}^{\text{confirmed}}_{1,\text{metro}}$ and $\hat{\mathbf{a}}^{\text{confirmed}}_{1,\text{metro}}$ represent the confirmed case number in the current metro and the predicted result of the validation network, respectively. $\mathbf{a}_{2,\text{metro}}$ is the vector of demographic attributes for the current metro, $\mathcal{M}_{\text{metro}}$ is the metadata. $s^{\text{macro}}_{\text{metro}}$ is the calculated macrolevel validation score.

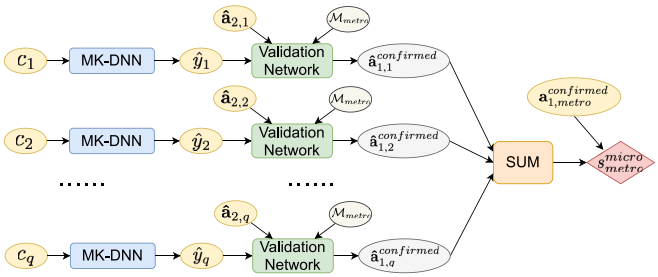


Fig. 7. Process of the microlevel validation. $\{c_1, \ldots, c_q\}$ are the coordinates of q districts. $\{\mathbf{a}_{2,1}, \ldots, \mathbf{a}_{2,q}\}$ are the corresponding demographic attributes vectors. $\mathbf{a}^{\text{confirmed}}_{1,\text{metro}}$ and $\{\hat{\mathbf{a}}^{\text{confirmed}}_{1,1}, \ldots, \hat{\mathbf{a}}^{\text{confirmed}}_{1,q}\}$ represent the confirmed case number in the current metro and the predicted results for q districts, respectively. $s^{\text{micro}}_{\text{metro}}$ is the calculated microlevel validation score.

2) Microlevel Validation: Since there are no true data of the confirmed case and demographic data regarding each coordinate, the macrolevel validation method cannot be directly applied in the microlevel. However, by taking advantage of the previously built KDE functions, we are able to estimate the demographic data on a microlevel district. Hence, with the same validation network trained in the macrolevel, we can predict the confirmed case number in a microlevel district using the estimated demographic data. Realizing this, we propose a heuristic method to indirectly validate the obtained risk index at the microlevel, which is to evaluate the sum of the predicted confirmed case numbers in multiple districts. In this way, the true confirmed case number in the macrolevel can be utilized for validation. By checking the difference between the sum value and true macrolevel data, we can obtain the accumulated error of microlevel predictions. Hence, to a certain extent, the validity of microlevel's obtained risk indices are evaluated.

The structure of the microlevel validation is shown in Fig. 7. Specifically, we first chop a metro into q districts and sample a coordinate for each of them, hence obtaining a coordinate set $\{c_1, \ldots, c_q\}$. For each coordinate c_i, the microlevel modeling in MK-DNN is applied to obtain risk index $\hat{y}_i$ for the corresponding district. Note that a district is fairly small compared with macrolevel areas. Thus, we directly use the obtained risk index of the coordinate to represent the corresponding district. Then, we estimate the demographic attributes for the corresponding district. The estimated demographic attributes

for c_i are denoted by $\hat{\mathbf{a}}_{2,i}$. A certain attribute $\hat{\mathbf{a}}_{2,i}^{\text{attr}} \in \hat{\mathbf{a}}_{2,i}$ is calculated by

$$\hat{\mathbf{a}}_{2,i}^{\text{attr}} = \mathbf{a}_{2,\text{metro}}^{\text{attr}} \frac{f_{\text{attr}}(c_i)}{\sum_{j=1}^{q} f_{\text{attr}}(c_j)}. \tag{8}$$

$f_{\text{attr}}(\cdot)$ is the KDE function for the certain attribute. Similar with microlevel MK-DNN, we directly adopt the corresponding macrolevel metadata as the metadata of each coordinate. Thus, the validation network trained in macrolevel validation can be applied to output predicted confirmed case number $\hat{\mathbf{a}}_{1,i}^{\text{confirmed}}$ for c_i. By summing up q predicted numbers, we obtain $\hat{\mathbf{a}}_{1,\text{sum}}^{\text{confirmed}}$, which is used to compared with the true confirmed case number of the corresponding metro $\mathbf{a}_{1,\text{metro}}^{\text{confirmed}}$ and finally generate the microlevel validation score $s_{\text{metro}}^{\text{micro}}$. Similar to (7), $s_{\text{metro}}^{\text{micro}}$ is calculated by

$$s_{\text{metro}}^{\text{micro}} = \exp\left(-\left(\frac{\hat{\mathbf{a}}_{1,\text{sum}}^{\text{confirmed}} - \mathbf{a}_{1,\text{metro}}^{\text{confirmed}}}{\mathbf{a}_{1,\text{metro}}^{\text{confirmed}}}\right)^2\right). \tag{9}$$

