Universiteit Leiden

Master Computer Science

Automated machine learning for COVID-19 forecasting

Name:	Jaco Tetteroo
Student ID:	s 1284819
Date:	04/11/2021
Specialisation:	Artificial Intelligence
1st supervisor:	Dr. Mitra Baratchi
2nd supervisor:	Prof. dr. Holger H. Hoos

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

Accurate forecasting of the spread of pandemics is necessary for policy makers to adequately respond to it. As a response to the COVID-19 outbreak, various sophisticated epidemic and machine learning models were deployed to take on this task. These models, however, rely on expert knowledge, carefully selected architectures and detailed data that is often only available for specific regions. Automated machine learning (AutoML) tackles this issue by automatically creating pipelines in a data-driven manner, resulting in high quality predictions. In this work we adapt the AutoML framework of auto-sklearn to the time series forecasting task. We compare two methods, a multi-output and a repeated single-output, for multi-step-ahead forecasting. We also study the usefulness of open mobility data sets published by Apple and Google to complement the open incidence data set of the ECDC. To combat concept drift, we experiment with three drift adaptation strategies, refitting our models on part of the data, the full data, or retraining the models completely. We compare our methods with six baselines over two sets, a global set composed of 58 countries around the world and a European set composed of 26 countries. We evaluate and compare the performance of methods in early, intermediate and late forecasting scenarios. We find that a simple persistence baseline is a strong competitor for this task. Our results over three scenarios separated in time show that the comparative performance of our models increase as more data becomes available. In the late forecasting scenario, our best method, a multi-output ensemble refitted on recent data and using Google mobility data alongside incidence data, outperforms all other methods and baselines for each country.

Contents

1 Introduction 4

2 Related work 8
2.1 Compartmental models . . . 8
2.2 Bayesian inference . . . 9
2.3 Regressive models . . . 10
2.4 Automated machine learning . . . 11

3 Problem statement 13

4 Proposed methods 15
4.1 Data sets . . . 15
4.2 Methods . . . 17

5 Experiments 23
5.1 Scenarios . . . 24
5.2 Experimental setup . . . 24
5.3 Methods and baselines . . . 26

6 Results 30
6.1 Early pandemic forecasting . . . 30
6.2 Intermediate pandemic forecasting . . . 36
6.3 Late pandemic forecasting . . . 43

7 Conclusion and future work 50

8 Appendix 58

Chapter 1

Introduction

It has become apparent that disease outbreaks can have a major impact on society globally. In December 2019, a coronavirus disease (COVID-19), caused by the severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), emerged in the city of Wuhan, China. Despite efforts in containing the disease, by the end of January World Health Organization estimated the virus to be internationally spread, advising governments to prepare for active surveillance and case management [1]. Governments have stood for the challenge to implement efficient containment strategies. There is an important trade-off here, as strict containment strategies are necessary to restrict the spread of COVID-19, but they come with repercussions. Economic sectors like restaurants and hospitality [2] have crippled due to a lack of demand, and food and agriculture [3] due to a lack of workforce. In order for policy makers to respond adequately to the pandemic, accurate forecasting of the spread of epidemics is necessary.

The oldest but still popular way to give insights in the dynamics of disease outbreaks are compartmental models, such as the SIR model [4]. The population of a region to which such a model is applied gets assigned to compartments. Through transition functions people may transgress through compartments. In case of the SIR model the compartments are named susceptible, infectious and removed. People within the susceptible compartment are healthy and can be infected by people in the infectious compartment. People in the removed compartment are either recovered and deemed immune, or have died from the disease. The transition functions consist of rates that are carefully selected by domain experts. To this end, approximations are made of the transmissibility, incubation time and recovery period. Compartmental models are great to simulate the course of epidemics under controlled circumstances. A disadvantage is that they require the unrealistic assumption that the full population is homogeneously mixed, such that every individual has an equal amount of contact with every other individual. To alleviate this issue compartmental models may be extended with contact networks. These networks use individuals or sub-populations as nodes which are connected when these are in contact with each other [5]. With these networks a form of spatial awareness is introduced, making the simulations more informative. Contact networks may be constructed using several types of data, like contact tracing networks where infected people are asked to share their recent contacts or movement networks where the travel patterns of people are captured [6]. In the case of contact tracing networks a major disadvantage is that the infection process drives the construction of the network, meaning large parts of the population are not mapped. Additionally, constructing such network data sets is labour intensive, and individuals within these networks are not anonymous.Movement networks map the movement of all individuals in a population, disregarding whether or not they are ill. The data for these networks is typically retrieved from mobile phone locations, making it possible to access large quantities of anonymised data. As part of Data for Good initiatives, large tech companies like Apple, Facebook, Foursquare and Google have provided mobility data sets. Nevertheless, the data in these sets is often not detailed enough to create realistic networks. In the case of Apple, Facebook and Google the data recorded contains increases or decreases in movement within the population. This data is highly aggregated, making it impossible to create networks on the individual or even regional level. The Foursquare data set is more detailed, storing venue locations and the number of people moving between such locations over a period of time, but is only available for specific regions, such as the country the United States of America or states within this country.

