Examining Deep Learning Models with Multiple Data Sources for COVID-19 Forecasting

Lijing Wang*†, Aniruddha Adiga†, Srinivasan Venkatramanan†, Jiangzhuo Chen†, Bryan Lewis† and Madhav Marathe*†

*Computer Science, University of Virginia, Charlottesville, VA 22903

†Biocomplexity Institute and Initiative, University of Virginia, Charlottesville, VA 22903

Email: {lw8bn,aa5dw,srini,chenj,brylew,marathe}@virginia.edu

Abstract—**The COVID-19 pandemic represents the most significant public health disaster since the 1918 influenza pandemic. During pandemics such as COVID-19, timely and reliable spatio-temporal forecasting of epidemic dynamics is crucial. Deep learning-based time series models for forecasting have recently gained popularity and have been successfully used for epidemic forecasting. Here we focus on the design and analysis of deep learning-based models for COVID-19 forecasting. We implement multiple recurrent neural network-based deep learning models and combine them using the stacking ensemble technique. In order to incorporate the effects of multiple factors in COVID-19 spread, we consider multiple sources such as COVID-19 confirmed and death case count data and testing data for better predictions. To overcome the sparsity of training data and to address the dynamic correlation of the disease, we propose clustering-based training for high-resolution forecasting. The methods help us to identify the similar trends of certain groups of regions due to various spatio-temporal effects. We examine the proposed method for forecasting weekly COVID-19 new confirmed cases at county-, state-, and country-level. A comprehensive comparison between different time series models in COVID-19 context is conducted and analyzed. The results show that simple deep learning models can achieve comparable or better performance when compared with more complicated models. We are currently integrating our methods as a part of our weekly forecasts that we provide state and federal authorities.**

I. INTRODUCTION

The COVID-19 pandemic is the worst outbreak we have seen since 1918; it has caused over 22 million confirmed cases globally and over 791,000 deaths in more than 200 countries as of August 26, 2020 [1]. The economic impact is equally staggering, estimates suggest an overall impact of 86.6 trillion U.S. dollars on the global GDP [2]. One effective way to control epidemics is to forecast the epidemic trajectory – a good and reliable forecast can help in planning and response operations. Two popular methods for forecasting COVID-19 dynamics are statistical time series models and compartmental mass action models at varying spatio-temporal scales [3], [4], [5], [6], [7], [8], [9]. There is also recent work on use of DNN and other ML techniques to forecast COVID-19 outbreak [10], [11]. These methods can make multi-fidelity predictions based on the model resolution. The models can often capture human decision making and thus provide a path for counterfactual forecasts. The Centers for Disease Control and Prevention (CDC) COVID-19 forecasting project shows that only one out of 36 teams is using deep learning-based methods for making projections of cumulative and incident deaths and incident hospitalizations due to COVID-19 in the United States [12] as of August 10, 2020. The primary challenge for these methods is the lack of training data. Other efforts focus on time series-based methodologies to learn patterns in historical epidemic data and other exogenous factors and leverage those patterns for forecasting [13], [14], [15], [16], [17], [18]. See [19], [20], [21], [22], [23], [24] for use of DNNs to forecast epidemic dynamics more broadly.

Our contributions. Our work focuses on exploring *deep learning*-based methods that incorporate *multiple sources* for *weekly 4 weeks ahead* forecasting of COVID-19 new confirmed cases at *multiple geographical resolutions* including country-, state-, and county-level. In the context of COVID-19, the problem is more complicated than seasonal influenza forecasting for the following reasons: (i) very sparse training data for each region; (ii) noisy surveillance data due to heterogeneity in epidemiological context e.g. disease spreading timeline and testing prevalence in different regions, (iii) system is constantly in churn – individual behavioral adaptation, policies and disease dynamics are constantly co-evolving. Given these challenges, we examine different types of time series models and propose an ensemble framework that combines simple deep learning models using multiple sources such as COVID-19 cases data and testing data. The multi-source data allows us to capture the above mentioned factors more effectively. To overcome the data sparsity problem we propose clustering-based training methods to augment training data for each region. We group spatial regions based on trend similarity and infer a model per cluster. Among other things this avoids overfitting due to sparse training data. As an additional benefit it aids in explicitly uncovering the spatial correlation across regions by training models with similar time series. Our main contributions are summarized below:

- First, we systematically examine time series-based deep learning models for COVID-19 forecasting and propose clustering-based training methods to augment sparse and noisy training data for high resolution regions which can avoid overfitting and explicitly uncover the similar spreading trends of certain groups of regions.
- Second, we implement a stacking ensemble framework

to combine multiple deep learning models and multiple sources for better performance. Stacking is a natural way to combine multiple methods and data sources.

- Third, we analyze the performance of our method and other published results in their ability to forecast weekly new confirmed cases at country, state, and county level. The results show that our ensemble model outperforms any individual models as well as several classic machine learning and state-of-the-art deep learning models;
- Finally, we conduct a comprehensive comparison among mechanistic models, statistical models and deep learning models. The analysis shows that for COVID-19 forecasting deep learning-based models can capture the dynamics and have better generalization capability as opposed to the mechanistic and statistical baselines. Simple deep learning models such as simple recurrent neural networks can achieve better performance than complex deep learning models like graph neural networks for high resolution forecasting.

II. Related work

COVID-19 is a very active area of research and thus it is impossible to cover all the recent manuscripts. We thus only cover important papers here.

A. COVID-19 forecasting by mechanistic methods

Mechanistic methods have been a mainstay for COVID-19 forecasting due to their capability of represent the underlying disease transmission dynamics as well as incorporating diverse interventions. They enable counterfactual forecasting which is important for future government interventions to control the spread. Forecasting performance depends on the assumed underlying disease model. Yang et al. [6] use a modified susceptible(S)-exposed(E)-infected(I)-recovered(R) (SEIR) model for predicting the COVID-19 epidemic peaks and sizes in China. Anastassopoulou et al. [3] provide estimations of the basic reproduction number and the per day infection mortality and recovery rates using an susceptible(S)-infected(I)-dead(D)-recovered(R) (SIDR) model. Giordano et al. [4] propose a new susceptible(S)-infected(I)-diagnosed(D)-ailing(A)-recognized(R)-threatened(T)-healed(H)-extinct(E) (SIDARTHE) model to help plan an effective control strategy. Yamana et al. [5] use a metapopulation SEIR model for US county resolution forecasting. Chang et al. [8] develop an agent-based model for a fine-grained computational simulation of the ongoing COVID-19 pandemic in Australia. Kai et al. [7] present a stochastic dynamic network-based compartmental SEIR model and an individual agent-based model to investigate the impact of universal face mask wearing upon the spread of COVID-19.