So far, both macro and microlevel validation scores are obtained. Comparing the difference between two scores can effectively help validate if macrolevel and microlevel obtained risk indices are consistent. In other words, a small difference demonstrates that the risk indices obtained at the microlevel are still effective under our risk definition, while a large difference demonstrates that the risk indices are far from the risk definition. In addition, it should be noted that the validation method is not used to validate the effectiveness of our risk definition. As risk index is essentially a subjective definition that weighs the severity of the pandemic, to the best of our knowledge, most existing risk indexing methods focus on extracting certain attributes to generate risk indices, rather than validating the effectiveness of the risk definition [16]. In our system, it is either not the main objective to validate the defined risk indices. The major challenge is to effectively obtain the risk index for both macro and micro areas, given a certain risk definition. From the above, this goal is achieved by using the MK-DNN model to extract features for all points (coordinates) on the map, hence enabling the model to obtain risk index for both macrolevel and microlevel areas. In the following section, we show the performance of the MK-DNN model and apply validation methods to help verify the results. Moreover, as the microlevel validation utilizes the KDEs of the MK-DNN model, the hyperparameter setup in KDEs may somehow influence the validation score. Hence, microlevel validation can additionally help optimize the hyperparameter in KDE. Further discussion is given in the following section.

IV. Performance Evaluation

In this section, we conduct the performance evaluation of the proposed MK-DNN model and show the validation results. In particular, we first introduce the details of the data set adopted for the simulation and data preprocessing. Then, we show the defined risk indices for macrolevel areas, followed by the training and testing results of macrolevel MK-DNN and the microlevel MK-DNN obtained hotspot map. We also introduce other well-known deep learning models and a state-of-the-art risk assessment model for performance comparison. In the end, we validate risk indices obtained by both macro and microlevel MK-DNN.

A. Data Set and Data Preprocessing

For disease-related attribute $\mathbf{a}_1$, we adpt the up-to-date data set "U.S. Metropolitan Daily Cases with Basemap" from a public research data repository, Harvard Dataverse [38], [39]. The data set includes time-series confirmed and death COVID-19 cases in the U.S. metropolitan areas from January 22, 2020, to July 29, 2020, 190 days. We picked the metropolitan areas in OH state for system performance evaluation, which includes 36 areas. Thus, both the confirmed and the dead cases data contain a total of $36 \times 190 = 6840$ raw data items. To fully incorporate the spatial–temporal correlations in the model learning, we also introduce the daily confirmed cases (new cases) and increment daily confirmed cases (acceleration), where the number of new cases is calculated by the confirmed case data of current and past one time slots, the acceleration is calculated by the data of current and past two time slots. Hence, the data correlations of time domain (including two past time slots) are considered by the model. Especially, the acceleration clearly depicts whether the pandemic curve is becoming flattened, which can be an important metric for risk indexing. Intuitively, as new cases and acceleration have sufficiently represented the severity of the pandemic status, in order to limit the redundancy of the data, we did not incorporate data of earlier time slots in our model. Thus, there are a total of four disease-related attributes adopted.

For demographic attributes $\mathbf{a}_2$, we collect data from the United States Census Bureau [40]. The data set was updated on July 1, 2019, which includes population, population density, median age, age distribution, gender distribution, and persons per household (2014–2018) for each metropolitan area. The age distribution includes population for age intervals $[0, 4]$, $[5, 24]$, $[25, 44]$, $[45, 64]$, $[65, 84]$, and $[85, \infty)$. Thus, there are a total of 11 demographic attributes.

Therefore, there are a total of $4+11 = 15$ attributes used for model learning. Among these attributes, four of them, including population density, median age, gender distribution, and persons per household, are used as metadata. Thus, there are a total of $K = 11$ feature density maps built. Moreover, we collect coordinates of each metro from the "United States Cities Database," a data set built by Simplemaps using the authoritative sources, such as the U.S. Geological Survey and U.S. Census Bureau [41]. The coordinate data set is updated as of September 11, 2019.

B. Define Risk

As introduced in Section III-B, (1) is applied to aggregate features in $\boldsymbol{A}_R$. Note that each attribute in $\boldsymbol{A}_R$ is normalized into the interval of $[0, 1]$ before generating the risk index. The weight α of the seven attributes in $\boldsymbol{A}_R$ is set to $[0.2, 0.1, 0.2, 0.2, 0.1, 0.1, 0.1]$, which is corresponding to the attribute of confirmed case, death case, daily case, acceleration, population, population density, and median age,

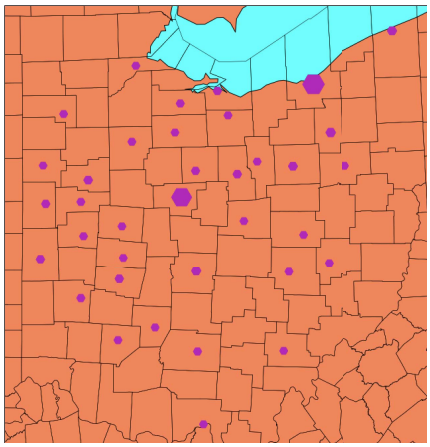


Fig. 8. Defined risk indices for metropolitan areas in Ohio state. The size of dot represents the risk index for the corresponding area.

respectively. The defined risk index for each metro is then normalized into the interval of [0, 10]. We show the generated risk indices for metropolitan areas as the date of July 29, 2020, in Fig. 8, where the size of the marker in the figure represents the defined risk index of the corresponding area. It can be easily noticed from the figure that there are two metropolitan areas having much higher risk levels than the rest. These two areas, respectively, correspond to Cleveland and Columbus metros in Ohio, which are the most crowded two metropolitan areas in Ohio. As of July 29, 2020, Cleveland and Columbus metropolitan areas have an accumulated 15 892 and 22 557 confirmed cases, respectively, which are the highest two in Ohio. Thus, it can be found that the defined risk indices properly indicate the severity level of infections in the corresponding areas.