A different approach to disease forecasting involves analysis of regressive models. These include the widely used ARIMA auto-regressive models and deep learning methods such as LSTMs. As many country governments across the world share data of the number of cases and deaths, it becomes possible to make forecasts for many countries at once. These models have the advantage over the aforementioned disease models that they do not rely on disease transmission variables. Instead, they are data-driven, meaning they can make forecasts depending on underlying patterns in the data. Still, these regressive models introduce new challenges. Constructing machine learning pipelines consists of many steps, such as data pre-processing, feature extraction and selection, model fitting, and the construction of model ensembles. For each of these steps choices need to be made and corresponding hyperparameters need to be tuned. Well tuned pipelines may yield faster training of models and higher forecasting quality. However, for each step many options are available and each of these have their own hyperparameter search space, resulting in a large combined algorithm and hyperparameter search space. Manual tuning often relies on simplifying assumptions, which may not fully capture the underlying characteristics of the data. Utilising the strengths of automated machine learning (AutoML), we can find solutions to this challenge. AutoML enables users to construct full classification or regression pipelines without requiring expert machine learning knowledge. AutoML is proven to extract competitive and high quality models automatically, often outperforming manually tuned models.

In this work, we adapt the AutoML framework *auto-sklearn* [7] to the task of COVID-19 forecasting. This framework fully automates the constructing and tuning of regression pipelines and supports multi step-ahead prediction since a recent update. We apply various disease and mobility data sets, analysing how they can best be used. There are two main challenges to this approach. Firstly, as auto-sklearn was not designed specifically for the task of time series regression, adjustments must be made to allow for suitable input windows and forecasting horizons. Secondly, because the data is collected while the pandemic progresses, inconsistencies in data quality may occur within the time series. Also, since the virus mutates over time, the underlying concept generating the data changes. When this concept drifts away, adaptations are needed to ensure high quality forecasts.

Our contributions are as follows:

- We adapt auto-sklearn to perform on time-series data for the purpose of forecasting COVID-19 mortality.
- We explore and evaluate the use of various open disease and mobility data sets and investigate if they can improve the COVID-19 forecasting task. To this end we introduce two methods of multi-step ahead forecasting: *multi-output* and *repeated single-output*.
- We experiment with different combinations of mortality data features and mobility data features derived from open data sets published by Apple, Google and the ECDC.
- We expose our ensembles to various concept drift adaptation techniques, and show that high forecast quality may be achieved by re-tuning ensembles created on older data with new data. This way we can outperform all baselines.
- We compare our methods against 6 baselines in terms of root mean squared error on a global dataset composed of data from 58 countries.
- We experiment on three scenarios, different in time and progression of the pandemic.
- We make our work and code publicly available[1].

The rest of this thesis is structured as follows. Chapter 2 covers related work on epidemic forecasting and AutoML. In Chapter 3 we provide a formal definition of time series forecasting. In Chapter 4 we give an overview of the used data sets and describe our implementation of and adjustments to auto-sklearn. Our

[1] `https://github.com/jacotetteroo/AutoML4COVID-19`

experimental set-up and evaluation is described in Chapter 5, of which we discuss the results in Chapter 6. We conclude in Chapter 7, suggesting directions for future work.

Chapter 2

Related work

2.1 Compartmental models

Traditionally epidemics are charted using compartmental models, like the SIR model [4]. This model splits the population of individuals in different compartments based on their health status. At each time step, the flow of individuals transitioning from one compartment to the other is described by differential equations, as shown in Equation 2.1. The S compartment holds healthy people that are susceptible to being infected. The I compartment holds infected and infectious people. The R compartment, finally, holds all people removed from the simulation, either because they recovered and gained immunity, or because they died. The total of the compartments is the total population N, which does not change over time. The transition parameters determine how many individuals move between compartments. The contact rate β is a measure that captures how many people individuals meet. The recovery rate γ captures how long it takes for people to recover.

$$\begin{aligned} \frac{dS}{dt} &= -\frac{\beta IS}{N}, \\ \frac{dI}{dt} &= \frac{\beta IS}{N} - \gamma I, \\ \frac{dR}{dt} &= \gamma I. \end{aligned} \tag{2.1}$$

