UNIVERSITÄT HEIDELBERG

Advanced Machine Learning

Final Project

Xianghe Ma, Matrikel Nr. 3690701
Hanmeng Jiang, Matrikel Nr. 3691875
Tong YU, Matrikel Nr. 3682111

September 27, 2021

Preliminaries

Our code can be found in a public Github repository:
https://gitlab.com/xiaohemaikoo/tlgnn-covid19

Abstract

Corona virus pandemic, which is first outbreak in late 2019, has caused huge economic losses and cost many lives all over the world. It is clear that human mobility is a main reason to its spread and asynchronous outbreaks across countries. In this paper our main work is to explore the impact of population movement on the spread of COVID-19. We mainly utilize confirmed cases reports and population mobility data from 4 European countries to train and test our proposed methods. In order to learn potential connection between population movement and the growth of infection rate in different regions, we build a sub-GNN model which is MPNN and combine it with LSTM model to predict new cases in the future. Then we use transfer learning to generalize the model in the data of different countries and regions, and get more accurate prediction results. In the experiment, we also applied several baseline methods for comparative analysis, and found that the transfer learning sub-GNN can still maintain high accuracy in medium and long-term prediction. Through our results we will be able to forecast the trend of the spread of COVID-19 in different countries. Therefore we provide useful data for decision maker to decide whether to open or close border travel restrictions and other anti-epidemic policies to prevent a real outbreak of epidemics among people.

Table of Contents

Chapter 1

Introduction[Hanmeng JIANG]

Corona virus pandemic, also, Covid-19, first outbreak in late 2019. Almost no one could imagine the Covid-19 pandemic would continue to strike all over the world till today for almost 2 years. Due to it is highly infectious, it now has spread to all over the world, and has infected at least 226 millions people and caused over 4.6 millions deaths[1]. Many governments of different countries underestimated how highly infectious it is and its severity despite some governments took strict lockdown policies at early stage. Till now although the vaccination rate grows fast, it still has caused a severe impact on global economy and taken many lives, therefore accurate predictions of new cases becomes more and more important in the next days to help governments take countermeasures towards Covid-19 pandemic on balancing the medical resources and the economic progress.
The main ways of virus spread is by human direct or indirect contacts, owing to one fact that people's mobility, all of us are moving from a location to another in most of time. Therefore, doing the research on tracing people's mobility might make sense. Facebook have the records of users' location at different times and we assume that these data could be meaningful and representative. This project is based on this hypothesis, to study and predict the new cases of COVID-19. Our main works are summarized as follows:

- We study researches on epidemic, especially on COVID-19.

[1]WHO. WHO Coronavirus (COVID-19) Dashboard. https://covid19.who.int/, Sep 2021.

- We take *Transfer Graph Neural Networks for Pandemic Forecasting* [1] as reference, build a model for learning the spreading of COVID-19 in a country via the country's graph based on MPNN.
- We add LSTM model to MPNN, to solve the dependency problem of long time-series data.
- We transfer a disease spreading model based on MAML from countries where the outbreak has been stabilized, to another country where the COVID spreading is still at its early stage.
- We evaluate baseline methods and the transfer learning MPNN method by comparing predictions with ground truth. Combined with the complex situation of COVID-19 epidemic spreading in the real world, we analyze the performance, pros and cons of each method.

Chapter 2

Related Work[Tong YU, Xianghe MA]

A lot of researches show that many reasons can accelerate the spread of Covid-19 virus. Sometimes weather and season can fluctuate the spreading rate of Covid-19 and the peak of infections usually emerges during winter times [2]. Fresh and frozen food can also contain Corona virus and then become the source to infect people [3]. In addition, human mobility is the highly agreed reason that make the Corona virus spread all over the world [4]. The more individuals move out from one area to others or move inside an area, the higher the likelihood that individuals in the destination district are threatened by the infection. This is a notable perception [5].

With the booming of machine learning and AI in data science in recent years, more and more researchers focus on studying the spread of epidemics through artificial intelligence techniques. A new survey shows that artificial intelligence and big data can help people to control the Covid-19 spread situation and inspires further studies to prevent scattered outbreaks of Covid-19 [6]. Lorch and Trouleau [7] build a SEIR model derived from the principles of infectious epidemics and data science. They use Bayesian optimization method to train the parameters. By evaluating the data form Bern, Switzerland under their SEIR model, they get results of the infectious rate related social activities. Flaxman and Seth [8] also proposed a Bayesian method but use fewer data. They collect the deaths data from 11 European coun-

tries to evaluate the time-varying reproduction number. Their results illustrate that human interventions have significant influence on the spread of disease, policies like lockdowns can reduce the infectious rate.

Most studies tend to utilize the time-series based models to make predictions about Covid-19 pandemic situations. Such studies like LSTM applied to predict new cases of pandemic are now widely used in the research. Long Short Term Memory networks – usually just called "LSTMs" – are a special kind of RNN(Recurrent Neural Network) model, capable of learning long-term dependencies. They were introduced by Hochreiter and Schmidhuber in 1997 [9], and were refined and popularized by many people in following work. They work tremendously well on a large variety of problems, and are now widely used in pandemic analysis. Chimmula and Zhang's work [10] give an example of using LSTM to analyze and predict new cases of COVID pandemic in Canada. Ma and Zheng et al. [11] used LSTM combined with Markov method to analyze the COVID pandemic all around the world.

Besides, researchers also make use of ARIMA(Auto Regressive Integrated Moving Average) model to analyze and predict pandemic. The ARIMA model was developed in the 1970s by George Box and Gwilym Jenkins as an attempt to describe changes on the time series using a mathematical approach. Kufel [12] utilize the ARIMA model for estimating the dynamics of Corona virus cases to predict future new cases in some European countries. It is a method of assessing the carried out non-drug countermeasures on the elements of the pandemic. Vikas Chaurasia and Saurabh Pal [13] use ARIMA and Regression Model to make predictions of worldwide COVID death cases.

There is also a widely used time series data forecasting third-party library Prophet. Prophet is open source software contributed by Facebook Data Science team. Some works of Covid-19 predictions take the advantages of Prophet to estimate new cases. There is a research in Bangladesh by Mahmud [14] which builds a time series model via Prophet to plot the daily cases and make predictions.

The time-series GNN are now recognized as an effective method on analyzing the epidemics trends. Scarselli, Gori and Tsoi [15] first creatively apply neural networks on graph structure data and form an abstract GNN model. In recent studies GNN models demonstrates superior performance in many classical graph based tasks. Graphs are the most naturally mathematical representations for a wide range of real-world data in many aspects such as biology, finance, traffic, internet, chemistry, computer science. Many classical NP-hard problems are represented by graphs like shortest path problem, travel salesperson problem, min-cost problem, graph matching problem. Utilizing the GNN model can make many complex problems trained by classical neural network layers. The GNN model is based on the information propagation mechanism. Each node updates its own node state by exchanging information with each other until it reaches a stable value. The output of GNN is to calculate the output at each node according to the current node state. Studies on time-series GNN models show a superiority to spatio-temporal graph tasks. Gao and Qian [16] come up with an spatio-temporal GNN model called STAN network to analyze and predict Covid-19 pandemic data. And Kapoor, Ben and Liu [17] also proposed a statistic GNN Networks to analyze mobility data for predicting new cases. The STAN consider a demographic information. They develop a hybrid model by using a graph attention network in order to capture spatio-temporal trends of Covid-19 cases and then make predictions on the number of new infections in the future. They collect infections data and utilize demographic and geographical factors to convert a Covid-19 pandemic problem into a graph problem which can be solved by deep learning method. On the other hand, in Kapoor's [17] model different nodes indicates the human mobility in district level, the spatial edges capture county-to-county movement of a time, and the temporal edges show node level dynamics over time. And They successfully applied their model to analyze human mobility data collected from the United States counties.