C. MK-DNN Evaluation

As the risk indices of macrolevel areas, i.e., metros, are defined, we set up the MK-DNN model for training and testing. First, a KDE is set up for each attribute in both disease-related and demographic data sets. Then, we generate the training samples of DNN based on KDE's output and set up the DNN model for two-level modeling. Both regression and classification tasks are conducted in the macrolevel DNN training process. Moreover, we compare our MK-DNN model with other commonly used deep learning models. After macrolevel training, we apply our model to the microlevel and show the obtained risk hotspot maps.

1) KDE Setup: For each attribute, we build up a corresponding KDE function. As it is aforementioned, according to the attribute value in each metro, we first assign the corresponding number of nodes to the coordinate of the metro and then apply (2) to build up the corresponding feature density map. We adopt the Gaussian function as the kernel function. The bandwidth h, as a hyperparameter, needs to be properly set for optimal performance. Ideally, h is expected to be as small as possible, leading to a much more accurate estimated density over a certain area. However, as h decreases, fewer nodes on the map are incorporated to calculate the density, which leads the generated feature map less smooth. Different h are tested in the following simulation for seeking the optimal bandwidth. Note that the unit of distance between coordinates is transferred into radians for the simulation, where a radian of 0.01 approximately equals 63.6 km.

As the KDEs are built up, the training sample $\mathbf{x}_{\text{metro}} = [x^1_{\text{metro}}, \ldots, x^K_{\text{metro}}]$ for a certain metro of interest can be formulated.

2) DNN Setup: The DNN used in the simulation contains three hidden layers with 64, 128, and 32 nodes, respectively. The input layer has the size of 15 (11 extracted features + 4 metadata attributes), as aforementioned. The size of the output layer is dependent on different tasks. In the regression task, the output layer size is 1, which is a continuous risk index in [0, 10]. In the classification task, the task label is transferred to discrete 11 levels, $[0, 1, 2, \ldots, 10]$, where each level i is represented by a one-hot vector $\mathbf{y} = [y_1, \ldots, y_{11}], \mathbf{y} \in \Delta^{11}$, where $y_i = 1$. Thus, the size of the output is changed to 11 correspondingly. The activation function for the hidden layers in both regression and classification tasks is set to "relu." For the output layer in regression and classification tasks, the activation functions are "sigmoid" and "softmax," respectively. The regression task uses MSE as the loss function, while the other uses multiclass cross-entropy.

3) Two-Level Modeling: Different bandwidths h, including 0.001, 0.002, 0.003, 0.004, and 0.006, have been tested in this section. For both regression and classification tasks, the number of training epochs is 150. 80% of data is used for training, and 20% for testing.

Our model is run on a PC with 64-GB RAM and a GPU of NVIDIA RTX 3090 and implemented with the python deep learning library Keras. We show the performance of MK-DNN in Fig. 9. The first two rows in the figure are the training and testing performance of macrolevel modeling in MK-DNN with different h. The first row is the regression loss and the second is classification accuracy. The third row is the risk hotspot map generated by the microlevel modeling of the corresponding MK-DNN. For the regression loss, it can be easily found the performance of "$h = 0.001$" and "$h = 0.003$" are better than the rest, where the MSE of which converge at the value lower than 0.2. As the bandwidth h increases, both the train and test loss dramatically increase, especially in "$h = 0.014$," where the test loss even cannot converge after 150 epochs of training. The reason is that when h becomes larger, each KDE incorporates more nodes to calculate the feature density of a certain coordinate, which leads the estimated density distribution less informative. Hence, the DNN model fails to learn effectively. This also can be noticed easily in the microlevel prediction results, which is in the third row of Fig. 9.

Regarding the microlevel prediction, the regression model of MK-DNN is applied to the sampled coordinates in Ohio. In particular, we chop the map into a grid with each grid size of 0.05 degrees (0.00087 of radian). For each grid, we implement the microlevel MK-DNN to obtain the risk indices. From the figure, we observe that when the bandwidth of KDE is small, the trained DNN becomes relatively rigid and only marks the metropolitan areas as high risk. As the bandwidth increases, the DNN marks certain microlevel areas as high risk. However,