Given more knowledge about the to be simulated disease, more complex compartmental models may be created by adding compartments. The SEIR model [8] extends the SIR model by injecting the exposed compartment, holding people infected by the disease but not yet capable of infecting others. Basic compartmental models require the unrealistic assumption that the full population is homogeneously mixed, such that every individual has an equal amount of contact with every other individual. To alleviate this issue compartmental models may be extended with contact networks. Liu et al. [9] argue that the assumption of homogeneous contact needed for the compartmental models is not realistic enough. They use a multi-layered contact network – where each layer entails a mode of contact – and a SIR model to simulate the propagation of flu and show that this approach gives more insights in underlying dynamics of the spread of diseases. Balcan et al. [10] similarly used a multi-scale network to simulate an influenza-like illness. Instead of individuals, they used sub-populations as nodes and gravitational flow derived from commuting and flight data as weights for the edges. This introduces a form of spatial awareness to the compartmental models, making the simulations more informative. In order to create realistic contact networks detailed mobility data sets are compulsory. Ideally, data sets encompass the entire population of a region, detailing where and how people have come in contact with each other. In reality, data sets are snapshots, often samples of a population, and recorded interactions are not enriched with duration or intensity [6]. Contact networks where individuals are simulated as a basis for the spread of diseases are called agent-based networks. To create agent-based networks one needs data sets containing movement patterns of individuals. For some regions such data sets are available; Aleta et al. [11] for instance create an agent-based network using a data set containing place visits published by Foursquare to simulate the spread of COVID-19 through a synthetic population in the Boston metropolitan area. In [12] Zhang et al. created a framework to visualise the effects of mobility on the spread of COVID-19. For this they used detailed trajectory data to simulate the mobility within the population. While for some countries the mobility data is detailed enough to create realistic contact networks, for most this is not the case. In our work, we make predictions for a large number of countries. Instead of detailed mobility data on the individual level, we use aggregated mobility data on a national level. This is not nearly detailed enough to construct contact networks but can still inform our data driven models.

2.2 Bayesian inference

Often, epidemic models suffer from missing or censored data. The effect of mobility, or the use of government interventions on the spread of the disease may be unknown. These effects can be inferred from data when using Bayesian inference. This is the reason why the Bayesian framework is quite useful for epidemic modeling [13]. This framework aims to find the posterior distribution for unknown parameters θ given observational data D as shown in Equation 2.2. Here $f(\theta)$ is the prior knowledge and $f(D|\theta)$ is the likelihood. As epidemics are continuous processes but available data is most often discrete, the Bayesian inference can account for the missing time steps.

$$f(\theta|D) \propto f(D|\theta)f(\theta) \tag{2.2}$$

Applied to COVID-19 Flaxman et al. [14] used a Bayesian hierarchical model to estimate the effect of governmental countermeasures . In their work they created a death forecast model that depends on the infections of the previous day and an infection forecast model that depends on the infections of the previous day as well as on the reproduction number. They estimate the reproduction number as the effect of government interventions aimed to reduce the spread of the disease. Using Bayesian inference with daily deaths observational data D the effects of interventions θ are approximated, which are in turn used to create the models to forecast the infections and deaths. This method was able to forecast early during the epidemic for multiple European countries. While the Bayesian inference is a data driven procedure, the model proposed by Flaxman et al. relies on a lot of manually tuned parameters. The distribution modelling the daily deaths, the infection-fatality ratio, the infection-to-death distribution and the distribution modelling the infections need to be carefully selected. Manual setting of parameters is typically done based on simplifying assumptions that might not hold in all scenarios. In our work, the underlying characteristics of the pandemic are treated as unknown and we avoid setting parameters but rely mainly on fully automated and data-driven approaches for modelling the underlying distribution.

2.3 Regressive models

Autoregression. Another classic approach is to use autoregressive methods. An autoregressive model is a regression model where the input variables are observations from previous time steps. It is referred to as AR(p), of order p, and is shown in Equation 2.3. Here, ϕ are parameters that are set by fitting the model as a linear regression on the training data, and ϵ is noise following a normal distribution.

$$y_{t+1} = c + \phi_1 \cdot y_t + \phi_2 \cdot y_{t-1} + ... + \phi_p \cdot y_{t-p+1} + \epsilon_t \tag{2.3}$$

For this model the only parameter to tune is p. This model can be extended by using integrated moving averages of the observed errors, which is an ARIMA(p, d, q) model, with p the order of the autoregression, d the degree of first differencing and q the order of the moving average. We show it in Equation 2.4. Here y'_t is the differenced series and is the combination of both lagged values of y and lagged errors ϵ of which χ are the coefficients [15]. In the ARIMA model parameters p, d and q can be tuned to make the time series stationary and control how many time steps are used for fitting.

$$y'_{t+1} = c + \phi_1 \cdot y'_t + ... + \phi_p \cdot y'_{t-p+1} + \chi_1 \cdot \epsilon_t + ... + \chi_q \cdot \epsilon_{t-q+1} + \epsilon_t \tag{2.4}$$

ARIMA models were successfully deployed to forecast COVID-19. Kumar et al. [16] used the ARIMA model to analyse the trend of 15 countries during the first three months of the pandemic. Alzahrani et al. [17] compared the ARIMA model with the simpler AR, MA and ARMA models making forecasts for four weeks for Saudi Arabia and found that ARIMA outperformed the others. They tuned p, d and q parameters of the ARIMA model using a grid search and evaluated performance gain via the Akaike Information Criterion [18]. Chakraborty and Gosh [19] extended an ARIMA model by adding a wavelet transformation on the residuals of the model. This improves the forecasts and is tested for Canada, France, India, South Korea, and the UK on a forecasting range of ten days.