Graph Convolutional Networks (GCNs) (Kipf Welling [18], 2017) are an efficient variant of Convolutional Neural Networks (CNNs) and

the most important networks of GNN. Recently, GCNs have achieved significant results in various application areas, including social networks, applied chemistry, natural language processing and computer vision. Recently, since GNNs are widely used in analyzing and predicting COVID, GCN become a relatively new technique in detection of COVID. Yu et al. [19] make use of GCN for detection of COVID-19. Thosini et al. research on COVID detection from chest X-ray and patient metadata by using GCN [20].

Transfer learning aims at transferring the shared knowledge from one task to other related tasks. Usually, the accomplishment of transfer learning is based on certain assumptions. For example, Lawrence and Platt [21], Schwaighofer et al. [22] assume that related tasks should share some (hyper-)parameters. By discovering the shared (hyper-) parameters, the knowledge can be transferred across tasks. If these assumptions fail to be satisfied, however, the transfer may be unsuccessful.

Model-Agnostic Meta-Learning(MAML) is a class of meta-learning algorithms created by Stanford Research and UC Berkeley Alum Dr. Chelsea Finn [23]. They propose a Model-Agnostic (model-independent) method, which now is already a very important model in academia, and can be compatible with any model that uses gradient descent algorithms and can be applied to a variety of tasks, including classification, regression, and even reinforcement learning. Since COVID-19, because of lack of data, MAML appears to be a good method to analyze new areas' COVID situation based on transferring knowledge from the areas that have already been analyzed. Tarun et al. [24] make COVID-19 diagnosis using MAML on limited chest X-ray images. Qian [25] make use of MAML for COVID-19 themed malicious repository detection.

Chapter 3

Methodology

3.1 Graph Neural Network[Tong YU]

A graph neural network (GNN) is a class of neural networks for processing data represented by graph data structures.
The graph refer to graphs in graph theory. It is a graph consisting of two component, vertices(or nodes) and edges. A graph G can be well described by the set of vertices V and edged E it contains. The edges of a graph can be either undirected or directed(Fig 3.1), depending on whether there exist directional dependencies between vertices.

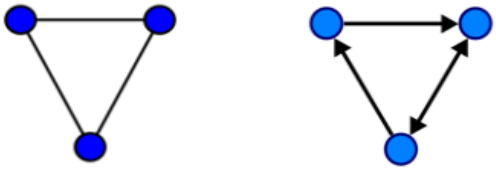

Figure 3.1: Undirected and directed graph(wiki)

The earliest graph neural network originated from Franco's paper [14], and its theoretical basis is Banach's Fixed Point Theorem. A typical application of GNN is node classification. In the node classification problem setup, given a graph G, each node v has its own feature x_v and a true label t_v. The main learning goal of GNN is to represent each node with a vector h_v, which is called embedding hidden state and contains the information of its neighborhood. In specific, GNN is realized by iteratively updating the hidden state of all nodes. At

time $t+1$, the hidden state of node v is updated as follows:

$$h_v^{t+1} = f(x_v, x_{co[v]}, h_{ne[v]}^t, x_{ne[v]}) \tag{3.1.1}$$

where $x_{co[v]}$ represents the features of the edges connecting with node v, $h_{ne[v]}^t$ denotes the embedding hidden state of the neighboring nodes of node v at time t, and $x_{ne[v]}$ denotes the features of the neighboring nodes of node v. The function f is the transition function that projects these inputs onto a vector space and function f is valid for all nodes. Then, applying Banach Fixed Point Theorem, we could get a unique solution for h_v and rewrite the above equation as a global update function which updates state of all nodes:

$$H^{t+1} = F(H^t, X) \tag{3.1.2}$$

where H and X denote the concatenation of all h and x, and such update can be referred to as *message passing* or *neighborhood aggregation.*
Then the output of the GNN is computed by passing the state h_v as well as the node feature x_v to an output function g.

$$o_v = g(h_v, x_v) \tag{3.1.3}$$

Both functions f and g can be represented by feed-forward fully-connected Neural Networks. Finally, assuming that there are a total of p supervision nodes, the model loss can be formalized as

$$loss = \sum_{i=1}^{p} (t_i - o_i) \tag{3.1.4}$$

which can be optimized via gradient descent.

3.2 Graph Convolutional Network[Tong YU]

Graph Convolutional Network (GCN) is the most important GNN and the most commonly used architecture in real-life applications. GCN generalize the operation of convolution from grid data to graph data(Fig 3.2). Instead of having a 2-D array as input, GCN takes a graph as an input.

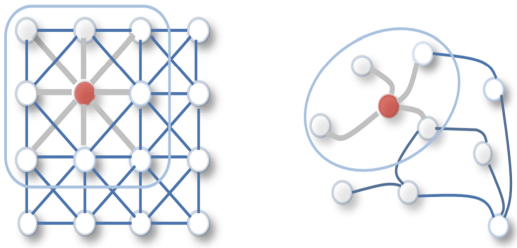

Figure 3.2: 2D Convolution vs. Graph convolution [26]

The main idea of GCN is to generate a node v's representation by aggregating its own features x_v and neighbors' features x_u. Any graph convolutional layer can be written as a nonlinear function, general formula of graph convolution is

$$H^{l+1} = f(H^l, A) \tag{3.2.1}$$

where $H^0 = X$ is the input of the first layer, $X \in R^{N*D}$, N is the number of nodes of a graph, D is the dimension of the feature vector of each node, A is the adjacency matrix, and different choices of f result in different variants of models. But there are two problems with this method, one is not considering the influence of the node itself on itself; the other is the adjacency matrix A is not normalized, which may cause problems when extracting graph features. For example, nodes with many neighbor nodes tend to have greater influence. Therefore we often have the following formula of propagation rule in graph convolutional layer:

$$H^{l+1} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^l W^l) \tag{3.2.2}$$

where $\tilde{A} = A + I_N$, $\tilde{D}$ is degree matrix of $\tilde{A}$, $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$, and $\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$ is actually adjacency matrix A after normalization. W^l is the matrix of trainable parameters of layer l, anf σ is non-linear activation function.

3.3 Message Passing Neural Network[Hanmeng JIANG]

Message Passing Neural Network(MPNN) is a powerful framework for GNN and is considered one of the most generic GNN architectures. Before the model was standardized into a single MPNN framework, different variants had been published by several independent researchers. Gilmer, J et al. [27] abstracts the commonalities of existing models and proposes them into the MPNN framework. This type of architecture was especially popular in chemistry to help predict the properties of molecules.

The input to a MPNN is an undirected graph G with node features x_v and edge features e_{vw}. MPNN operates in two phases. First is a message passing phase, which propagates information across the graph in order to build a neural representation of the whole graph.

$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw}) \tag{3.3.1}$$

which is a sum of all messages M_t obtained from the neighbours. M_t is an arbitrary function that depends on hidden states h_v^t, h_w^t and edges e_{vw} of the neighbouring nodes v, w, and function M_t is actually the transition function f in GNN.