B. COVID-19 forecasting by time series models

Time series models, such as statistical models and deep learning models, are popular for their simplicity and forecasting accuracy in the epidemic domain. One big challenge is the lack of sufficient training data in the context of COVID-19 dynamics. Another challenge is that the surveillance data is extremely noisy (hard to model noise) due to rapidly evolving epidemics. However, additional data becomes available and the surveillance systems mature these models become more promising. Harvey et al. [13] propose a new class of time series models based on generalized logistic growth curves that reflect COVID-19 trajectories. Petropoulos et al. [14] produce forecasts using models from the exponential smoothing family. Ribeiro et al. [15] evaluate multiple regression models and stacking-ensemble learning for COVID-19 cumulative confirmed cases forecasting with one, three, and six days ahead in ten Brazilian states. Hu et al. [16] propose a modified auto-encoder model for real-time forecasting of the size, lengths and ending time in China. Chimmula et al. [17] use LSTM networks to predict COVID-19 transmission. Arora et al. [18] use LSTM-based models for positive reported cases for 32 states and union territories of India. Magri et al. [10] propose a data-driven model trained with both data and first principles. Dandekar et al. [11] use neural network aided quarantine control models to estimate the global COVID-19 spread.

C. Deep learning-based epidemic forecasting

Recurrent neural networks (RNN) has been demonstrated to be able to capture dynamic temporal behavior of a time sequence. Thus it has become a popular method in recent years for seasonal influenza-like-illness (ILI) forecasting. Volkova et al. [19] build an LSTM model for short-term ILI forecasting using CDC ILI and Twitter data. Venna et al. [25] propose an LSTM-based method that integrates the impacts of climatic factors and geographical proximity. Wu et al. [20] construct CNNRNN-Res combining RNN and convolutional neural networks to fuse information from different sources. Wang et al. [21], [24] propose TDEFSI combining deep learning models with casual SEIR models to enable high-resolution ILI forecasting with no or less high-resolution training data. Adhikari et al. [22] propose EpiDeep for seasonal ILI forecasting by learning meaningful representations of incidence curves in a continuous feature space. Deng et al. [23] design cola-GNN which is a cross-location attention-based graph neural network for forecasting ILI. Regarding COVID-19 forecasting, Amol et al. [26] examined a novel forecasting approach for COVID-19 daily case prediction that uses graph neural networks and mobility data. Gao et al. [27] proposed STAN that uses a spatio-temporal attention network. Aamchandani et al. [28] presented DeepCOVIDNet to compute equidimensional representations of multivariate time series. These works examine their models on daily forecasting for US state or county levels.

Our work focuses on time series deep learning models for COVID-19 forecasting that yield weekly forecast at multiple resolution scales and provide 4 weeks ahead forecasts (equal to 28 days ahead in the context of daily forecasting). We use an ensemble model to combine multiple simple deep learning models. We show that compared to state-of-the-art time series models, simple recurrent neural network-based models can achieve better performance. More importantly, we show that

the ensemble method is an effective way to mitigate model overfitting caused by the super small and noisy training data.

III. METHOD

A. Problem Formulation

We formulate the COVID-19 new confirmed cases forecasting problem as a regression task with time series of multiple sources as the input. We have N regions in total. Each region is associated with a time series of multi-source input in a time window T. For a region r, at time step t, the multi-source input is denoted as $\mathbf{x}_{r,t} \in \mathbb{R}^S$ where S is the feature numbers. We denote the training data as $\mathbf{X}_{r,t} = [\mathbf{x}_{r,t-T+1}, ..., \mathbf{x}_{r,t}] \in \mathbb{R}^{T \times S}$. The objective is to predict COVID-19 new confirmed cases at a future time point $t + h$ where h refers to the horizon of the prediction. We are interested in a predictor f that predicts new confirmed case count at time $t + h$, denoted as $\mathbf{z}_{r,t+h}$, by taking $\mathbf{X}_{r,t}$ as the input where t is the most recent time of data availability.

$$\hat{\mathbf{z}}_{r,t+h} = f(\mathbf{X}_{r,t}, \theta) \qquad (1)$$

where θ denotes parameters of the predictor and $\hat{\mathbf{z}}_{r,t+h}$ denotes the prediction of $\mathbf{z}_{r,t+h}$.

B. Recurrent Neural Networks (RNNs)

For brevity, we assume a region is given, thus we omit subscript r in this subsection. An RNN model consists of k-stacked RNN layers. Each RNN layer consists of T cells, denoted as $\langle cell_{t-T+1}, \cdots, cell_t \rangle$. The input is $\mathbf{X}_t$, the output from the last layer k is denoted as $\mathbf{h}^{(k)}$. Let $H^{(i)}, 1 \leq i \leq k$ be the dimension of the hidden state in $layer_i$. For the first layer $layer_1$, $cell_t$ will work as:

$$\mathbf{h}_t^{(1)} = \tanh(\mathbf{W}_i^{(1)} \cdot \mathbf{x}_t + \mathbf{U}_i^{(1)} \cdot \mathbf{h}_{t-1}^{(1)} + \mathbf{b}_i^{(1)}) \in \mathbb{R}^{H^{(1)}} \quad (2)$$

where $\tanh$ is activation function; $\mathbf{W} \in \mathbb{R}^{H^{(1)} \times S}, \mathbf{U} \in \mathbb{R}^{H^{(1)} \times H^{(1)}}$, and $\mathbf{b} \in \mathbb{R}^{H^{(1)}}$ are learned weights and bias; $\mathbf{h}_t^{(1)}$ is the output of $cell_t$ and $\mathbf{h}_{t-1}^{(1)}$ is from $cell_{t-1}$. The cell computation is similar in the $layer_i$, but with $\mathbf{x}_t$ being replaced by $\mathbf{h}_t^{(i-1)} \in \mathbb{R}^{H^{(i-1)}}$, and $\mathbf{W} \in \mathbb{R}^{H^{(i)} \times H^{(i-1)}}$. The first RNN layer takes $\mathbf{x}_{t-T+1}, \cdots, \mathbf{x}_T$ as the input, the second layer takes $\mathbf{h}_{t-T+1}^{(1)}, ..., \mathbf{h}_t^{(1)}$ as the input, and the rest of the layers behave in the same manner. The RNN module can be replaced by Gated Recurrent Unit (GRU) [29] or Long Short-term Memory (LSTM) [30] which avoid short-term memory and gradient vanishing problems of vanilla RNNs.

The output of the k-stacked RNN layers is fed into a fully connected layer:

$$\hat{\mathbf{z}}_t = \psi(\mathbf{w} \cdot \mathbf{h}_t^{(k)} + \mathbf{b}) \in \mathbb{R}^H \qquad (3)$$

where H is the output dimension, $\mathbf{w} \in \mathbb{R}^{H \times H^{(k)}}$, $\mathbf{b} \in \mathbb{R}^H$, and ψ is a linear function.