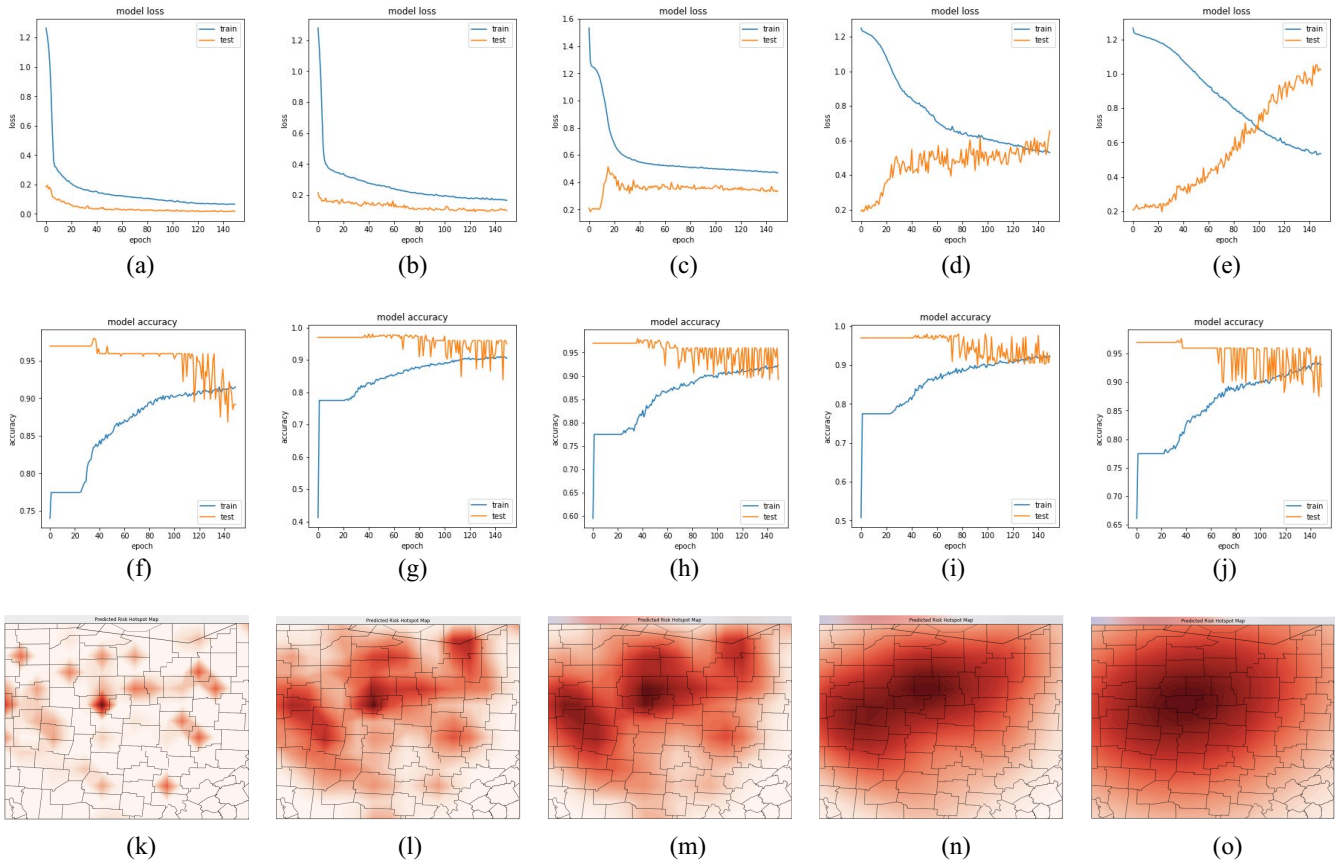


Fig. 9. Performance of MK-DNN. The first two rows are the learning results of regression and classification tasks, respectively. The third row is the risk hotspot maps obtained by the microlevel modeling. (a) Regression loss, $h = 0.001$. (b) Regression loss, $h = 0.003$. (c) Regression loss, $h = 0.006$. (d) Regression loss, $h = 0.01$. (e) Regression loss, $h = 0.014$. (f) Classification accuracy, $h = 0.001$. (g) Classification accuracy, $h = 0.003$. (h) Classification accuracy, $h = 0.006$. (i) Classification accuracy, $h = 0.01$. (j) Classification accuracy, $h = 0.014$. (k) Predicted risk hotspot, $h = 0.001$. (l) Predicted risk hotspot, $h = 0.003$. (m) Predicted risk hotspot, $h = 0.006$. (n) Predicted risk hotspot, $h = 0.01$. (o) Predicted risk hotspot, $h = 0.014$.

as the bandwidth becomes too large, as shown in Fig. 9(n) and (o), it is hard to differentiate the risks between areas, which is because the KDE improperly incorporates a much larger area to calculate feature density.

D. Validation

After the MK-DNN is trained, we further conduct our validation simulation to validate both macro and microlevel obtained risk indices. In macrolevel validation, we first train the validation network by inputting the obtained risk and demographic attributes to validation network. The label of the input samples are the corresponding confirmed case number. Note that the validation network is trained based on the data from macrolevel areas, i.e., metros with disease-related data. Besides, the risk indices used in the macrolevel validation network training is the output of the regression model of macrolevel MK-DNN. Here, we, respectively, use bandwidths of $0.001, 0.003$, and 0.006 to test the training performance and the validation score for macrolevel validation. In the following microlevel validation, we, respectively, apply three trained validation networks to further validate the risk obtained by microlevel MK-DNN with the corresponding bandwidth. Similar as before, $36(\text{metros}) \times 190(\text{timeslots}) = 6840$ data items are formed for validation network training and testing. The ratio of training and testing is 80%:20%. The loss function is set to MSE.