Then we update the hidden state of the node v_t:

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1}) \tag{3.3.2}$$

where function U_t is actually the output function g in GNN. Simply speaking, the hidden state of the node v_t is obtained by updating the old hidden state with the newly obtained message. And we repeat this *message passing* algorithm for a specified number of times. After that, we could either perform node-level predictions, or reach the second phase of MPNN, which is called *readout* phase, where we could use a *readout* function R with learnable parameters to perform graph-level predictions, and the second phase of MPNN is the main difference from GNN.

$$\hat{y} = R(\{h_v^T | v \in G\}) \tag{3.3.3}$$

In this phase, we extract all newly update hidden states and create a final feature vector describing the whole graph.
In our work of **MPNN**, we use the following *message passing* or *neighborhood aggregation* scheme:

$$H^{i+1} = f(\tilde{A}H^i W^{i+1}) \tag{3.3.4}$$

where $H^0 = X$ is the input of the first layer, $X \in R^{N*D}$, N is the number of nodes of a graph, D is the dimension of the feature vector of each node, W^i is the matrix of trainable parameters of layer i, and function f is a non-linear activation function, in our case is ReLU. To consider the two main problems of GCN which is talked about in last section, we following Kipf and Welling [18], we normalize the adjacency matrix A such that the sum of the weights of the incoming edges of each node is equala to 1, and use matrix $\tilde{A}$ instead of matrix A.
We represent each country as a graph G, and the above model 3.3.4 is applied to all graphs $G^{(1)}, ..., G^{(T)}$. Given a model with K neighborhood aggregation layers, the matrix $\tilde{A}$ and $H^0, ..., H^K$ are specific to

a single graph, while the weight matrices $W^1, ...W^K$ are shared across all graphs. We concatenate the matrices $H^0, ..., H^K$ horizontally, we utilize skip connections from each layer to the output layer which consists of a sequence of fully-connected layers. Finally, we choose the mean squared error as loss function:

$$\mathcal{L} = \frac{1}{nT} \sum_{t=1}^{T} \sum_{v \in V} (y_v^{(t+1)} - \hat{y}_v^{(t+1)})^2 \tag{3.3.5}$$

where $y_v^{(t+1)}$ denotes the reported number of cases for region v at day $t+1$ and $\hat{y}_v^{(t+1)}$ denotes the predicted number of cases.

3.4 Long Short-Term Memory[Xianghe MA]

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture [28].
Recurrent neural networks (RNN) are a class of neural networks that are helpful in modeling sequence data. Derived from feed-forward networks, RNNs exhibit similar behavior to how human brains function. But unlike standard feed-forward neural networks, LSTM or RNN has feedback connections.

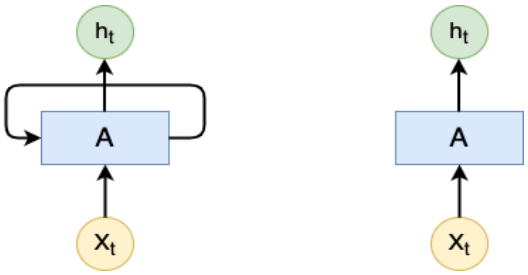

Figure 3.3: RNN and feed-forward Neural Network

In a feed-forward neural network, the information only moves in one direction, from the input layer, through the hidden layers, to the output layer. Feed-forward neural networks have no memory of the input they received, and it only considers the current input. It can't remember anything about what happened in the past except its train-

ing. But in a RNN the information cycles through a loop. When it makes a decision, it considers the current input and also what it has learned from the inputs it received previously(see Fig 3.3) .
But there are two major problems of RNN, exploding gradients or vanishing gradients which are caused by long-term dependency problem of inputs. Therefore, Long short-term memory (LSTM) is explicitly designed to avoid the long-term dependency problem. Long Short-Term Memory (LSTM) networks are an extension of RNN that extend the memory. The units of a LSTM are used as building units for the layers of a RNN, often called a LSTM network.
LSTM enables RNN to remember inputs over a long period of time because LSTM contains information in a memory. This memory can be seen as a gated cell, which means the cell decides whether or not to store or delete information during training, based on the importance it assigns to the information. The assigning of importance happens through weights, which are learned by the training. In a LSTM there are three gates: input, forget and output gate. These gates determine whether or not to let new input in (input gate), delete the information that is unimportant (forget gate), or let it impact the output at the current timestep (output gate). The gates in an LSTM contain sigmoid activations, which squish values between 0 and 1. The sigmoid activations are to update or forget data by multiplying by 1 or 0, causing values to be "kept" or be "forgotten". Then the problematic issues of exploding gradients and vanishing gradients are solved through LSTM because it keeps the gradients steep enough, which keeps the training relatively short and the accuracy high.
MPNN+LSTM In our work, given a sequence of graphs $G^{(1)}, ..., G^{(T)}$, we could obtain a sequence of representations $H^{(1)}, ..., H^{(T)}$, then we could combine our MPNN with LSTM by feeding these representations into LSTM so that we could capture the long range dependencies in these graphs. We use a stack of two LSTM layers. The new representations of the regions correspond to the hidden state of the last time step of the second LSTM layer. These representations are then passed on to an output layer. The framework of MPNN+LSTM

archtecture sees below figure 3.4.

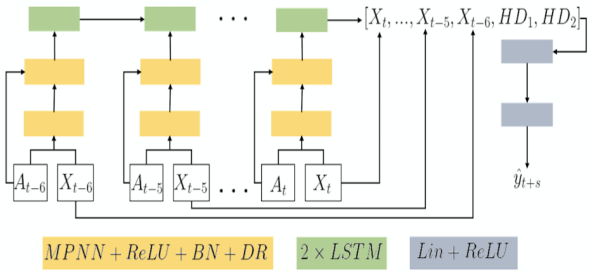

Figure 3.4: Framework of MPNN+LSTM architecture [1]

3.5 Transfer Learning Based on MAML[Xianghe MA]

Transfer learning (TL) is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem [29]. Two common transfer learning approaches are Develop Model Approach and Pre-trained Model Approach. The difference between these two methods is that Develop Model Approach needs to train a model from a related predictive modeling problem with an abundance of data, but Pre-trained Model Approach is that a pre-trained model needs to be chosen from the pool of candidate models. After having the appropriate model, transfer learning then reuse the model and tune the model to fit and achieve the new task.

Model-Agnostic Meta-Learning(MAML) [23], is a model and task-agnostic algorithm for meta-learning that trains a model's parameters such that a small number of gradient updates will lead to fast learning on a new task. Meta-learning, also known as "learning to learn", intends to design models that can learn new skills or adapt to new environments rapidly with a few training examples. Model-agnostic means that MAML is a more like a framework that provides a meta-learner for training base-learner.

In general machine learning problems, we train model parameters in the data set D_{train}, and test the model performance in D_{test}. In

MAML problem setting, we deal with Meta-sets δ. For each data set $D \in \delta$, D is divided into D_{train} and D_{test}, just like general machine learning problems. At the same time, Meta-sets δ is divided into multiple Meta-sets $\delta_{meta-train}$, $\delta_{meta-val}$, $\delta_{meta-test}$, as shown in Fig 3.5. Our goal is to train a "learning process" on $\delta_{meta-train}$. The process inputs a D_{train} and outputs a learner(e.g. classifier), which can achieve good fitness on the corresponding D_{test}.