C. Multi-source Attention RNNs

The Multi-source attention RNN model consists of m k-stacked RNN models, each of which encodes a time series of one feature. Assume the output of branch r is $\mathbf{h}^r \in \mathbb{R}^{H_r}$ in which we omit subscript t for brevity. An attention layer is used to measure the impact of multi-source on new confirmed cases. We assume the time series of new confirmed cases is encoded in branch r, and we define attention coefficient a_j as the effect of feature j on target feature:

$$a_j = \psi(\mathbf{w}_r^T \cdot \mathbf{h}^r + \mathbf{w}_j^T \cdot \mathbf{h}^j + b_j) \in \mathbb{R} \qquad (4)$$

where $\mathbf{w}_r \in \mathbb{R}^{H_r}$, ψ is RELU function. Then the output of attention layer is:

$$\mathbf{h}^a = \psi(\mathbf{w}_a \sum_{j=1}^m a_j \cdot \mathbf{h}^r + \mathbf{b}_a) \in \mathbb{R}^{H_a} \qquad (5)$$

where $\mathbf{w}_a \in \mathbb{R}^{H_a \times H_r}$, $\mathbf{b}_a \in \mathbb{R}^{H_a}$, ψ is the tanh function. The output layer is a dense layer that outputs $\hat{\mathbf{z}}_t$:

$$\hat{\mathbf{z}}_t = \psi(\mathbf{w}_o \cdot \mathbf{h}^a + \mathbf{b}_o) \in \mathbb{R}^H \qquad (6)$$

where $\mathbf{w}_o \in \mathbb{R}^{H \times H_a}$, $\mathbf{b} \in \mathbb{R}^H$, ψ is the linear function. In our paper, all the features have the same length of time series. However, the multi-source attention RNN model enables training with the input that has a different length of time series of the features, which is superior in heterogeneous availability of multiple factors.

D. Clustering-based Training

Deep learning models usually require a large amount of training data which is not the case in the context of COVID-19. Particularly, for regions where the pandemic starts late, there are only a few valid data points for weekly forecasting. Thus training a single model for each such region, which we call *vanilla* training, is highly susceptible to overfitting. One modeling strategy is to train a model for a group of selected regions which to some extent overcomes the data sparsity problem. It is more likely that groups of regions exhibit strong correlations due to various spatio-temporal effects and geographical or demographic similarity. We explore a clustering-based approach that simultaneously learns COVID-19 dynamics from multiple regions within the cluster and infers a model per cluster. Various types of similarity metrics can be used to uncover the trend similarity allowing for an explainable time series forecasting framework.

Generalizing the earlier problem formulation, we denote the historical available time series for a region r as $\mathbf{X}_r = [\mathbf{x}_{r,1}, ..., \mathbf{x}_{r,T_r}] \in \mathbb{R}^{T_r \times S}$ where T_r is the time span of the available surveillance data. T_r is increasing as new data becomes available and it varies across different regions. The set of time series for N regions is denoted as $\mathcal{X} = \{\mathbf{X}_r | r = 1, \cdots, N\}$. The clustering process aims to partition the $\mathcal{X}$ into $k(\leq N)$ sets $\mathcal{C} = \{C_1, ..., C_k\}$.

In our work, the trend is represented as the time series of new confirmed cases and we cluster the time series in two ways – geography-based clustering (*geo-clustering*) and
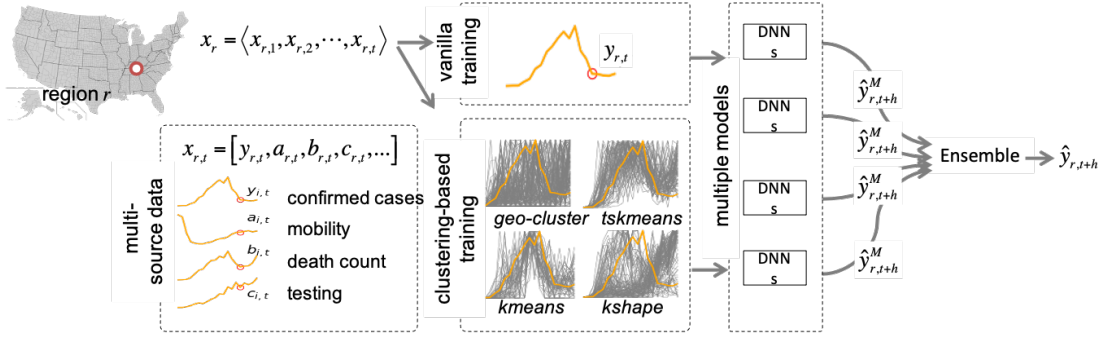
Fig. 1: Framework of deep learning based multi-source ensemble.

algorithm-based clustering (*alg-clustering*). *Geo-clustering*: Clustering is based on their geographical proximity, e.g. partition counties $\mathcal{X}$ based on their state codes for the US. We propose this method due to differences across regions with respect to their size, population density, epidemiological context, and differences in how policies are being implemented. Thus we assume those who belong to the same jurisdictions would have strong relationship in COVID-19 time series. *Alg-clustering*: Clustering using (*i*) k-means [31] which partitions N observations into k clusters in which each observation belongs to the cluster with the nearest mean; (*ii*) time series k-means (tskmeans) [32] that clusters time series data using the smooth subspace information; (*iii*) kshape [33] uses a normalized version of the cross-correlation measure in order to consider the shapes of time series while comparing them. Note that kmeans requires the time series to be clustered must have the same length, while geo-clustering, tskmeans and kshape allow for clustering on different lengths of time series. Alg-clustering discovers implicit correlation of epidemic trends which does not assume any geographical knowledge. We denote the set of above methods as $\mathcal{A} = \{A_{vani}, A_{geo}, A_{km}, A_{ts}, A_{ks}\}$.

E. Ensemble

Ensemble learning is primarily used to improve the model performance. Ren et. al. [34] present a comprehensive review. In this paper, we implement *stacking* ensemble. It is to train a separate dense neural network using the predictions of individual models as the inputs. We use leave-one-out cross validation to train and predict for each region. For each target value $\mathbf{z}_t$, we train the ensemble model using the training samples from the same region but other time points.

F. Probabilistic Forecasting

In the epidemic forecasting domain, probabilistic forecasting is important for capturing the uncertainty of the disease dynamics and to better support public health decision making. We implement MCDropout [35] for each individual predictors to demonstrate estimation of prediction uncertainty. However, the ensemble predictions are point estimation by the definition of stacking.