Deep learning. Deep learning was first applied in the epidemiology field just three years ago . Wu et al. [20] predicted flu in the United States using a combination of a CNN, RNN and residual links. They were able to achieve robust improvement over autoregressive models using multiple real-world data sets. Aiken et al. [21] compared autoregressive models with a GRU RNN to predict flu prevalence. They found that on larger prediction horizons the RNN was able to achieve significantly lower RMSE. Fu et al. [22] predicted influenza using an attention based LSTM. One of the observations they made was that the sequence length of their training data highly influenced the performance of their model. Applied to COVID-19, many work is done using LSTMs [23, 24, 25, 26]. Shahid et al. [27] perform a comparative study using a GRU, LSTM and Bi-directional LSTM. To train deep neural networks, one needs a lot of training instances. As for early epidemics the number of instances is limited, it may be challenging to create sufficiently detailed models. Typically, the architecture used has great influence on the performance of the model and should be carefully constructed. In our work this is not necessary as we use the underlying characteristics of the pandemic to automatically create our models.

2.4 Automated machine learning

The creation of regression pipelines encompasses many steps; data pre-processing, feature pre-processing, hyperparameter optimisation and algorithm selection. Sequential Model Based Optimisation (SMBO) is a black box optimisation framework that has been used for the purpose of hyperparameter optimization. Hutter et al. [28] used (SMBO) to automatically optimise hyperparameters of machine learning algorithms. SMBO stood as a basis for sequential model based optimisation and algorithm configuration (SMAC) [29]. This is a general purpose algorithm configurator, which made it possible to both select algorithms and tune hyperparameters. Auto-WEKA [30] is an AutoML framework around the WEKA software package using SMAC for its configuration. This framework fully automated the creation and tuning of classification and regression pipelines. Auto-sklearn [7] is an AutoML framework by Fuerer et al. around the popular Python package scikit-learn [31]. This framework includes meta-learning as a warm start for the configuration search and creates ensembles of pipelines. In more recent updates, they extended their framework with multi-output regression. This option makes it suitable for forecasting with a range of multiple days. TPOT [32] is a tree-based pipeline optimization tool for AutoML. Similar to auto-sklearn, it is build upon scikit-learn . Instead of using SMBO they use another approach for hyperparameter optimisation which is genetic programming. To validate pipelines internally, cross validation is used. This is also default in auto-sklearn.

However, in auto-sklearn other validation schemes, like holdout validation, are also possible. It is also possible to automatically construct deep neural networks. Frameworks that support this are Auto-Keras [34] and Auto-PyTorch [35], build Python packages. These frameworks find solutions to neural architecture search (NAS), where they aim to find the optimal neural network, minimising a loss function. Han et al. [36] used TPOT and H2O to forecast COVID-19 mortality data from Ceará. In their study, they found that TPOT outperforms regression models not automatically tuned, achieving an higher R^2 score. Marques et al. [37] compared H2O to an LSTM using the countries Brazil, China, the United States of America, Italy and Singapore and found that H2O outperformed the LSTM in terms of MAE, MSE and R^2.

In this work, we adapt auto-sklearn to the task of COVID-19 forecasting. As data is limited when forecasting the pandemic, using autoML systems generating deep neural networks is unfeasible. TPOT and auto-sklearn are comparable to each other, but because TPOT relies on cross validation to validate its pipelines, this is less suitable for time series forecasting. The cross validation scheme splits the data in k folds, training the models on $k - 1$ and evaluating on the one that was left out. As this happens iteratively changing which fold to evaluate on, the models are trained on future data to predict data in the past. Auto-sklearn supports holdout sets as validation scheme, ensuring we can train our models without relying on future data.

Chapter 3

Problem statement

We view the forecasting of COVID-19 a time series forecasting task. A time series holds discrete observations indexed over time. In other time series the sample rate may vary, but in our case this is constant, due to the availability of daily case and death data. Considering a time series containing COVID-19 incidence numbers of length n as $\mathbf{x} = [x_1, ..., x_n]$ with $\mathbf{x} \in \mathbb{R}^n$, a time series segmentation window of size w, a time step t and a forecasting horizon of size h, we want to use a segment of historical observations $[x_{t-w}, ..., x_t]$ from the time series up to observation x_t to forecast future data points $[x_{t+1}, ..., x_{t+h}]$. For the task of COVID-19 forecasting the time series we consider are the mortality rate of a country, where x_t denote the number of new deaths at time step t. When we consider using mobility time series $\mathbf{m} = [m_1, ..., m_n]$ alongside incidence data x , we extend the notation to use $[x_{t-w}, m_{t-w} ..., x_t, m_t]$ for the forecasting of $[x_{t+1}, ..., x_{t+h}]$. In our approach $\mathbf{m}$ holds the percentual increase of mobility for a country in a given form, such as the increase of time spent driving, or the increase of time spent visiting recreational areas. This format is dictated by the mobility data, provided by Apple and Google, that we study in this work.

To make comparisons between different countries, areas or cities possible, we normalize the incidence data by the size of its population N.