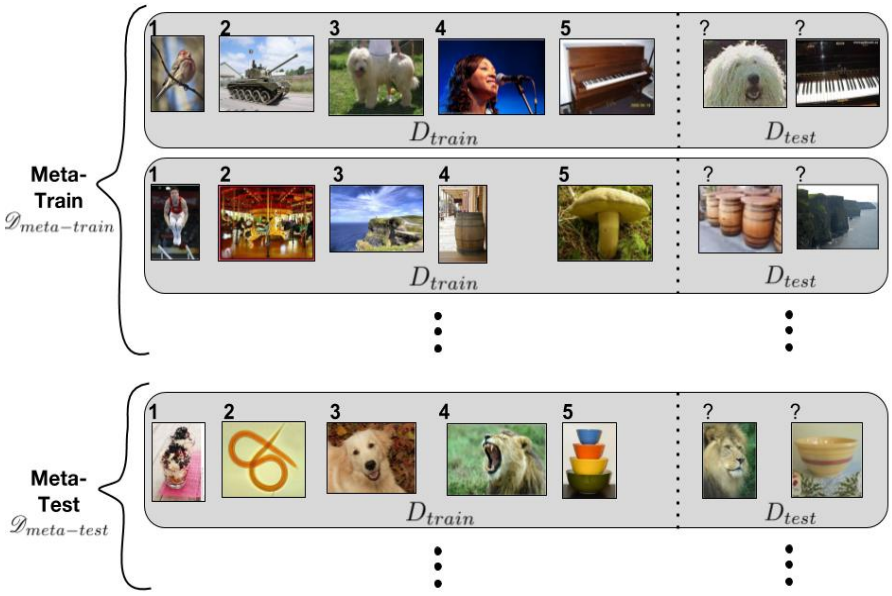

Figure 3.5: Example of MAML setup [30]

The goal of MAML is to find parameters θ that are sensitive to the tasks by changing the direction of the gradient descent (shown in Fig 3.6), and subtle changes to these parameters can cause great changes in the loss function. Therefore, when there are a small amount of training data, the gradient descent of these parameters can make the loss drop quickly and achieve a good fine-tune effect.

Algorithm of MAML is shown in Fig 3.7, the number of outer loop is specified by the hyperparameter *metatrain_iterations*. For each outer loop, sample the number of *meta_batch_size* tasks from the $\delta_{meta-train}$ as a batch (general machine learning tasks a batch is K samples, here is K tasks). For each inner loop, a θ_i' is calculated separately for each task and the corresponding *loss* is calculted at the same time.

MPNN+TL In our work, a model trained in the whole cycle of the

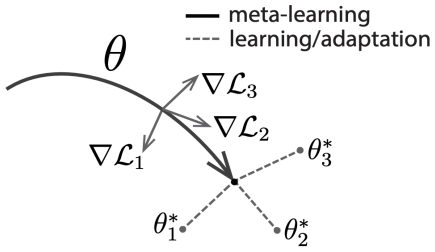

Figure 3.6: Diagram of MAML algorithm, which optimizes for a representation that can quickly adapt to new tasks [23]

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
6: Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
7: **end for**
8: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
9: **end while**

Figure 3.7: Algorithm of MAML [16]

epidemic can capture patterns of its different phases, which is missing from a new model working in a country at the start of its infection. Therefore to incorporate past knowledge from models running in other countries, we employ a transfer learning method based on Model-Agnostic Meta-Learning(MAML). In specific, our Meta train set $M_{tr} = D^{(1)}, ..., D^{(P)}$, denotes the data sets of p countries that we could use to obtain a set of parameters θ. Then these learnt parameters can be used to initilize model or the countries left out in the Meta test set M_{te}. And for each dataset $D^{(k)}, k \in 1, ...p$(each country) is divided into different training sets of increasing size. For each combination of these two, we train a different model. Then the set of tasks for a country k is $D^{(k)} = \{(Tr_{i,j}^{(k)}, Te_{i,j}^{(k)}) : 14 \geq i \geq T_{max}, 1 \geq j \geq dt\}$ where $(Tr_{i,j}^{(k)}, Te_{i,j}^{(k)})$ is a dataset(concluding train and test set) associated with country k where the train set comprises of the first i days

of the data and the task is to predict the number of cases in the j-th day ahead.

In the MAML algorithm, θ is randomly initialized and undergoes gradient descent steps during the MAML training phase. The algorithm of MPNN combined with MAML is shown below.

Algorithm: MPNN+TL

Input: $M_{tr}, M_{te}, \alpha, \alpha_m, n\_\text{epochs}$
Output: θ

1: Initialize θ randomly
2: for $D \in M_{tr}$ do
3: for $(Tr, Te) \in D$ do
4: for Batch $b \in Tr$ do
5: $\theta_t = \theta - \alpha \nabla_\theta \mathcal{L}(f_\theta(b))$
6: $\theta = \theta - \alpha_m \nabla_\theta \mathcal{L}(f_{\theta t}(Te))/|M_{tr}|$
7: for $(Tr, Te) \in M_{te}$ do
8: for Epoch $e \in n\_\text{epochs}$ do
9: for Batch $b \in Tr$ do
10: $\theta = \theta - \alpha \nabla_\theta \mathcal{L}(f_\theta(b))$
11: error $+= E(f_\theta(Te))$
12: return error$/|M_{te}|$

In each task, we minimize the loss on the task's train set for a specific θ_t, as shown on line 3-5 of the algorithm. Then, we use the emerging θ_t to compute the gradient with respect to θ in the task's test set as shown on line 6 of the algorithm. The standard update includes the gradient of θ_t and θ, which is the hessian matrix, as illustrated below.

$$\frac{\partial \mathcal{L}_T(f_{\theta t}(Te_{i,j}^{(k)}))}{\partial \theta} = \nabla_{\theta t} \mathcal{L}(f_{\theta t}(Te_{i,j}^{(k)})(I - \alpha \nabla_\theta^2 \mathcal{L} f_\theta(Tr_{i,j}^{(k)}))) \quad (3.5.1)$$

We drop the term that contains the hessian, because it was shown to have insignificant contribution in practice. Finally we train θ on the train set of M_{te} and test on its test set, as shown on lines 7-10 and 11 in the algorithm).

3.6 ARIMA[Tong YU]

ARIMA, short for 'Auto Regressive Integrated Moving Average' is actually a class of models that 'explains' a given time series based on its own past values, that is, its own lags and the lagged forecast errors, so that equation can be used to forecast future values. In model ARIMA(p, d, q), p denotes the number of lag observations included in the model, also called the lag order. d is the number of times that the raw observations are differenced, also called the degree of differencing, q denotes The size of the moving average window, also called the order of moving average.

To apply ARIMA model for forecasting, we plug in time series data for the variable of interest. First we need to specify the appropriate number of lags or amount of differencing to be applied to the data and check for stationarity. Then we output the results, which are often interpreted similarly to that of a multiple linear regression model.

Chapter 4

Experiments

4.1 Data[Xianghe MA]

We use data from Facebook as they release lot of datasets for researchers to build models to solve real life problems. The datasets that we choose are movement maps data show how many facebook location history collected from users phones moved from one area to another. The facebook researchers Maas et al. 2019 [31] worked out predictions about the spread of Covid-19 using the data from Facebook Data For Good program[1]. We use a pre-processed data that present the number of movement from one area to another area in each day in a country. The statistics are build on the administrative NUTS3 region levels. And we choose data from 4 European countries. They are England, France, Spain and Italy. The data is mainly cover the period from March 2020 to June 2020, when the COVID-19 epidemic swept Europe for the first time.

The data of new reported cases are gathered from open source data and World Health Organization Dashboard. A brief summary of our preprocessed data is demonstrated in the table 4.1.

A common phenomenon is that when an epidemic breaks out, the number of reported cases per day in each region is often uneven. The actual number of reported cases mostly depends on the detection capacity of hospitals and clinics and how many people plan to test the Covid-19 virus on that day. In this situation, the number of daily

[1]Facebook Data For Good public datasets https://dataforgood.facebook.com/, Sep 2021.