G. Proposed Framework

Fig. 1 shows the framework of the proposed method. It works as follows: (1) we choose a geographical scale and resolution, e.g. counties in the US; (2) we collect and process multi-source training data; (3) we cluster regions into certain groups based on their similarities between time series of new confirmed cases; (4) we train multiple predictors per cluster and ensemble individual predictors to make final predictions.

1) Multiple data sources: In order to model the co-evolution of multiple factors in COVID-19 spread, we incorporate the following data sources in our models to make future forecasts. *COVID-19 Surveillance Data* [36] and *Case Count Growth Rate (CGR)* quantify case count and case count changes of COVID-19 time series. *COVID-19 Testing Data* [37], *Testing Rate (TR)* and *Testing Positive Rate (TPR)* quantify the COVID-19 testing coverage in each region. We denote the set of multiple sources as $\mathcal{D}$ where $\mathcal{D}$ can be expanded by combining any new data sources. We generate $\mathcal{X}$ by preprocessing $\mathcal{D}$. Details of data description and generation are shown in section IV-A.

2) Multiple RNN-based models: By combining different data sources (single feature, m features, attention m features), RNN modules (RNN, GRU, LSTM), and training methods (vanilla, geo, kmeans, tskmeans, kshape), we implement multiple individual models. For country, US state and US county levels, models include: *RNN*, *GRU*, *LSTM* use vanilla training with single feature; *RNN-m*, *GRU-m*, *LSTM-m* use vanilla training with m features; *RNN-att*, *GRU-att*, *LSTM-att* are attention-based models using vanilla training with m features. For US county level, to investigate the effect of clustering training, we implement additional models using RNN module and single feature: *RNN-geo*, *RNN-kmeans*, *RNN-tskmeans* and *RNN-kshape*. We analyze the effects by varying clustering methods while fixing other factors. Thus other combinations of modules, features and training methods are omitted in this work. We denote the set of individual models as $\mathcal{M}$. Note that $\mathcal{M}$ is not limited to the models we implemented in this paper. It can be expanded by adding or improving upon any of the individual components.

3) Training and forecasting: Algorithm 1 presents how the proposed framework works. We first preprocess the collected data sources $\mathcal{D}$ to generate $\mathcal{X}$ based on the data availability for

different resolutions. Each feature is in the form of time series of weekly data points at a given geographical resolution. We design various models $\mathcal{M}$ for different resolutions based on $\mathcal{D}$. Next, each model in $\mathcal{M}$ is trained using its corresponding cluster of training data. For region r, given an input $\mathbf{X}_{r,t}$, a model M will output $\hat{\mathbf{z}}_{r,t+h}^{M}$. Then the outputs of individual models in $\mathcal{M}$ will be combined using stacking ensemble which will output the final prediction $\hat{\mathbf{z}}_{r,t+h}$ for region r at time $t+h$.

Algorithm 1: Pseudocode of the proposed framework

Input: $\mathbf{X}_{r,t}$: the input time series for region r; $\mathbf{X}_r$: historical time series for region r; $\mathcal{X}$: the set of time series for N regions; $\mathcal{A}$: the set of clustering methods; $\mathcal{M}$: the set of model types;

Output: $\hat{\mathbf{z}}_{r,t+h}$: new confirmed case forecasting at time $t + h$

Data: $\mathcal{D}$: the set of data sources

1 Preprocess $\mathcal{D}$ to generate $\mathcal{X}$ and $\mathbf{X}_{r,t}$

2 $O_r \leftarrow \emptyset$ // The set of individual model predictions

3 **for** A *in* $\mathcal{A}$ **do**

4     $C_r \leftarrow \{\mathbf{X}_r\}$

    /* Start clustering   */

5     $C_A \leftarrow A.fit(\mathcal{X})$ // C_A is the clustering results using method A

6   **for** i *in* $1, \ldots, N$ **do**

7   **if** $\mathbf{X}_r$ *and* $\mathbf{X}_i$ *belong to the same cluster in* C_A **then**

8           $C_r := C_r \cup \mathbf{X}_i$

    /* Start training, forecasting   */

9   **for** M *in* $\mathcal{M}$ **do**

10   **if** M *is* A *related* **then**

11   train M using C_r

12           $\hat{\mathbf{z}}_{r,t+h}^{M} := M(\mathbf{X}_{r,t}, \theta)$

13           $O_r := O_r \cup \hat{\mathbf{z}}_{r,t+h}^{M}$

14 $\hat{\mathbf{z}}_{r,t+h} := \mathcal{F}(O_r, w)$ // $\mathcal{F}$ is ensemble algorithm

---

4) Multi-step forecasting: For single feature, we use a recursive forecasting approach to make multi-step forecasting. That is appending the most recent prediction to the input for the next step forecasting. For multiple features that include exogenous time series as the input, we train a separate model for each step ahead forecasting.

IV. EXPERIMENT SETUP

A. Data

- **COVID-19 surveillance data** is obtained via the UVA COVID-19 surveillance dashboard [36]. It contains daily confirmed cases (**CF**) and death count (**DT**) at the resolution of county/state in the US and national-level data for other countries. Daily case counts and death counts are further aggregated to weekly counts.

TABLE I: Dataset Summary.

Data set	# regions	# weeks	# features
Global	8	25	6
US-State	50	25	8
US-County	2952	25	7

- **Case count growth rate (CGR)**: Denoting the new confirmed/death case count at week t as n_t, the CGR of week $t + 1$ is computed as $log(n_{t+1} + 1) - log(n_t + 1)$, where we add 1 to smooth zero counts. We compute confirmed CGR (**CCGR**) and death CGR (**DCGR**).
- **COVID-19 testing data** via JHU COVID-19 tracking project [37]. It includes multiple data like positive and negative testing count for state and country level of the US. We compute testing per 100K (**TR**) and testing positive rate (**TPR**) i.e. positive/(positive+negative).

All data sources are weekly and ends on Saturday. It starts from Week ending March 7th and ends at Week ending August 22nd (25 weeks) at **Global**, **US-State** and **US-County** resolutions. The global dataset includes Austria, Brazil, India, Italy, Nigeria, Singapore, the United Kingdom, and the United States. The summary of each dataset is shown in Table I. We chose 2020/03/07 as the start week since commercial laboratories began testing for SARS-CoV-2 in the US on March 1st, 2020. Thus the COVID-19 surveillance data before that date is substantially noisy. The forecasting week starts from 2020/05/23 and we make 4 weeks ahead forecasting at each week until 2020/08/22. For example, if we use time series of data from 2020/03/07 to 2020/05/16 to train models, then the forecasting weeks are 2020/05/23, 2020/05/30, 2020/06/06, and 2020/06/13. Then we move one week ahead to repeat the training and forecasting.