Fig. 10 shows the training and testing loss of the validation networks and the corresponding validation scores. After 500 epochs of training, the test loss and the corresponding validation score are as follows. When $h = 0.001$, the loss and score converge at 89.46 and 0.82, respectively. When $h = 0.003$, they converge at 50.62 and 0.87. Moreover, they converge at 55.58 and 0.85, when $h = 0.006$. It should be noticed that the input and the output of the validation network are actually the output and input of the MK-DNN model, respectively. Thus, it is reasonable to see that the obtained risk is highly related to the confirmed case number because the MK-DNN is fully trained for risk prediction. Furthermore, we compared our model with other existing methods in Table I. In particular, we show the validation scores of three most significant metropolitan areas in Ohio, which are Cleveland, Columbus, and Akron, along with the average results of all 36 metros in Ohio. We adopt risk indices defined in [16] for comparison, which is called α-Satellite risk assessment. In particular, we change our MK-DNN's obtained risk indices to risk indices given by α-Satellite and keep the rest setup the same for a fair comparison. It can be noticed that our obtained risk indices and α-Satellite obtain similar validation scores by using the same MK-DNN, which indicates both our obtained risk indices

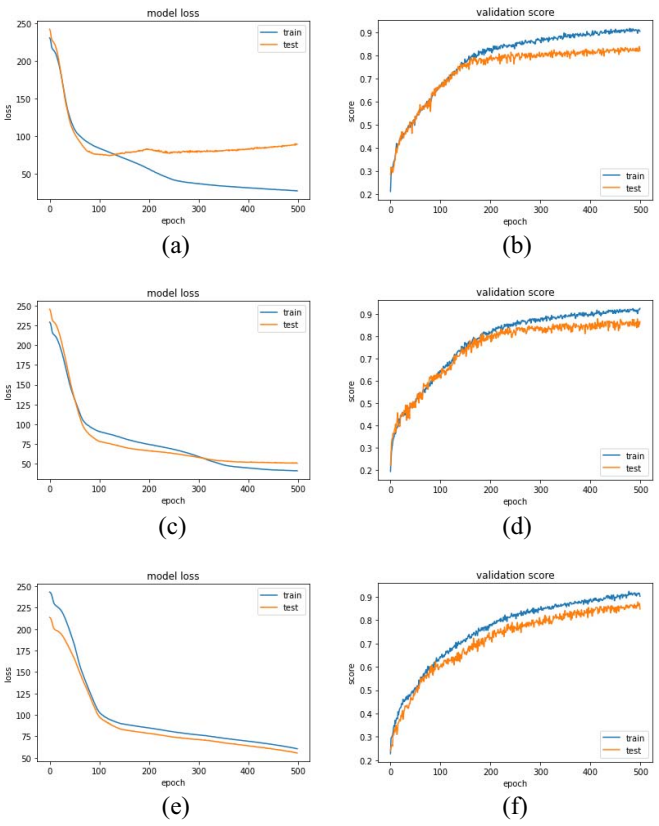


Fig. 10. Results of macrolevel validation. (a) Loss of the macrolevel validation network, $h = 0.001$. (b) Macrolevel validation score, $h = 0.001$. (c) Loss of the macrolevel validation network, $h = 0.003$. (d) Macrolevel validation score, $h = 0.003$. (e) Loss of the macrolevel validation network, $h = 0.006$. (f) Macrolevel validation score, $h = 0.006$.

and α-Satellite are informative and highly related to confirmed case. Moreover, we compare the performance of our MK-DNN with regular DNN, and logistic regression model. It also can be observed that the performance of regular DNN and logistic regression model significantly decays compared with the MK-DNN, because the MK-DNN extracts more features by applying MKDE. In the next step, we further extend the validation to the microlevel, in which there are no true data of confirmed case. Considering the macrolevel validation score as a benchmark, we check the decay on the microlevel validation score to evaluate the validity of MK-DNN microlevel output. The decay of the performance can further help evaluate the bandwidth setup in the MK-DNN model.