Table 4.1: Overview of the data

Country	ITALY	FRANCE	SPAIN	ENGLAND
Number of regions	103	94	52	151
Period	24/2-5/6	10/3-26/5	20/2-20/6	13/3-12/5
Avg new cases	20.6	5.5	38	14.2
Total cases	225486	40000	241591	130764
Avg internal mobility	91194	115944	118645	88669
Avg external mobility	328	553	106	334

cases doesn't reflect the actual local Covid-19 infection dynamics directly. Hospital Covid-19 testing capabilities usually vary with times and regions, thus sometimes the number of daily cases can not accurately represent the spreading of Covid-19 in the area. For example, when the city government makes a decision to conduct centralized testing in some suspected Covid-19 outbreak areas, a large number of confirmed cases are often reported in just a few days. This means that there is a sudden increase in confirmed cases due to the sudden change of detection level, which indicates that the time series data of reported cases are mutative due to factors other than virus transmission. Therefore the real transmission and infection of Covid-19 are often inconsistent with case reports. This inconsistency of data makes it difficult to improve the prediction accuracy of a Covid-19 spreading prediction model. Because each region in a country performs Covid-19 test varying a lot, there are many differences and inconsistencies in the new cases reported daily. Especially it shows from Figure 4.1 that the maximum difference goes larger than the mean and standard deviation among countries.

On the other hand, when a large number of external and internal population movement occur in a short time, the region usually has a higher probability of Covid-19 epidemic outbreak, it will result in more reported cases of infection. This hypothesis comes from the results of epidemiological observation on the spread of an airborne virus [5]. Based on this observation and hypothesis, we can build data structures to learn the hidden relationship between the number of case reports and population mobility, and then predict the transmission of

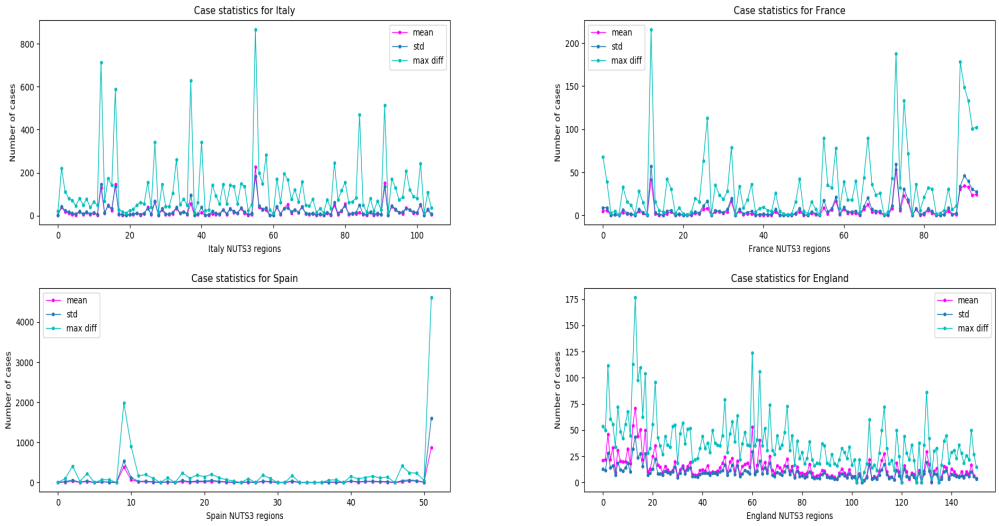

Figure 4.1: Mean cases, standard deviation, maximum difference of cases per day in each regions in Italy,France,Spain,England

Covid-19 virus in the future.

4.2 Graph construction[Hanmeng JIANG]

This research is based on the graph model $G = (V, E)$, therefore we use each country to present the node: V and $n = |V|$ denotes the number of nodes. To correspond the different period of one given country, we create the series of graphs, e.g, country G in date $t \rightarrow G^t$. A single date's mobility data is transformed into a weighted, directed, self-loop graph whose vertices represent the NUTS3 regions and edges capture the mobility patterns. For instance, the weight $w^t_{v,u}$ of the edge (v, u) from vertex v to vertex u denotes the total number of people that moved from region v to region u at time t. The mobility between administrative regions u and v at time t forms an edge, which multiplied by the number of cases c^t_u of region u at time t, provides a relative score expressing how many infected individuals might have moved from u to v. To be more specific, let $x^t_u = (c^{t-d}_u,...,c^t_u)^T \in R^d$ be a vector of node attributes, which contains the number of cases for each one of the past d days in region u. Here we don't just choose the previous one day to predicate, that is because it exists highly irregular between days of reported case, especially in decentralized

regions. By intuition, message passing over this network computes a feature vector for each region with a combined score from all regions, as illustrated below.

$$\mathbf{A}^{(t)}\mathbf{X}^{(t)} = \begin{bmatrix} w_{1,1}^{(t)} & w_{2,1}^{(t)} & \dots & w_{n,1}^{(t)} \\ w_{1,2}^{(t)} & w_{2,2}^{(t)} & \dots & w_{n,2}^{(t)} \\ \vdots & \vdots & \vdots & \vdots \\ w_{1,n}^{(t)} & w_{2,n}^{(t)} & \dots & w_{n,n}^{(t)} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1^{(t)} \\ \mathbf{x}_2^{(t)} \\ \vdots \\ \mathbf{x}_3^{(t)} \end{bmatrix} = \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \vdots \\ \mathbf{z}_3 \end{bmatrix}$$

A^t is the adjacency matrix of G^t and x^t is a matrix whose rows contain the attributes of the different regions. After computing, we get the $z_u \in R^d$ which represent the combination of the mobility within and towards region u with the number of reported cases both in u and in all the other regions. We also consider the inner-movement of a given region, which is $w_{u,u}$. And this could point out the evolution of the disease especially in the lockdown period when people's movements are limited. To visualize concretely how the representations are extracted from the message passing, following Figure 4.2 and equation shows that we use the linear combination to extract the information from graph: people from different regions move to region u, $\mathbf{x}$ represents a vector of past cases in that region.

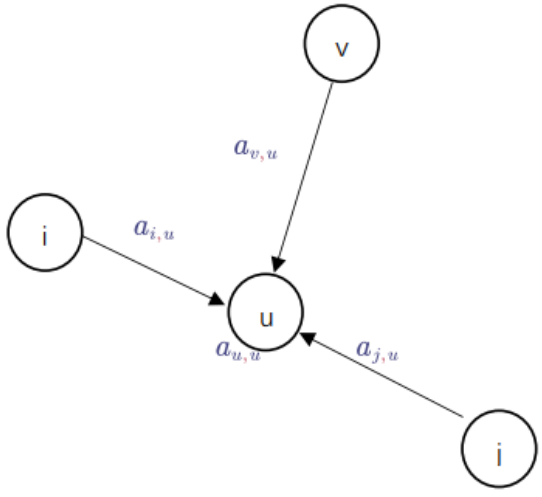

Figure 4.2: Message passing around region u

$$Z_u = (x_j a_{j,u} + x_i a_{i,u} + x_v a_{v,u}) + x_u a_{u,u} \tag{4.2.1}$$

where $Z_u \in R^d$ represents the estimate of the number of new latent cases in region u, and it is divided into the cases received from other regions and the new cases caused due to mobility inside $u(x_u a_{u,u})$.