We formulate the configuration and selection of models as a Combined Algorithm Selection and Hyperparameter (CASH) Optimisation problem [30]. Given a set of algorithms $\mathcal{A} = A^{(1)}, ..., A^{(k)}$ with hyperparameter spaces $\Lambda^{(1)}, ..., \Lambda^{(k)}$ we search the optimal algorithm with optimal hyperparameter settings $A^*_{\lambda^*}$ following Eq. 3.1.

$$A^*_{\lambda^*} \in \underset{A^{(j)} \in \mathcal{A}, \lambda \in \Lambda^{(j)}}{\operatorname{argmin}} \frac{1}{k} \cdot \sum_{i=1}^{k} \mathcal{L}(A^{(j)}_{\lambda}, \mathcal{D}^{(i)}_{\text{train}}, \mathcal{D}^{(i)}_{\text{valid}}) \tag{3.1}$$

Here $\mathcal{L}$ is the loss generated by algorithm A when trained using set $\mathcal{D}_{\text{train}}$ and validated using set $\mathcal{D}_{\text{valid}}$. This loss is the mean squared error between the forecast made by algorithm A with hyperparameter settings λ and the true observations in the validation set, unseen by algorithm A. As we are optimising full pipelines, so optimising A means we are optimising the combination of pre-processors P, features F and regressors R, or $A = \{P, F, M\}$. Part of this process is optimising the input window size w, which is a newly added feature pre-processing step.

Chapter 4

Proposed methods

4.1 Data sets

The data used for our predictions comes from three sources: mortality data from the European Centre for Disease Prevention and Control and mobility data from Apple and Google. We show meta-data of these sources in Table 4.1 and describe the sources in the subsequent subsections.

Table 4.1: Meta-data of data sources. The end dates marked with an asterisk (*) are not actual end dates, as these data sets are at the date of writing still updated regularly.

Data source	Category	Countries	Start date	End date
ECDC 1 [38]	Mortality	214	2019-12-31	2020-12-14
ECDC 2 [39]	Mortality	30	2021-02-28	2021-07-10*
Apple [40]	Mobility	63	2020-01-13	2021-07-10*
Google [41]	Mobility	135	2020-02-15	2021-07-10*
Early / Intermediate	Combined	58	2020-02-15	2020-12-14
Late	Combined	26	2020-03-01	2021-07-10

Mortality data

The mortality data is collected by the European Centre for Disease Prevention and Control [38, 39]. The data is split into two sets, with the main difference being the period over which time-series are collected and the number of countries. Both data sets hold the daily number of new cases and the daily number of new deaths. Additionally, they denote in which continent each country lies and provide the country population size of the previous year. For the first data set this is the population size of 2019 and for the second data set this is the population size of 2020.

The EDCD 1 data set has data from the December 31st, 2019 until December 14th, 2020. At the start of the data set not all countries appear, as COVID-19 was not first encountered in all countries at the same time. The data is provided for 214 countries from all around the world. The ECDC 2 data set contains more recent data starting on the first of March 2021 and is still daily updated. The data in this set is collected for 28 countries in the European Union.

Both data sets are maintained and adjusted by ECDC when numbers are deemed inaccurate, due to delays in reporting. We use the daily new deaths as part of our input and as truth value to evaluate our estimations. To make sure the date is comparable between countries we normalise the daily new deaths to depict the number of daily new deaths per 1,000,000 people within the population.

Mobility data

We use two different mobility datasets, published by Apple and Google. Both these data sets represent movement of a population as a percentual increase or decrease as compared with a baseline established at a time step earlier than the start of the data set.

Apple. The Apple Mobility Trend Reports [40] contain the percentual increase or decrease of the use of modes of transportation as compared with a baseline volume on January 13th, 2020. The modes of transportation they specify are walking, driving and use of transit. This latter mode is not available for all countries, however, so in our features we only use the increase or decrease in the use of walking and driving as means of transportation. The data set includes data starting from January 13th, 2020 and is still regularly updated with new values. It holds data for 63 countries from all parts of the world. Many African countries are, however, missing.

Google. The Google Community Mobility Reports [41] contain the percentual increase or decrease of place visits as compared with a baseline period from January 3th to February 6th, 2020. The places are categorised in the following six categories: retail and recreation, grocery and pharmacy, parks, transit stations, workplaces and finally residential. The data set starts at February 15th, 2020 and is still regularly updated with new values. It holds data for 135 countries from all parts of the world.

Combined data

We merged the mortality data and the mobility data into two combined data sets. The first combined data set captures the first year of the pandemic and is studied in the *early forecasting* experiment and the *intermediate forecasting* experiment in Chapter 5. We used the intersection of dates and countries of the first ECDC data set and both mobility data sets. There were some missing values, which we imputed by taking the average of the value 7 days before the missing data point and the value 7 days after the missing data point. This way the imputed value fits well between the previous and next week and daily trends are preserved. For the country of Serbia the number of missing values exceeded 10%, which is why we omitted it from the data set. The resulting combined data set contains data from February 15th, 2020 until December 14th, 2020. It holds the following 58 countries, which, for convenience, we call the *global* collection of countries. Argentina, Australia, Austria, Belgium, Brazil, Bulgaria, Cambodia, Canada, Chile, Colombia, Croatia, Czechia, Denmark, Egypt, Estonia, Finland, France, Germany, Greece, Hungary, India, Indonesia, Ireland, Israel, Italy, Japan, Latvia, Lithuania, Luxembourg, Malaysia, Mexico, Morocco, Netherlands, New Zealand, Norway, Philippines, Poland, Portugal, Romania, Russia, Saudi Arabia, Singapore, Slovakia, Slovenia, South Africa, South Korea, Spain, Sweden, Switzerland, Taiwan, Thailand, Turkey, Ukraine, United Arab Emirates, United Kingdom, United States, Uruguay and Vietnam.