B. Metrics

The metrics used to evaluate the forecasting performance are: *root mean squared error (RMSE)*, *mean absolute percentage error (MAPE)*, *Pearson correlation (PCORR)*.

C. Baselines

To serve as baselines for comparing the individual models, we also implemented SEIR compartmental model and several statistical time series models as well as state-of-the-art deep learning models. There are a few deep learning models proposed recently for COVID-19 forecasting which have not been peer reviewed, thus we do not consider any models published within 2 months upon our completion of this paper.

- **Naive** uses the observed value of the most recent week as the future prediction.
- **SEIR** [38] is an SEIR compartmental model for simulating epidemic spread. We calibrate model parameters based on surveillance data for each region. Predictions are made by persisting the current parameter values to the future time points and run simulations.
- **Autoregressive (AR)** uses observations from previous time steps as input to a regression equation to predict

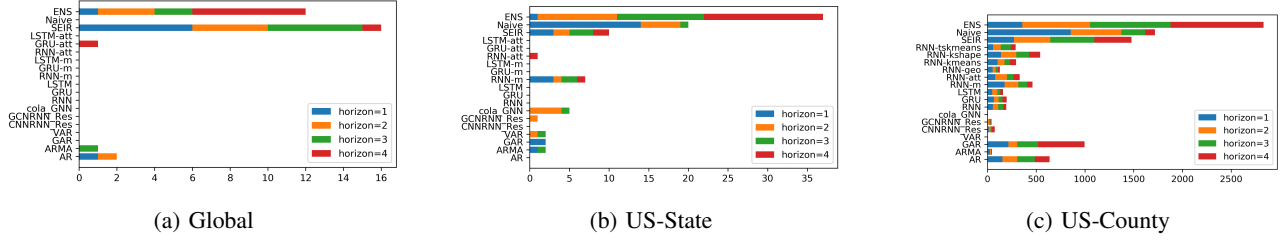|		
(a) Global	(b) US-State	(c) US-County

Fig. 2: The distribution of best RMSE performance among individual methods. x-axis denotes the number of best performance achieved by each method.

the value at the next time step. We train one model per region using AR order 3.

- **Global Autoregression (GAR)** trains one global AR model using the data available from each region. This is similar to the clustering-based methods that we proposed in this paper. We train one model per resolution using AR order 3.
- **Vector Autoregression (VAR)** is a stochastic process model used to capture the linear interdependencies among multiple time series. We train one model per resolution using AR order 3.
- **Autoregressive Moving Average (ARMA)** [39] is used to describe weakly stationary stochastic time series in terms of two polynomials for the autoregression (AR) and the moving average (MA). We set AR order to 3 and MA order to 2.
- **CNNRNN-Res** [20] uses RNNs, CNNs, and residual links to capture spatio-temporal correlation within and between regions. We train one model per region. We set the residual window size as 3 and all the other parameters are set as the same as the original paper.
- **Cola-GNN** [23] uses attention-based graph neural networks to combine graph structures and time series features in a dynamic propagation process. We train one model per resolution. We set RNN window size as 3 and all the other parameters are set as the same as the original paper.

D. Settings and Implementation Details

We set training window size $T = 3$ for all RNN-based models due to the short length of available CF and DT. We examine weekly CF forecasting at county and state level for US and country level for 8 countries of which at least one country is from each continent. The forecasting is made to 1, 2, 3, 4 weeks ahead at each time point i.e. $h = \{1, 2, 3, 4\}$. All RNN-based models consist of 2 recurrent neural network layers with 32 hidden units, 1 dense layer with 16 hidden units, 1 dropout layer with 0.2 drop probability. We set batch size as 32, epoch number as 500. Stacking ensemble model consists of 1 dense layer with 32 hidden units and RELU activation function. We train ensemble with batch size 8 and epoch number as 200. Adam optimizer with default settings and early stopping with patience of 50 epochs are used for all model training. Geo-clustering and alg-clustering methods are applied when training county level models. We set the
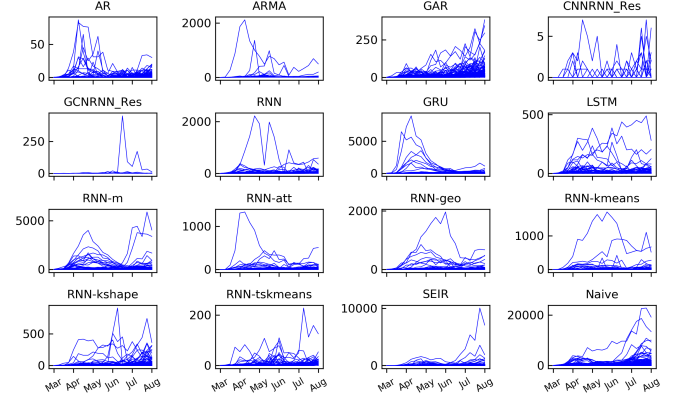

Fig. 3: (US-county) The curves of weekly new confirmed cases grouped by individual models where the best RMSE performance is achieved. y-axis denotes new confirmed case count and x-axis denotes weeks (25 weeks).

number of clusters for alg-clustering method as $k = 50$. The clustering is conducted on the normalized training curves using MinMaxScaler. Single feature means time series of CF. For country level forecasting, m features include CF, DT, CCGR, DCGR. For US state level forecasting, m features include CF, DT, CCGR, DCGR, TR and TPR. And CF, DT, CCGR, and DCGR are included for US county level forecasting. AR-based models and CNNRNN-based models are trained with single feature time series. For all models, we run 50 Monte Carlo predictions. For SEIR method, we calibrate a weekly effective reproductive number (R_{eff}) using simulation optimization to match the new confirmed cases per 100k. We set the disease parameters as follows: mean incubation period 5.5 days, mean infectious period 5 days, delay from onset to confirmation 7 days and case ascertainment rate of 15% [40].

V. RESULTS

A. Forecasting Performance

We evaluate the model performance of horizon 1, 2, 3, and 4 at county-, state- and national-level using RMSE, MAPE and PCORR. To mitigate the performance bias caused by our settings, we divide the individual models into several categories based on different modules, training methods, features. Then we calculate the average performance per category. Note that an individual model may belong to multiple categories. **RNNs** includes models mainly consist of RNN module. **GRUs**

includes models mainly consist of GRU module. **LSTMs** includes models mainly consist of LSTM module. **GNNRNNs** includes models mix CNN, RNN, GNN modules. **ARs** includes autoregression based models. **Vanillas** includes models in RNNs that use single feature and vanilla training. **Clusters** includes models in RNNs that use single feature and geo, kmeans, tskmeans, kshape clustering training. **SglFtrs** includes *RNN, GRU, LSTM*. **MulFtrs** includes *RNN-m, GRU-m, LSTM-m, RNN-att, GRU-att, LSTM-att*. **SEIRs** includes *SEIR*. **Naive** includes *Naive*. **ENS** is stacking ensemble of RNNs, GRUs and LSTMs. GNNRNNs excludes *cola-GNN* and ARs excludes *VAR* for US-county forecasting due to their failures to make reasonable forecasting. For more details please refer to Table II note.