In the microlevel validation, we first chop a metro (macrolevel area) into grids with the same setup in MK-DNN microlevel modeling and apply the microlevel MK-DNN to obtain risk index for each grid. The validation network trained in macrolevel validation is applied on each grid to predict the corresponding confirmed case. We also compare our method with α-Satellite in this section. As there are microlevel areas in microlevel, α-Satellite may have missing risk values for certain areas. We complement them with the corresponding macrolevel area's risk given by α-Satellite. The results are shown in Table I. It can be found $h = 0.003$ has the highest score of 0.8518 compared with others, which is fairly

TABLE I
MACROLEVEL AND MICROLEVEL VALIDATION SCORE FOR THREE MAJOR METROS IN OHIO STATE AND THE AVERAGE RESULT OF ALL METROS

h	Method	Cleveland	Columbus	Akron	Average
Macro Level Validation					
0.001	①	0.8490	0.8062	0.8298	0.8213
0.001	②	0.8364	0.8155	0.8301	0.8318
0.003	①	0.8712	**0.8563**	0.8608	**0.8732**
0.003	②	**0.8766**	0.8477	**0.8842**	0.8711
0.006	①	0.8713	0.8338	0.8605	0.8564
0.006	②	0.8610	0.8406	0.8696	0.8613
N/A	③	0.8295	0.8091	0.8194	0.8189
N/A	④	0.7868	0.7559	0.7502	0.7749
Micro Level Validation					
0.001	①	0.8023	0.7895	0.8061	0.7926
0.001	②	0.7319	0.7287	0.7521	0.7358
0.003	①	**0.8561**	0.8432	**0.8925**	**0.8518**
0.003	②	0.7319	0.7287	0.7521	0.7358
0.006	①	0.8303	**0.8522**	0.8188	0.8289
0.006	②	0.7853	0.8249	0.7614	0.7867

①: Our risk definition + MK-DNN
②: Risk indices defined in [16] + MK-DNN
③: Predicted Risk + DNN
④: Predicted Risk + Logistic Regression

close to the macrolevel validation score of 0.8732. As the comparison, the scores of α-Satellite significantly decays as there are missing values in some microlevel areas. In other words, simply applying macrolevel risk indices to microlevel may not be accurate and lead to the decrease of the validation score. Moreover, when h becomes lower or higher, the validation score decreases with varying degrees. As discussed in the simulation of microlevel modeling, an excessively low or high h will lead the KDE function to become too rigid or smooth, respectively. Thus, the microlevel validation score not only validates the microlevel MK-DNN obtained risk indices but also helps select 0.003 as the best bandwidth in the MK-DNN model among all the choices.

So far, in the simulation, we first show the visualization results of our defined risk indices. Then, we present the learning performance of our MK-DNN model, which demonstrates the model can accurately predict the risk index for the macrolevel areas and further obtain risk indices for microlevel areas. From the learning performance of MK-DNN, it can be concluded that the MKDE-extracted features are informative and based on which the model can be trained to output effective results. In the validation, we compare our model with a state-of-the-art existing risk assessment model and other well-known learning models. From the comparison, we first demonstrate the MK-DNN outperforms the regular DNN and logistic regression models and then observe that our system can effectively obtain more informative risk indices for microlevel areas compared with the existing risk assessment model.

V. CONCLUSION

In this work, we proposed a risk assessment system to help IoMT applications combat COVID-19 based on MKDE and

DNN, which is called MK-DNN. Considering the limitation in the public COVID-19 data set, multiple authoritative data types are acquired and utilized in the system. A two-level modeling was designed to enable the MK-DNN effectively obtain risk indices on a more fine-grained map, especially for the area that lacks disease-related data. Furthermore, since the MK-DNN cannot be trained in the microlevel due to the lack of disease-related data, a heuristic risk validation method was proposed, which is to evaluate the accumulated error in microlevel outputs through the designed validation score functions and further help optimize the hyperparameter in MK-DNN. Simulation on the real-world data shows the accuracy and validity of our MK-DNN risk assessment system.

REFERENCES

[1] W.-J. Guan *et al.*, "Clinical characteristics of coronavirus disease 2019 in China," *New England J. Med.*, vol. 382, no. 18, pp. 1708–1720, 2020.

[2] C. Wang, P. W. Horby, F. G. Hayden, and G. F. Gao, "A novel coronavirus outbreak of global health concern," *Lancet*, vol. 395, no. 10223, pp. 470–473, 2020.

[3] M. Otoom, N. Otoum, M. A. Alzubaidi, Y. Etoom, and R. Banihani, "An IoT-based framework for early identification and monitoring of COVID-19 cases," *Biomed. Signal Process. Control*, vol. 62, Sep. 2020, Art. no. 102149.

[4] A. H. M. Aman, W. H. Hassan, S. Sameen, Z. S. Attarbashi, M. Alizadeh, and L. A. Latiff, "IoMT amid COVID-19 pandemic: Application, architecture, technology, and security," *J. Netw. Comput. Appl.*, vol. 117, Jan. 2021, Art. no. 102886.

[5] G. J. Joyia, R. M. Liaqat, A. Farooq, and S. Rehman, "Internet of Medical Things (IoMT): Applications, benefits and future challenges in healthcare domain," *J. Commun.*, vol. 12, no. 4, pp. 240–247, 2017.

[6] B. Lin and S. Wu, "COVID-19 (coronavirus disease 2019): Opportunities and challenges for digital health and the Internet of Medical Things in China," *OMICS J. Integr. Biol.*, vol. 24, no. 5, pp. 231–232, 2020.

[7] Y. Guo, T. Ji, Q. Wang, L. Yu, G. Min, and P. Li, "Unsupervised anomaly detection in IoT systems for smart cities," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2231–2242, Oct.–Dec. 2020.