The second combined data set is an extension of the first combined data set, using the second ECDC mortality data set. This second set allows us to forecast later in the pandemic and is studied in the *late forecasting* experiment in Chapter 5. For this data set, we used the intersection of dates and countries of the second ECDC mortality data set and both mobility data sets. As there were countries for which some early dates were not recorded, this combined data set starts at March the 1st, 2021 and ends July the 10th, 2021. As the second ECDC data set only contains countries within the European Union, this set ends up with 26 countries. These countries are the following, which, for convenience we call the *European* collection of countries. Austria, Belgium, Bulgaria, Croatia, Czechia, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Ireland, Italy, Latvia, Lithuania, Luxembourg, Netherlands, Norway, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, and Sweden.

4.2 Methods

As the amount of possible pre-processors, feature pre-processors and regression models with each their corresponding hyperparameter settings is huge, we utilise auto-sklearn [7], an automation framework for configuring pipelines. Auto-sklearn is a wrapper around the popular Python module Scikit-learn [31]. Scikit-learn is used for regression and classification problems, providing ways to pre-process data, select features, fit models and evaluate the results. A typical usage of scikit-learn includes creating a pipeline linking these steps together. To create a pipeline there are many choices to be made, such as the choice of pre-processing steps, which models are used and what are the best hyperparameters for both the models and the pre-processors. Different choices may vastly influence the predictive performance of the pipeline, which is why we can benefit from making these choices automatically. Auto-sklearn automates the process of creating good pipelines. Within auto-sklearn, pipelines consisting of data pre-processors, feature pre-processors and machine learning models are evaluated on a validation set. It is built upon Sequential Model-based optimization for general Algorithm Configuration [29] (SMAC). SMAC constructs a model capable of predicting the performance of an algorithm on the corresponding configuration space. This model selects a list of promising configurations based on their expected improvement over the incumbent, the best seen configuration. A local search is performed near these promising configurations to find configurations with even higher expected improvement. In each iteration the incumbent is updated to store the best found configuration. The process of constructing an optimal pipeline can be warm-started by means of a meta-learning module. Before the search for good pipelines starts the input data set is compared with 140 data sets from the OpenML [42] repository, speeding up the search. Strong pipelines are saved in a resulting ensemble, which may subsequently be used for making forecasts.

In the following sections we explain how we implement our framework. Vanilla auto-sklearn does not support the time series of varying size as input, which is why we implement a feature pre-processor that limits the size of input sequences. We discuss the recent addition of multi-output regression that auto-sklearn included in their framework. This enables us to make predictions with longer forecasting horizons. This setting has no meta-learning available, which is why we create a method that makes repeated single-output predictions to achieve the same horizon of forecasts. Furthermore we detail our training strategy. We discuss adaptation mechanisms used to handle concept drift. At the end of the chapter we give an overview of the configuration search space.

Variable window size. To predict the value of $[x_{t+1}, ..., x_{t+h}]$we train the models with sequences of the time series in the form of $[x_{t-w}, ..., x_t]$. In vanilla auto-sklearn this window size w is unchangeable. The number of features provided to the model is static, as each instance should normally be represented by a combination of all features. This would mean that when we use lags of the time series as features, the number of lags is predetermined as well. This may however not be desirable. When making predictions with different regressors, not all parts of the time series may be relevant and depending of the configuration it can be good to use a longer or shorter time input sequence. This is why we implement the *variable window size* feature pre-processor as proposed by Wang et al. [43]. This pre-processor has hyperparameter w that is optimised within auto-sklearn. The pre-processor takes the input sequence with predetermined static length and cuts off the first values, resulting in an input sequence in the form of $[x_{t-w}, ..., x_t]$. In their work they have experimented on a large set of time series tasks and showed that the variable window size had major impact on the accuracy of the framework. They have tested the pre-processor for generally large (up to 200 time steps) windows. As larger windows limit the number of data instances we can use, we limit our windows to a maximum of 30 days.

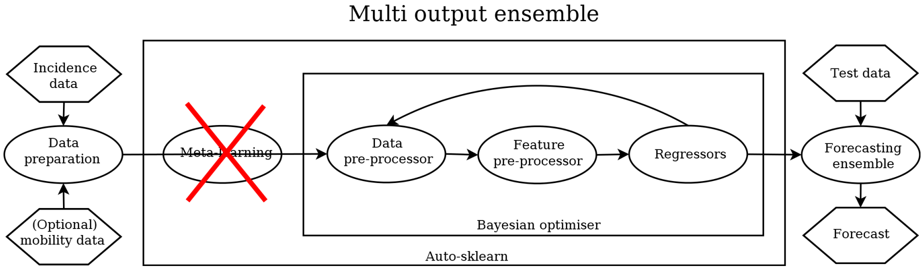


Figure 4.1: Schematic overview of the multi-output ensemble, as implemented by auto-sklearn. This ensemble creates multiple predictions at once, but has no access to meta-learning. Within the framework pipelines are constructed to form a forecasting ensemble. By feeding this ensemble test data predictions can be made.