Table II presents the numerical results. In general, we observe that (*i*) at US state and county level ENS performs the best on 2, 3 and 4 weeks ahead forecasting while Naive performs the best on 1 week ahead. (*ii*) SEIR outperforms others at global level forecasting on horizon 1, 2 and 3. (*iii*) Models with a single type of DNN modules outperform those with mixed types of modules. (*iv*) Models trained with vanilla methods outperform models trained with clustering-based methods. We will investigate and explain this observation in the next two paragraphs. (*v*) Models trained with multiple features outperform models trained with a single feature at US state and county level.

To better understand the model performance distribution over all regions, we select one individual method from each category without overlapping and count frequency of the best performance (**FRQBP**) per method. Fig. 2 presents the aggregate counts of 1, 2, 3, 4 horizons. Note that methods with larger counts do not necessarily have better MAPE, RMSE and PCORR performance. The observations are in general consistent with those from Table II but with more specific observations regarding FRQBP: (*vi*) the best 1 week ahead predictions are mostly achieved by Naive methods. (*vii*) For US state and county level, the best 2, 3, 4 weeks ahead predictions are achieved by ENS and the value increases as horizon increases. (*viii*) Alg-clustering-based models and models with multiple features achieve more best performance than vanilla models. (*ix*) GAR and AR have larger FRQBP than DNN models at US county level.

Furthermore, in Fig. 3 we show the US county level curves of weekly new confirmed cases grouped by individual methods where the best RMSE performance is achieved. It is interesting to observe that different methods achieve best performance over regions with different patterns, such as when the curves of weekly new confirmed cases have large fluctuation between subsequent weeks, the deep learning-based methods are able to capture the dynamics well as opposed to SEIR and Naive methods. The naive and SEIR models assume certain level of regularity in the time series, which tends to be violated in the curves pertaining to deep learning methods. LSTM, RNN-kmeans, RNN-kshape, and RNN-tskmeans are outstanding in capturing dynamics with various patterns which show their generalization capability for time series forecasting. However,

as we mentioned above the good performance in FRQBP does not indicate a better average performance on RMSE, MAPE, and PCORR since the latter also depends on the scales of ground truths. AR and GAR perform well on capturing dynamics of small number of cases. The CNNRNN-based methods does not perform well on county level forecasting. The likely reason is that the complexity of these models is much higher than simple RNN-based models and the complexity increases as the number of regions increases. Thus overfitting happens with such a small training data size at county level.

We want to highlight that in order to investigate deep learning models for COVID-19 forecasting, the ensemble framework in this paper only combine DNN models. However it can but not necessarily include baselines like SEIR and Naive who perform very well in this task. We encourage researchers to ensemble models of various types to average the forecasting errors made by a particular poor model.

B. Sensitivity Analysis and Discussion

In this section, we show sensitivity analysis on model types, feature number, and clustering method for individual models.

1) RNN modules: We compare RMSE performance of models with pure RNN, GRU, LSTM modules. Fig. 4 shows the comparison between RNN, GRU, LSTM methods for three resolution datasets. We observe that RNN performs the best on 1 week ahead forecasting while GRU and LSTM outperform RNN on 3 and 4 weeks ahead forecasting at state and county level. The results indicate that RNN tends to perform better than GRU and LSTM for short-term forecasting while it loses advantage for long-term forecasting.

2) Number of features: In our framework, we involve multiple data sources to model the co-evolution of multiple factors in epidemic spreading. We implement individual models either with single feature or with m features. In addition, we use an attention layer to model the effect of other features on the target feature. Fig. 5 presents the model performance of GRU, GRU-m, and GRU-att at three datasets. In general, GRU-m and GRU-att using m features outperform GRU using single feature in most cases except for 1 and 2 week ahead forecasting at global level. Note that for global forecasting, there is no testing information which is a critical factor for revealing COVID-19 dynamics.

3) Clustering method: Clustering-based training is applied in our framework to mitigate the likely overfitting due to small training data size. We compare US county level model performance of RNN, RNN-geo, RNN-kmeans, RNN-tskmeans, RNN-kshape. The comparison is shown in 6. In general, we observe RNN, RNN-geo and RNN-kshape outperform RNN-kmeans and RNN-tskmeans. RNN-geo performs the best for 1 and 2 week ahead forecasting while RNN-kshape performs the best for 3 and 4 week ahead forecasting. This indicates that geo-clustering can capture near future co-evolution dynamics within a state informed by similar local epidemiological environments. Kshape clustering can further capture far future dynamics informed by other counties with similar trends.

TABLE II: RMSE, MAPE and PCORR performance of different methods on the three datasets with horizon = 1, 2, 3, 4. Bold face indicates the best results of each column.