4.3 Experimental setup[Tong YU]

In the experiment, we organize the obtained observation data into time series data according to the unit of day. Each country has at least 60 days data. When training a model, the data from the first day to T^{th} day are selected as the training data. The number of cases will then be predicted on dt days from day T. Considering that the model needs a certain amount of data training process to predict the future, we choose to learn the relationship between new reported cases and population movement for at least two weeks, and then start to make prediction for the next date. Here, we set the time window of training data $T \geq 15$, initially set it to 15, and then increase T by 1 in each training cycle to increase the length of training time window. For the selection of prediction time window, we consider that the future is predicted in days. The larger the prediction time span, the more inaccurate the prediction result will be. Meanwhile we hope that the prediction for a long term in the future will also have a certain accuracy, so that we can provide reasonable data support for longer-term epidemic prevention and control measures according to the predictions. Therefore, we set the maximum span of the prediction time window to 14 days, that is dt<14. Considering that the outbreak and control of the epidemic is often a short period of several months, and the amount of data observed for prediction will be relatively limited, thus for each dt we will train a model separately based on a fixed training window, and let the model specifically predict the news cases in the future dt period to obtain higher prediction accuracy. On the

Table 4.2: Experimental setup for MPNN

Batch Size	Dropout rate	Optimizer(LR)	Epochs	Hidden units
8	0.5	$Adam(10^{-3})$	300	64

other hand, we also set up validation data sets to cross validate the model during training. We select the data within $T - 10$ days with step size is two days as the validation set.

We choose train epochs for MPNN model to be at most 300. Early stop may occur after 100 epochs. We choose the Adam algorithms to be our optimizer for MPNN, and its learn rate is 10^{-3}. Another important hyper-parameter of MPNN model is hidden units. It is used for neighborhood aggregation layer. The number of its hidden units is set to be 64. LSTM also have 64 hidden units in the MPNN+LSTM model. The other hyper-parameters are demonstrated in the table 4.2.

We use Pytorch framework to implement our models. For neural network models we use mean squared error from ground truth and predictions to be loss function. And we compare each model by computing the absolute error of predictions corresponding to ground truth labels.

$$Loss = \frac{1}{ndt} \sum_{t=T+1}^{T+dt} \sum_{v \in V} (\hat{y}_v^{(t)} - y_v^{(t)})^2 \tag{4.3.1}$$

$$Error = \frac{1}{ndt} \sum_{t=T+1}^{T+dt} \sum_{v \in V} |\hat{y}_v^{(t)} - y_v^{(t)}| \tag{4.3.2}$$

4.4 Baselines[Hanmeng JIANG]

We compare the classic models against the following baselines and benchmark methods, which have been applied to the problem of COVID-19 forecasting:

AVG: The average number of cases in the specific region up to the

Table 4.3: Average error for dt in 3,7 and 14 in each region

Model	Up to next 3 Days				Up to next 7 Days				Up to next 14 Days			
	England	France	Italy	Spain	England	France	Italy	Spain	England	France	Italy	Spain
AVG	9.65	8.40	20.38	44.50	9.59	9.15	21.23	46.27	11.00	8.25	22.19	46.53
LAST DAY	7.01	7.37	16.80	32.88	7.12	7.17	18.09	36.86	8.26	7.63	19.89	42.55
AVG WINDOW	6.28	6.54	15.57	32.09	7.14	6.05	16.21	35.77	8.24	6.94	18.95	41.89
PROPHET	10.28	10.14	24.51	54.29	12.05	11.17	27.08	61.85	15.94	14.30	32.79	79.02
ARIMA	13.44	10.51	35.08	40.19	14.23	10.13	36.95	41.55	15.47	10.78	39.47	45.12
LSTM	9.00	7.84	22.81	51.04	8.65	7.95	22.07	48.89	9.04	7.71	22.49	46.98
TL_BASE	9.45	7.50	19.00	42.01	12.30	9.04	22.85	51.94	13.08	12.04	24.74	59.21
MPNN	6.10	5.86	14.19	35.56	6.77	5.75	15.25	38.31	7.88	6.90	17.75	43.69
MPNN+LSTM	6.11	6.08	15.26	32.85	6.50	6.98	16.26	34.41	6.99	7.21	16.85	35.01
MPNN+TL	**5.94**	**5.69**	**13.88**	**29.23**	**6.13**	**6.85**	**14.21**	**31.28**	**6.51**	**5.83**	**16.45**	**34.28**

time of the test day

AVG WINDOW: The average number of cases in the past D days for the specific region where D is the size of how long we want to use(window).

LAST DAY (Kapoor et al. [32]): The number of cases in the previous days is the prediction for the next days.

LSTM (Chimmula and Zhang [33]): A two-layer LSTM that takes as input the sequence of new cases in a region.

ARIMA (Kufel [34]): A simple autoregressive moving average model where the input is the whole time-series of the region up to before the testing day for the previous week.

PROPHET (Mahmud [35]): A state-of-the-art forecasting model for various types of time series [2].The input is similar to ARIMA.

TL_BASE: A MPNN that is trained on all data from the three countries and the train set of the fourth (concatenated), and tested on the test set of the fourth. This serves as a baseline to quantify the usefulness of the transfer learning method.

Then we use these different methods to predict next 3 days, 7 days and 14 days separately, and get the following results as shown in Table 4.3 and Fig 4.3.

Here we note that since we only rely on the number of confirmed cases, we can not utilize these models to work with recovery,deaths

[2]https://github.com/facebook/prophet

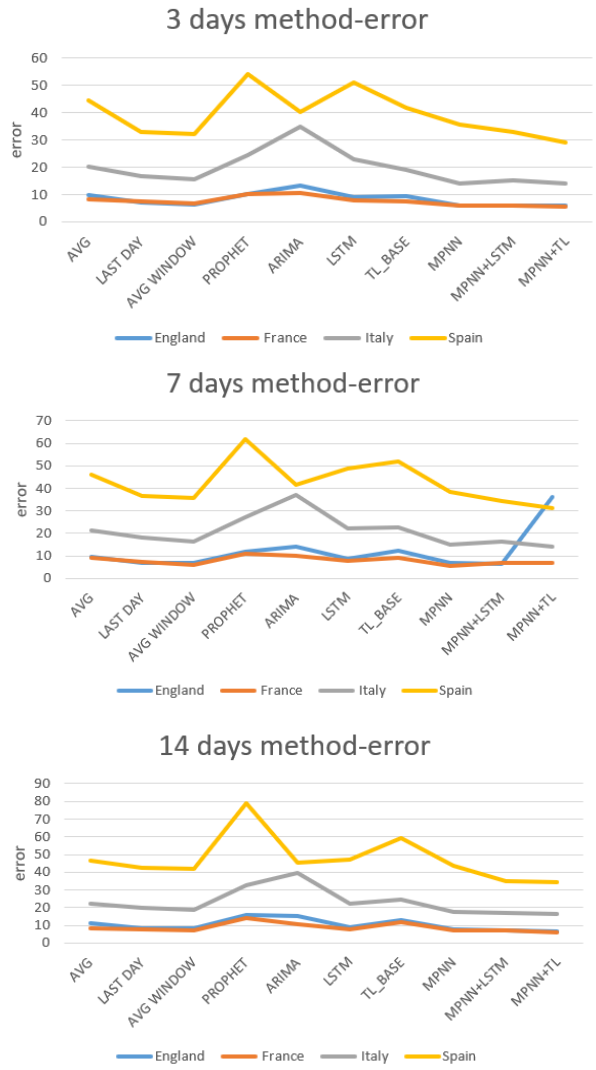

Figure 4.3: Predication error of different methods in 3 days, 7 days and 14 days

and others.