[8] T. Yang, M. Gentile, C.-F. Shen, and C.-M. Cheng, "Combining point-of-care diagnostics and Internet of Medical Things (IoMT) to combat the COVID-19 pandemic," *Diagnostics*, vol. 10, no. 4, p. 224, 2020.

[9] The New York Times. (Mar. 2020). *Coronavirus in the U.S.: Latest Map and Case Count*. [Online]. Available: https://www.nytimes.com/interactive/2020/us/coronavirus-us-cases.html

[10] China Data Lab. (Dec. 2020). *US COVID-19 Daily Cases With Basemap*. [Online]. Available: https://doi.org/10.7910/DVN/HIDLTK

[11] *COVID-19 Data Collection*. Accessed: Dec. 17, 2020. [Online]. Available: https://dataverse.harvard.edu/dataverse/covid19

[12] T. Chakraborty and I. Ghosh, "Real-time forecasts and risk assessment of novel coronavirus (COVID-19) cases: A data-driven analysis," *Chaos Solitons Fractals*, vol. 135, Jun. 2020, Art. no. 109850.

[13] S. W. Hermanowicz. (2020). *Forecasting the Wuhan Coronavirus (2019-nCoV) Epidemics Using a Simple (Simplistic) Model*. [Online]. Available: https://doi.org/10.1101/2020.02.04.20020461

[14] Z. Hu, Q. Ge, L. Jin, and M. Xiong, "Artificial intelligence forecasting of COVID-19 in China," 2020. [Online]. Available: arXiv:2002.07112.

[15] K. Jahanbin and V. Rahmanian, "Using Twitter and Web news mining to predict COVID-19 outbreak," *Asian Pac. J. Tropical Med.*, vol. 13, no. 8, pp. 378–380, 2020.

[16] Y. Ye *et al.*, "α-satellite: An AI-driven system and benchmark datasets for hierarchical community-level risk assessment to help combat COVID-19," 2020. [Online]. Available: arXiv:2003.12232.

[17] J. Chen *et al.*, "Deep learning-based model for detecting 2019 novel coronavirus pneumonia on high-resolution computed tomography," *Sci. Rep.*, vol. 10, no. 1, pp. 1–11, 2020.

[18] Y. Song *et al.*, "Deep learning enables accurate diagnosis of novel coronavirus (covid-19) with CT images," *IEEE/ACM Trans. Comput. Biol. Bioinformat.*, early access, Mar. 11, 2021, doi: 10.1109/TCBB.2021.3065361.

[19] S. Wang *et al.*, "A deep learning algorithm using ct images to screen for corona virus disease (covid-19)," *Eur. Radiol.*, to be published.

[20] G. S. Randhawa, M. P. Soltysiak, H. El Roz, C. P. de Souza, K. A. Hill, and L. Kari, "Machine learning using intrinsic genomic signatures for rapid classification of novel pathogens: COVID-19 case study," *PLoS ONE*, vol. 15, no. 4, 2020, Art. no. e0232391.

[21] X. Xu *et al.*, "A deep learning system to screen novel coronavirus disease 2019 pneumonia," *Engineering*, vol. 6, no. 10, pp. 1122–1129, 2020.

[22] A. S. S. Rao and J. A. Vazquez, "Identification of COVID-19 can be quicker through artificial intelligence framework using a mobile phone–based survey when cities and towns are under quarantine," *Infect. Control Hospital Epidemiol.*, vol. 41, no. 7, pp. 826–830, 2020.

[23] W. Shi *et al.*, "Deep learning-based quantitative computed tomography model in predicting the severity of COVID-19: A retrospective study in 196 patients," *Ann. Transl. Med.*, vol. 9, no. 3, p. 216, 2021.

[24] L. Yan *et al.* (2020). *Prediction of Survival for Severe COVID-19 Patients With Three Clinical Features: Development of a Machine Learning-Based Prognostic Model With Clinical Data in Wuhan*. [Online]. Available: https://doi.org/10.1101/2020.02.27.20028027

[25] P. Wang, X. Zheng, J. Li, and B. Zhu, "Prediction of epidemic trends in COVID-19 with logistic model and machine learning technics," *Chaos Solitons Fractals*, vol. 139, Oct. 2020, Art. no. 110058.

[26] M. S. Majumder and K. D. Mandl. *Early Transmissibility Assessment of a Novel Coronavirus in Wuhan, China*. Accessed: Jan. 26, 2020. [Online]. Available: https://ssrn.com/abstract=3524675

[27] L. Wang *et al.*, "An epidemiological forecast model and software assessing interventions on the covid-19 epidemic in China," *J. Data Sci.*, vol. 18, no. 3, pp. 409–432, 2020.