	Global				US-State				US-County			
RMSE(↓)	1	2	3	4	1	2	3	4	1	2	3	4
ARs	38067	46065	53942	57905	3255	3546	3822	4933	77	92	101	120
CNNRNNs	36895	49589	62499	69172	3511	4253	4615	5546	114	138	147	149
RNNs	31232	34877	44838	55403	2200	2940	3593	4605	60	80	96	110
GRUs	31172	36503	41513	55325	1936	2666	3520	4507	58	78	96	111
LSTMs	28023	35252	43130	53907	2031	2682	3576	4483	60	79	97	111
Vanillas	26323	33337	44273	54620	2135	2611	3415	4162	65	79	95	109
Clusters	-	-	-	-	-	-	-	-	72	91	103	117
SglFtrs	26878	33513	44838	54909	1824	2614	3533	4610	56	77	97	112
MulFtrs	32052	36648	42604	55008	1559	2154	3091	4114	47	66	84	97
SEIRs	**8761**	**9393**	**13879**	22805	2310	3362	4558	4635	65	75	82	96
Naive	15427	24899	27415	29318	**1095**	1936	1969	2466	**37**	**48**	60	71
ENS	18166	23204	28150	**19558**	1261	**1547**	**1599**	**2109**	45	49	**59**	**61**
MAPE(↓)	1	2	3	4	1	2	3	4	1	2	3	4
ARs	173	167	187	195	2301	2571	1549	1821	129	119	121	127
CNNRNNs	95	123	173	197	1833	2656	1370	1777	148	187	202	191
RNNs	82	95	105	133	1265	1662	772	1084	116	142	153	162
GRUs	61	68	86	94	1335	1870	604	834	93	118	131	143
LSTMs	43	64	71	89	1453	1848	650	947	94	119	129	143
Vanillas	35	52	75	91	1092	1733	335	533	84	95	100	115
Clusters	-	-	-	-	-	-	-	-	140	167	171	179
SglFtrs	37	57	86	105	891	1260	509	719	94	122	139	152
MulFtrs	75	87	94	112	1448	1839	732	1101	101	127	139	142
SEIRs	**12**	**12**	**18**	28	996	**1067**	555	585	344	331	308	292
Naive	20	29	38	29	**796**	1198	565	590	**75**	98	95	83
ENS	26	30	31	**22**	1048	1177	**524**	**509**	90	**95**	**91**	**80**
PCORR(↑)	1	2	3	4	1	2	3	4	1	2	3	4
ARs	0.8787	0.8335	0.8040	0.7995	0.8713	0.8257	0.7161	0.5214	0.7712	0.6070	0.5586	0.3062
CNNRNNs	0.9016	0.8479	0.8015	0.8217	0.7654	0.6441	0.5195	0.3119	0.1828	-0.0232	0.0246	0.0636
RNNs	0.9477	0.9167	0.8690	0.7950	0.9094	0.8403	0.7974	0.6129	0.8321	0.7103	0.6086	0.5161
GRUs	0.9295	0.8968	0.8719	0.7966	0.9426	0.9152	0.8349	0.6791	0.8520	0.7377	0.5819	0.4776
LSTMs	0.9312	0.8829	0.8329	0.8030	0.9218	0.8776	0.7844	0.6782	0.8513	0.7226	0.5655	0.4779
Vanillas	0.9453	0.9106	0.8447	0.7703	0.9301	0.9094	0.8497	0.7521	0.8307	0.7528	0.6350	0.5297
Clusters	-	-	-	-	-	-	-	-	0.8167	0.6544	0.5242	0.4146
SglFtrs	0.9388	0.8989	0.8306	0.7560	0.9392	0.9035	0.8175	0.6635	0.8607	0.7347	0.5752	0.4744
MulFtrs	0.9348	0.8988	0.8716	0.8193	**0.9662**	**0.9522**	0.8978	0.7882	0.9292	0.8656	0.7679	0.7247
SEIRs	**0.9957**	**0.9954**	**0.9851**	0.9576	0.5806	0.5138	0.5379	0.3622	0.8632	0.7997	0.7809	0.7000
Naive	0.9888	0.9715	0.9498	0.9300	0.9764	0.9563	0.9208	0.8110	**0.9546**	0.9071	0.8485	0.7748
ENS	0.9660	0.9397	0.9163	**0.9725**	0.9601	0.9488	**0.9476**	**0.9072**	0.9162	**0.9166**	**0.8622**	**0.8789**

RNNs: *RNN, RNN-geo, RNN-m, RNN-att, RNN-kmeans, RNN-tskmeans, RNN-kshape*. **GRUs**: *GRU, GRU-m , GRU-att*. **LSTMs**: *LSTM, LSTM-m, LSTM-att*. **GNNRNNs**: *cola-GNN, GCNRNN-Res, CNNRNN-Res*. **ARs**: *AR, ARMA, VAR, GAR*. **Vanillas**: *RNN*. **Clusters**: *RNN-geo, RNN-kmeans, RNN-tskmeans, RNN-kshape*. **SglFtrs**: *RNN, GRU, LSTM*. **MulFtrs**: *RNN-m, GRU-m, LSTM-m, RNN-att, GRU-att, LSTM-att*. **Naive**: *naive*. **SEIRs**: *SEIR*. **ENS** is stacking ensemble of the union of RNNs, GRUs, and LSTMs. CNNRNNs excludes *cola-GNN* and ARs excludes *VAR* for US-county forecasting due to their failures to make reasonable forecasting.

VI. CONCLUSION

In this work, we developed an ensemble framework that combines multiple RNN-based deep learning models using multiple data sources for COVID-19 forecasting. The multiple data sources enable better forecasting performance. To mitigate the likely overfitting to noisy and small size of training datasets, we proposed clustering-based training method to further improve DNN model performance. We trained stacking ensembles to combine individual deep learning models of simple architectures. We show that the ensemble in general performs the best among baseline individual models for high resolution and long term forecasting like US state and county level. Ensembles play a very important role for improving model performance for COVID-19 forecasting. A comprehensive comparison between SEIR methods, DNN-based methods and AR-based methods are conducted. In the context of COVID-19, our experimental results show that different models are likely to perform best on different patterns of time series. Despite the lack of sufficient training data, DNN-based methods can capture the dynamics well and show strong generalization ability for high resolution forecasting as opposed to SEIR and Naive methods. Among multiple DNN-based models, spatio-temporal models are more likely to overfitting due to the high model complexity for high resolution forecasting.

(a) Global

(b) US-State

(c) US-County

Fig. 4: Sensitivity analysis on RNN modules.

(a) Global

(b) US-State

(c) US-County

Fig. 5: Sensitivity analysis on feature number.

(a) US-County MAPE

(b) US-County RMSE

(c) US-County PCORR

Fig. 6: Sensitivity analysis on clustering training.

REFERENCES

[1] WHO, *WHO Coronavirus Disease (COVID-19) Dashboard*, 2020 (accessed August 26, 2020). [Online]. Available: https://covid19.who.int/

[2] E. Duffin, "Impact of the coronavirus pandemic on the global economy-statistics & facts," *Statistica Report, April*, vol. 3, p. 2020, 2020.

[3] C. Anastassopoulou, L. Russo, A. Tsakris, and C. Siettos, "Data-based analysis, modelling and forecasting of the covid-19 outbreak," *PloS one*, vol. 15, no. 3, p. e0230405, 2020.

[4] G. Giordano, F. Blanchini, R. Bruno, P. Colaneri, A. Di Filippo, A. Di Matteo, and M. Colaneri, "Modelling the covid-19 epidemic and implementation of population-wide interventions in italy," *Nature Medicine*, pp. 1–6, 2020.

[5] T. Yamana, S. Pei, and J. Shaman, "Projection of covid-19 cases and deaths in the us as individual states re-open may 4, 2020," *medRxiv*, 2020.

[6] Z. Yang, Z. Zeng, K. Wang, S.-S. Wong, W. Liang, M. Zanin, P. Liu, X. Cao, Z. Gao, Z. Mai *et al.*, "Modified seir and ai prediction of the epidemics trend of covid-19 in china under public health interventions," *Journal of Thoracic Disease*, vol. 12, no. 3, p. 165, 2020.