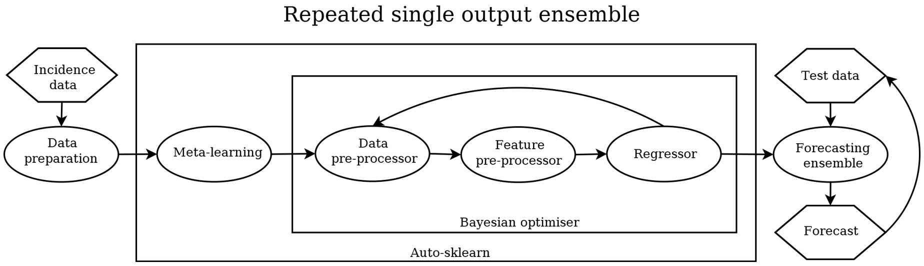


Figure 4.2: Schematic overview of the repeated single-output ensemble. This ensemble creates one prediction at each time step. To create predictions for a longer time period, for each new time step the predictions of previous steps are used. This method has access to meta-learning, but can not use additional data sources as input.

Forecast horizon. Since version 0.8 auto-sklearn supports multi-output regression, such that forecasts with forecast horizon $h > 1$ may be performed without the need for training multiple models. We use this way of multi step-ahead forecasting in our method we call *multi-output*. We show a schematic overview of this method in Figure 4.1. To make a multi-output prediction, separate regressors are fitted for each value of output. This means that each model consists of h regressors. As this output format was implemented much later than others, there is no meta-learning available for multi-output regression. This is why we create a new way to make predictions with an horizon $h > 1$. The *repeated single-output* forecasting scheme is a model trained for single-output regression, but once it starts forecasting its output is appended to the input sequence. In a sense, the model rolls over its own predictions. For instance, when we want to predict the value of x_{t+2} we use the sequence $[x_{t-w+1}, ..., x_t, x'_{t+1}]$ as input. In this sequence x'_{t+1} denotes the prediction of value x_{t+1}. Note that when we append values to the input sequence, we remove values at the start of the sequence. Each model uses one regressor. We show a schematic overview of this method in Figure 4.2. The advantage of this method over the multi-output regression method is that it does gain the benefits of meta-learning. However, as it is not trained specifically for forecasting multiple days in future, predictions further away may suffer from errors made earlier. Another disadvantage is that this method can not use external changing variables as input, as only one time series is predicted.

The distribution of the repeated single-output ensemble differs from the multi-output regression ensembles. The repeated single-output ensemble generates one prediction for each day in the horizon. The multi-output ensembles generate multiple predictions for each day in the horizon. This is because each step, the ensembles predict for a number of days equal to h. As the forecasting window shifts h steps over the test set, the first day in the forecasting window yields h predictions, the second $h - 1$, and so forth until the last day that receives 1 prediction. In the case of multi-output regression we take the mean prediction for each day as final forecast.

Training strategy. As tuning many parameters requires lots of data instances to prevent overfitting, we put together the time series of all countries in the data set for training, as opposed to training separate models for separate countries. This way we create a joint model capable of forecasting for many countries. We normalised the incidence data by the size of the population of each country. The mobility data depicts percentual changes in mobility, which does not require further normalisation to make comparisons between countries possible. To make sure it handles individual countries well, we pass the country name as categorical feature to each instance. For testing, we separate time series per country again. This way we can compare the forecasting quality between countries. The default options of auto-sklearn shuffle the data while training. However, we do not shuffle it to ensure the temporal integrity of the data set. Another default setting is the use of cross validation as resampling strategy. Applied to time series this would mean that for most folds future data is used to predict previous values. To negate this problem we use a holdout set for validation. This set is situated at the end of the training set, just previous to the start of the test set, to be sure that the ensembles can't learn future information. As optimisation metric we use the mean squared error. This ensures the regressor line tries to fit the set of points as close as possible. To make sure our ensembles are fully trained on the data, we refit the ensembles on the full train and validation set after validation is finished. This means that while the pipeline stays the same, the weights of the regressors are updated with both the train and validation set. This way we make sure there is no gap in knowledge just before the forecasting starts.