4.5 Transfer learning MPNN[Xianghe MA]

The predictions in three time shifts ahead compared to ground truth in different countries are illustrated in Fig 4.4 (e.g. 1 day ahead, 7 days ahead and 13 days ahead).

In order to evaluate the accuracy of the prediction made by each region of the four countries relative to the real data, we randomly selected to forecast the new cases on 08/05 which is two weeks earlier than the training data set before 24/04. Then compared the predic-

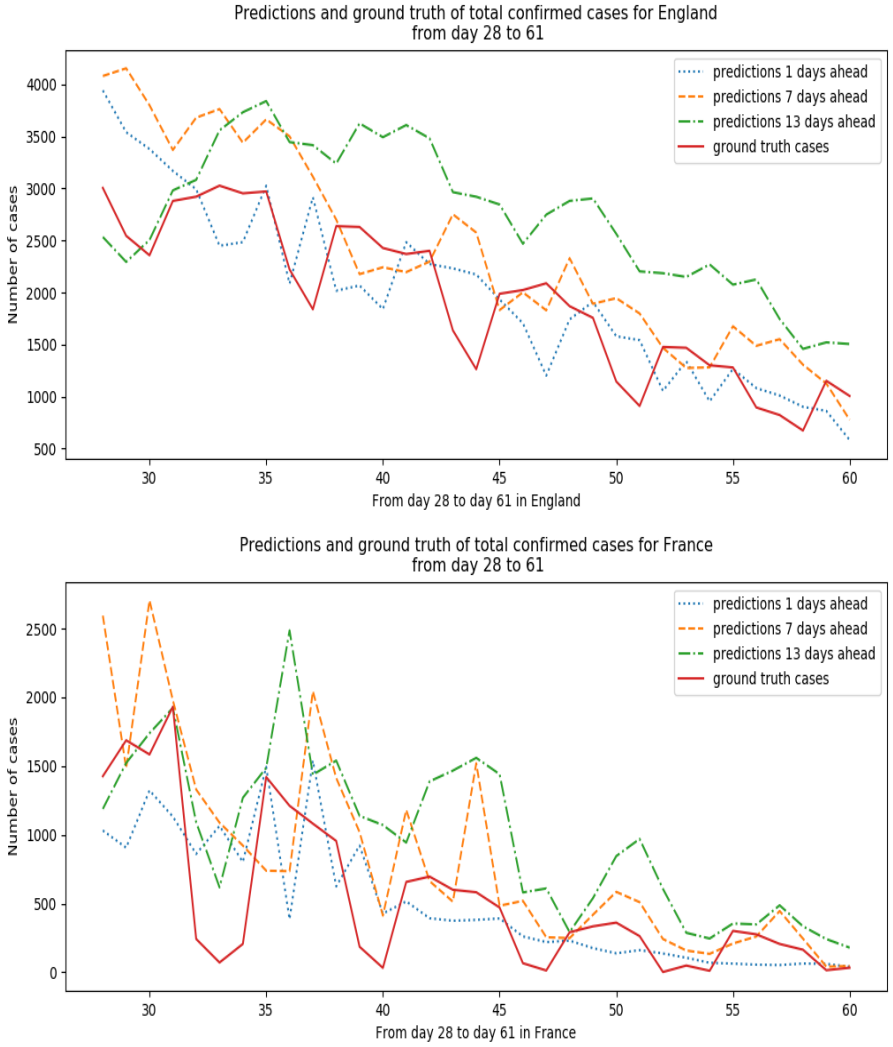

Figure 4.4: Predictions and ground truth of total confirmed cases of transfer learning MPNN in France and England

tion results with the real confirmed cases data of each region. The comparison is in figure 4.5. From the linear chart of the number of cases in each region on 08/05, we can see that the predicted results of the England and Italy fluctuate similar to the real value. However, the predicted trend of France and Spain is quite different from the real data. It shows the prediction accuracy is also unstable when making long-term prediction.

On the other hand, by summarizing the case data and forecasts of various regions in a country and drawing them into time series graphics, we can see that the short, medium and long-term prediction results of MPNN + TL model maintain good stability. Although there may

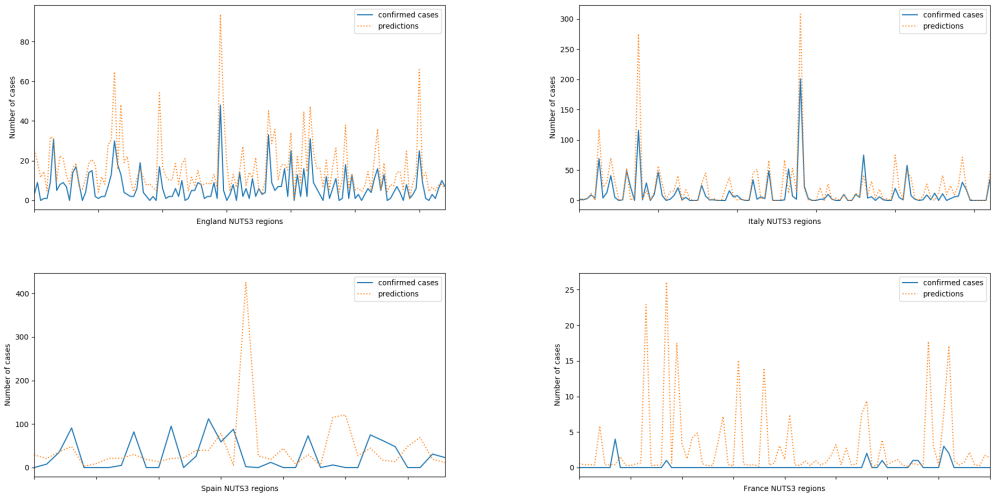

Figure 4.5: Predictions and ground truth confirmed cases of transfer learning MPNN at 08/05 in 4 countries. Predictions are made 2 weeks ahead of train set.

be large errors in the results of one day, the overall trend predicted in time series is consistent with the trend of real daily new cases, this can be seen from figure 4.4.

Chapter 5

Conclusion[Xianghe MA, Hanmeng Jiang]

In general, the baseline models and the transfer learning MPNN have good performance in predicting the new cases of COVID-19, moreover the comprehensive performance of the transfer learning MPNN model based on graph neural network is slightly better than other models. The factors affecting virus transmission are complex and diverse in the real world. This disease usually delays symptoms for several days and Covid-19 is an mRNA virus with high mutation probability. At present, it has produced a lot of mutated virus versions. Mutated viruses usually show different symptoms and incubation periods, especially their infectious power is very different. Different countries and regions have different population densities, different social cultures, and their proportion of main virus variants is also different. These differences also affect the efficiency of transfer learning, for example, the transmission ability of delta Covid-19 variants is much stronger than that of the original COVID-19. When training set is trained by the original COVID-19 virus epidemic data, then transfer to new variants of the COVID-19 virus, it may cause great error. This is one of the reasons why we can see from the results that the effect of transfer learning is very good in predicting 08/05 cases in Britain and Italy, but poor performance in France and Spain.