[7] D. Kai, G.-P. Goldstein, A. Morgunov, V. Nangalia, and A. Rotkirch, "Universal masking is urgent in the covid-19 pandemic: Seir and agent based models, empirical validation, policy recommendations," *arXiv preprint arXiv:2004.13553*, 2020.

[8] S. L. Chang, N. Harding, C. Zachreson, O. M. Cliff, and M. Prokopenko, "Modelling transmission and control of the covid-19 pandemic in australia," *arXiv preprint arXiv:2003.10218*, 2020.

[9] C. C. Kerr, R. M. Stuart, D. Mistry, R. G. Abeysuriya, G. Hart, K. Rosenfeld, P. Selvaraj, R. C. Nunez, B. Hagedorn, L. George *et al.*, "Covasim: an agent-based model of covid-19 dynamics and interventions," *medRxiv*, 2020.

[10] L. Magri and N. A. K. Doan, "First-principles machine learning modelling of covid-19," *arXiv preprint arXiv:2004.09478*, 2020.

[11] R. Dandekar and G. Barbastathis, "Neural network aided quarantine control model estimation of global covid-19 spread," *arXiv preprint arXiv:2004.02752*, 2020.

[12] CDC, *COVID-19 Forecasts: Deaths*, 2020 (accessed August 10, 2020). [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/covid-data/forecasting-us.html

[13] A. Harvey and P. Kattuman, "Time series models based on growth curves with applications to forecasting coronavirus," *Covid Economics, Vetted*

and Real-Time Papers, no. 24, 2020.

[14] F. Petropoulos and S. Makridakis, "Forecasting the novel coronavirus covid-19," *PloS one*, vol. 15, no. 3, p. e0231236, 2020.

[15] M. H. D. M. Ribeiro, R. G. da Silva, V. C. Mariani, and L. dos Santos Coelho, "Short-term forecasting covid-19 cumulative confirmed cases: Perspectives for brazil," *Chaos, Solitons & Fractals*, p. 109853, 2020.

[16] Z. Hu, Q. Ge, L. Jin, and M. Xiong, "Artificial intelligence forecasting of covid-19 in china," *arXiv preprint arXiv:2002.07112*, 2020.

[17] V. K. R. Chimmula and L. Zhang, "Time series forecasting of covid-19 transmission in canada using lstm networks," *Chaos, Solitons & Fractals*, p. 109864, 2020.

[18] P. Arora, H. Kumar, and B. K. Panigrahi, "Prediction and analysis of covid-19 positive cases using deep learning models: A descriptive case study of india," *Chaos, Solitons & Fractals*, p. 110017, 2020.

[19] 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, p. e0188941, 2017.

[20] Y. Wu, Y. Yang, H. Nishiura, and M. Saitoh, "Deep learning for epidemiological predictions," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 2018, pp. 1085–1088.

[21] L. Wang, J. Chen, and M. Marathe, "Defsi: Deep learning based epidemic forecasting with synthetic information," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 9607–9612.

[22] B. Adhikari, X. Xu, N. Ramakrishnan, and B. A. Prakash, "Epideep: Exploiting embeddings for epidemic forecasting," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 577–586.

[23] S. Deng, S. Wang, H. Rangwala, L. Wang, and Y. Ning, "Graph message passing with cross-location attentions for long-term ili prediction," *arXiv preprint arXiv:1912.10202*, 2019.

[24] L. Wang, J. Chen, and M. Marathe, "Tdefsi: Theory-guided deep learning-based epidemic forecasting with synthetic information," *ACM Transactions on Spatial Algorithms and Systems (TSAS)*, vol. 6, no. 3, pp. 1–39, 2020.

[25] S. R. Venna, A. Tavanaei, R. N. Gottumukkala, V. V. Raghavan, A. S. Maida, and S. Nichols, "A novel data-driven model for real-time influenza forecasting," *IEEE Access*, vol. 7, pp. 7691–7701, 2019.

[26] A. Kapoor, X. Ben, L. Liu, B. Perozzi, M. Barnes, M. Blais, and S. O'Banion, "Examining covid-19 forecasting using spatio-temporal graph neural networks," *arXiv preprint arXiv:2007.03113*, 2020.

[27] J. Gao, R. Sharma, C. Qian, L. M. Glass, J. Spaeder, J. Romberg, J. Sun, and C. Xiao, "Stan: Spatio-temporal attention network for pandemic prediction using real world evidence," *arXiv preprint arXiv:2008.04215*, 2020.

[28] A. Ramchandani, C. Fan, and A. Mostafavi, "Deepcovidnet: An interpretable deep learning model for predictive surveillance of covid-19 using heterogeneous features and their interactions," *arXiv preprint arXiv:2008.00115*, 2020.

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

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

[31] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the royal statistical society. series c (applied statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[32] X. Huang, Y. Ye, L. Xiong, R. Y. Lau, N. Jiang, and S. Wang, "Time series k-means: A new k-means type smooth subspace clustering for time series data," *Information Sciences*, vol. 367, pp. 1–13, 2016.

[33] J. Paparrizos and L. Gravano, "k-shape: Efficient and accurate clustering of time series," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, 2015, pp. 1855–1870.

[34] Y. Ren, L. Zhang, and P. N. Suganthan, "Ensemble classification and regression-recent developments, applications and future directions," *IEEE Computational Intelligence Magazine*, vol. 11, no. 1, pp. 41–53, 2016.

[35] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*, 2016, pp. 1050–1059.

[36] UVA, *UVA COVID-19 Surveillance Dashboard*, 2020 (accessed May 14, 2020). [Online]. Available: https://nssac.bii.virginia.edu/covid-19/dashboard/

[37] JHU, *The COVID-19 Tracking Project*, 2020 (accessed July 23, 2020). [Online]. Available: https://covidtracking.com/data/download

[38] S. Venkatramanan, J. Chen, S. Gupta, B. Lewis, M. Marathe, H. Mortveit, and A. Vullikanti, "Spatio-temporal optimization of seasonal vaccination using a metapopulation model of influenza," in *2017 IEEE International Conference on Healthcare Informatics (ICHI)*. IEEE, 2017, pp. 134–143.

[39] J. Contreras, R. Espinola, F. J. Nogales, and A. J. Conejo, "Arima models to predict next-day electricity prices," *IEEE transactions on power systems*, vol. 18, no. 3, pp. 1014–1020, 2003.

[40] S. A. Lauer, K. H. Grantz, Q. Bi, F. K. Jones, Q. Zheng, H. R. Meredith, A. S. Azman, N. G. Reich, and J. Lessler, "The incubation period of coronavirus disease 2019 (COVID-19) from publicly reported confirmed cases: Estimation and application," *Annals of internal medicine*, vol. 172, no. 9, pp. 577–582, 2020.