[28] H. Zhu *et al.* (2020). *Host and Infectivity Prediction of Wuhan 2019 Novel Coronavirus Using Deep Learning Algorithm*. [Online]. Available: https://doi.org/10.1101/2020.01.21.914044

[29] W. C. Roda, M. B. Varughese, D. Han, and M. Y. Li, "Why is it difficult to accurately predict the COVID-19 epidemic?" *Infect. Disease Model.*, vol. 5, pp. 271–281, 2020. [Online]. Available: https://doi.org/10.1016/j.idm.2020.03.001

[30] M. M. Sajadi, P. Habibzadeh, A. Vintzileos, S. Shokouhi, F. Miralles-Wilhelm, and A. Amoroso. (2020). *Temperature and Latitude Analysis to Predict Potential Spread and Seasonality for COVID-19*. [Online]. Available: https://ssrn.com/abstract=3550308

[31] *Cases in the U.S.* Accessed: May 11, 2020. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/cases-updates/cases-in-us.html

[32] *WHO Coronavirus (COVID-19) Dashboard*. Accessed: May 11, 2020. [Online]. Available: https://covid19.who.int/table/

[33] C. Luo, J. Ji, Q. Wang, X. Chen, and P. Li, "Channel state information prediction for 5G wireless communications: A deep learning approach," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 1, pp. 227–236, Jan.–Mar. 2020.

[34] L. Yu, Q. Wang, Y. Guo, and P. Li, "Spectrum availability prediction in cognitive aerospace communications: A deep learning perspective," in *Proc. Cogn. Commun. Aerosp. Appl. Workshop (CCAA)*, 2017, pp. 1–4.

[35] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[36] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*, vol. 26. London, U.K.: Chapman & Hall/CRC, 1986,

[37] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, Apr. 2017.

[38] China Data Lab. (2020). *US Metropolitan Daily Cases With Basemap*. [Online]. Available: https://doi.org/10.7910/DVN/5B8YM8

[39] T. Hu *et al.*, "Building an open resources repository for covid-19 research," *Data Inf. Manag.*, vol. 4, pp. 130–147, 2020.

[40] *U.S. Census Bureau Quickfacts: Ohio*. Accessed: May 15, 2020. [Online]. Available: https://www.census.gov/quickfacts/OH

[41] *United States Cities Database*. Accessed: May 15, 2020. [Online]. Available: https://simplemaps.com/data/us-cities

[42] R. Chen *et al.*, "Covid-19 vulnerability map construction via location privacy preserving mobile crowdsourcing," *in Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2020, pp. 1–6, doi: 10.1109/GLOBECOM42002.2020.9348141.

[43] J. Chen, R. Chen, X. Zhang, and M. Pan, "A privacy preserving federated learning framework for COVID-19 vulnerability map construction," *Proc. IEEE Int. Conf. Commun. (ICC)*, Montreal, QC, Canada, Jun. 2021.

Qianlong Wang (Graduate Student Member, IEEE) received the B.E. degree in electrical engineering from Wuhan University, Wuhan, China, in 2013, the M.E. degree in electrical and computer engineering from Stevens Institute of Technology, Hoboken, NJ, USA, in 2015, and the Ph.D. degree in computer engineering from Case Western Reserve University, Cleveland, OH, USA, in 2021.

He is currently an Assistant Professor with the Department of Computer and Information Sciences, Towson University, Towson, MD, USA. His research interests include robust and secure deep learning network, security and privacy in distributed and decentralized system, e.g., cyber–physical system and Internet of Things, and related applications, e.g., smart transportation, smart city, and smart healthcare.

Yifan Guo (Graduate Student Member, IEEE) received the B.S. degree in information and computing sciences from the Beijing University of Posts and Telecommunications, Beijing, China, in 2013, and the M.S. degree in computer science from Northwestern University, Evanston, IL, USA, in 2015. He is currently pursuing the Ph.D. degree with the Department of Electrical Engineering and Computer Science, Case Western Reserve University, Cleveland, OH, USA.

His research interests include artificial intelligence, machine learning, and big data.

Tianxi Ji (Graduate Student Member, IEEE) received the B.E. degree in telecommunication engineering from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2014, and the M.S. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 2016. He is currently pursuing the Ph.D. degree with the Department of Electrical Engineering and Computer Science, Case Western Reserve University, Cleveland, OH, USA.

His research interests include big data, edge computing, as well as security and privacy.

Xufei Wang (Graduate Student Member, IEEE) received the B.E. degree in Internet of Things from the Huazhong University of Science and Technology, Wuhan, China, in 2016. He is currently pursuing the Ph.D. degree with the Department of Electrical Engineering and Computer Science, Case Western Reserve University, Cleveland, OH, USA.

His research interests include optimization in big data applications, networking in cyber–physical systems, and Internet-of-Things-related applications, e.g., smart healthcare and smart transportation.

Bingfang Hu received the Ph.D. degree in pharmacogenetics from Sun Yat-sen University, Guangzhou, China, in 2015.

Her research interests include big data, biostatistics, pharmacology, and pharmacogenetics.

Pan Li (Member, IEEE) is currently an Associate Professor with the Department of Electrical, Computer, and Systems Engineering, Case Western Reserve University, Cleveland, OH, USA. His research interests include big data analytics and computing, security and privacy, and network science and economics.

Dr. Li received the NSF CAREER Award in 2012. He has served as an Editor for IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE WIRELESS COMMUNICATIONS LETTERS, IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS–Cognitive Radio Series, and IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, and a Feature Editor for IEEE WIRELESS COMMUNICATIONS.