In this project we presented the model for COVID-19 forecasting which based on people's movement so that the government may take

effective measures to control the epidemic then find the balance point between public health,economic and stability. This model's kernel idea is using GNNs. Specifically, we transferred human mobility data to a graph where nodes correspond to regions and edge weights to measures of human mobility between their end-points.Then, we derive variants of the family of MPNNs to generate representations for the regions based on their interactions with each other. Besides,considering different countries might be in different periods of the epidemic, we also propose to transfer a well-performing disease spreading model not only in other regions, countries, phases even different continents. Experiments conducted on data from England, Spain, Italy and France show that our model does a better job than traditional or recent approaches in predicting the number of daily new COVID-19 cases.
Since this model works in some terms, we would like to do the further research in predicating via other ways, for example, social distancing, friendship, religion. And calculating the probability of people's movement could give the policymakers a broader sight. Furthermore, we hope to accelerate the computing speed which means this model may be improved by hyper-parameter estimation.

Bibliography

[1] Panagopoulos, G., Nikolentzos, G., Vazirgiannis, M. (2021). Transfer Graph Neural Networks for Pandemic Forecasting. Proceedings of the AAAI Conference on Artificial Intelligence, 35(6), 4838-4845. Retrieved from https://ojs.aaai.org/index.php/AAAI/article/view/16616

[2] Gavenčiak, Tomáš, et al. "Seasonal variation in SARS-CoV-2 transmission in temperate climates." medRxiv (2021).

[3] Han, Jie Zhang, Xue He, Shanshan Jia, Puqi. Can the coronavirus disease be transmitted from food? A review of evidence, risks, policies and knowledge gaps. Environmental Chemistry Letters. 19. 1-12. 10.1007/s10311-020-01101-x (2020).

[4] Kapoor, Amol, et al. "Examining covid-19 forecasting using spatio-temporal graph neural networks." arXiv preprint arXiv:2007.03113 (2020).

[5] Colizza, Vittoria, et al. "Prediction and predictability of global epidemics: the role of the airline transportation network." arXiv preprint q-bio/0507029 (2005).

[6] Pham, Quoc-Viet, et al. "Artificial intelligence (AI) and big data for coronavirus (COVID-19) pandemic: A survey on the state-of-the-arts." arXiv preprint arXiv:2107.14040 (2021).

[7] Lorch, Lars, et al. "A spatiotemporal epidemic model to quantify the effects of contact tracing, testing, and containment." (2020).

[8] Flaxman, Seth, et al. "Estimating the effects of non-pharmaceutical interventions on COVID-19 in Europe." Nature 584.7820 (2020): 257-261.

[9] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.

[10] Chimmula, V. K. R.; and Zhang, L. Time Series Forecasting of COVID-19 transmission in Canada Using LSTM Networks. Chaos, Solitons Fractals 109864.(2020).

[11] Ma, R., Zheng, X., Wang, P. et al. The prediction and analysis of COVID-19 epidemic trend by combining LSTM and Markov method. Sci Rep 11, 17421 (2021).

[12] Kufel, Tadeusz. "ARIMA-based forecasting of the dynamics of confirmed Covid-19 cases for selected European countries." Equilibrium. Quarterly Journal of Economics and Economic Policy 15.2 (2020): 181-204.

[13] Chaurasia, V., Pal, S. COVID-19 Pandemic: ARIMA and Regression Model-Based Worldwide Death Cases Predictions. SN COMPUT. SCI. 1, 288 (2020). https://doi.org/10.1007/s42979-020-00298-6.

[14] Mahmud, S. Bangladesh COVID-19 Daily Cases Time Series Analysis using Facebook Prophet Model. Available at SSRN 3660368 (2020).

[15] Scarselli, Franco, et al. "The graph neural network model." IEEE transactions on neural networks 20.1 (2008): 61-80.

[16] Gao, Junyi, et al. "STAN: spatio-temporal attention network for pandemic prediction using real-world evidence." Journal of the American Medical Informatics Association 28.4 (2021): 733-743.

[17] Kapoor, Amol, et al. "Examining covid-19 forecasting using spatio-temporal graph neural networks." arXiv preprint arXiv:2007.03113 (2020).

[18] Kipf, T. N.; and Welling, M. 2017. Semi-supervised classi- fication with graph convolutional networks. In 5th Interna- tional Conference on Learning Representations.

[19] Xiang Yu, Siyuan Lu, Lili Guo, Shui-Hua Wang, Yu-Dong Zhang, ResGNet-C: A graph convolutional neural network for detection of COVID-19, Neurocomputing, Volume 452, 2021, Pages 592-605,ISSN 0925-2312.

[20] Mudiyanselage, T.B., Senanayake, N., Ji, C., Pan, Y., Zhang, Y. (2021). Covid-19 Detection from Chest X-ray and Patient Metadata using Graph Convolutional Neural Networks. ArXiv, abs/2105.09720.

[21] Lawrence, N. D., and Platt, J. C. 2004. Learning to learn with the informative vector machine. In Proc. of the 21st ICML. Banff, Alberta, Canada: ACM.

[22] Schwaighofer, A.; Tresp, V.; and Yu, K. 2005. Learning gaussian process kernels via hierarchical bayes. In NIPS 17.

[23] Chelsea Finn, Pieter Abbeel, Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. Proceedings of the 34th International Conference on Machine Learning, PMLR 70:1126-1135, 2017.

[24] Tarun Naren, Yuanda Zhu, and May Dongmei Wang. 2021. COVID-19 diagnosis using model agnostic meta-learning on limited chest X-ray images. In Proceedings of the 12th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics (BCB '21). Association for Computing Machinery, New York, NY, USA, Article 42, 1–9.

[25] Adapting Meta Knowledge with Heterogeneous Information Network for COVID-19 Themed Malicious Repository Detection. Yiyue Qian, Yiming Zhang, Yanfang Ye, Chuxu Zhang. Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence Main Track. Pages 3684-3690.

[26] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang and P. S. Yu, "A Comprehensive Survey on Graph Neural Networks," in IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 1, pp. 4-24, Jan. 2021, doi: 10.1109/TNNLS.2020.2978386.

[27] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., Dahl, G. E. (2017). Neural Message Passing for Quantum Chemistry. Retrieved from http://arxiv.org/abs/1704.01212

[28] Sepp Hochreiter; Jürgen Schmidhuber (1997). "Long short-term memory". Neural Computation. 9 (8): 1735–1780. doi:10.1162/neco.1997.9.8.1735. PMID 9377276. S2CID 1915014.

[29] West, Jeremy; Ventura, Dan; Warnick, Sean (2007). "Spring Research Presentation: A Theoretical Foundation for Inductive Transfer". Brigham Young University, College of Physical and Mathematical Sciences. Archived from the original on 2007-08-01. Retrieved 2007-08-05.

[30] Ravi, Sachin and H. Larochelle. "Optimization as a Model for Few-Shot Learning." ICLR (2017).

[31] Maas, Paige, et al. "Facebook Disaster Maps: Aggregate Insights for Crisis Response Recovery." KDD. Vol. 19. 2019.

[32] Kapoor, A.; Ben, X.; Liu, L.; Perozzi, B.; Barnes, M.; Blais,M.; and O'Banion, S. 2020. Examining COVID-19 Forecasting using Spatio-Temporal Graph Neural Networks.In 16th International Workshop on Mining and Learning with Graphs.

[33] Chimmula, V. K. R.; and Zhang, L. 2020. Time Series Forecasting of COVID-19 transmission in Canada Using LSTM Networks. Chaos, Solitons Fractals 109864.

[34] Kufel, T. 2020. ARIMA-based forecasting of the dynamics of confirmed Covid-19 cases for selected European countries. Equilibrium. Quarterly Journal of Economics and Economic Policy 15(2): 181–204.

[35] Mahmud, S. 2020. Bangladesh COVID-19 Daily Cases Time.S eries Analysis using Facebook Prophet Model. Available at SSRN 3660368.