Drift adaptation mechanisms. Since the infectiousness of COVID-19 may change over time, for instance due to mutations or vaccinations, the underlying concept generating the data may change. When the concepts changes vastly, this is detrimental to the performance of machine learning algorithms. This change is known as concept drift. In Equation 4.1 we show a formal definition of concept drift between two time steps t_0 and t_1 [44].

$$\exists X : p_{t_0}(X, y) \neq p_{t_1}(X, y) \tag{4.1}$$

In this definition p_{t_0} is the joint distribution between the set of input sequences X and target values y at time t_0. Currently, auto-sklearn has no drift detection mechanism. However, as we use two data sets, separated in time, with a different population size, we can be sure some concept drift exists in our time series. After all, the population size is used to scale the death incidence data. Celik and Vanschoren [45] created several concept drift adaptation mechanisms for automated machine learning frameworks. We implement three methods based on their work that do not use drift detection to cope with concept drift. For each of the methods we first construct ensembles using the old data set. The *full refit* method then uses the full combination of both data sets to refit the ensembles. The *partial refit* method instead uses only the new data set to refit the ensembles. The *retrain* method discards the ensembles and constructs new ones with the new data set. These methods can be seen as a kind of forget mechanism, with varying degrees of aptness to forget. The full refit method places most emphasis on older data in comparison with the others. The partial refit method still uses the older data in the form of ensembles, but the weights are only updated with new data. The retrain method forgets the old data altogether and only uses new data for its predictions. Depending on the magnitude of the concept drift there can be merit for each method.

Configuration search space. Regression pipelines created in auto-sklearn consist of data pre-processors, feature pre-processors and regressors. Each of these components have their own hyperparameter search space. Each pipeline is a configuration of two categorical data pre-processors, two numerical data pre-processors, one feature pre-processor and one regressor. Each of these components have their own hyperparameters that need to be tuned. We show the components available in our framework with their corresponding number of hyperparameters in Table 4.2. These are all the default auto-sklearn components for regression, with the addition of our newly included variable window.

Table 4.2: Configuration search space.

Name	Category	Number of hyperparameters
Categorical encoding	Data pre-processor	1
Category coalescence	Data pre-processor	2
Imputation strategy	Data pre-processor	1
Rescaling	Data pre-processor	5
Feature pre-processor	Selector	1
Window size	Feature pre-processor	1
Extra trees for regression	Feature pre-processor	9
Fast ica	Feature pre-processor	4
Feature agglomeration	Feature pre-processor	4
Kernel pca	Feature pre-processor	5
Kitchen sinks	Feature pre-processor	2
Nystroem sampler	Feature pre-processor	5
Pca	Feature pre-processor	2
Polynomial	Feature pre-processor	3
Random trees embedding	Feature pre-processor	7
Regressor	Selector	1
Decision tree	Regressor	8
Extra trees	Regressor	9
Gaussian process	Regressor	3
K nearest neighbors	Regressor	3
Random forest	Regressor	9

Chapter 5

Experiments

The challenges of forecasting epidemics change over time. When an epidemic is still novel, there is limited data available. Especially when using complex methods, this limitation may hamper the forecasting quality. It is in this stage highly beneficial to be able to create accurate estimations early, as the number of infections follow an exponential growth. For instance, in the Chinese province Hubei, where COVID-19 started, the doubling time was estimated at 2.5 days [46]. Later on, when the epidemic is in full sway, more data is available, but the differences between countries may also become larger. As the amount of data enables our framework to train better, it is interesting to see to what extent the performance improves over early forecasting. As the disease may mutate over time and as the population may gain (partial) immunity due to vaccination, there may arise a concept drift [47]. As we learn from observations generated by the process of the epidemic, the observations change when the generative process changes. It is interesting to see how to deal with such a change in concept.To shed light on these challenges, we propose the following questions:

- **Q1.** How accurate is our modified AutoML system for the early forecasting of COVID-19?
- **Q2.** Given more training data, can we improve the accuracy of COVID-19 forecasts?
- **Q3.** How accurate is our modified AutoML system when it is adapted to concept drift?

To answer these questions, we design three scenarios, each answering one of these questions. The scenarios are distinguishable by time. In the *early* scenario we focus on forecasting when limited data is available. For the *intermediate* scenario we use data for the first year of the pandemic. In the *late* scenario we add an additional mortality data set, situated three months after the first one. In the next section, we detail the different scenarios. Later in this chapter we discuss our experimental setup.

5.1 Scenarios

Early forecasting. The early forecasting scenario aims to test the performance of our methods when the epidemic is still novel, thus answering question **Q1**. In this scenario we use a train set with data from February the 15th, 2020 until April the 21st, 2020, and evaluate the forecast on a test set of the next 14 days, until May the 5th, 2020. We evaluate our methods using the 58 countries in the global collection of countries. However, as we wish to compare with Bayesian inference method as a baseline which is discussed in the next section, we limit the countries to the following 11 countries when comparing with the Bayesian inference model: Austria, Belgium, Denmark, France, Germany, Italy, Norway, Spain, Sweden, Switzerland, United Kingdom.

Intermediate forecasting. The intermediate forecasting scenario aims to test the performance of our methods with less data scarcity. By using more date than the previous scenario we aim to answer **Q2**. In this scenario we use a train set with data from February the 15th, 2020 until November the 14th, 2020. We evaluate on a test set of the next 30 days, until December the 15th, 2020. Again, we use the 58 countries in the global collection of countries.

Late forecasting. The late forecasting scenario aims to test our performance when coping with concept drift, thus answering question **Q3**. To do this, we use an additional mortality data set, situated about three months after the first. Along with potential implicit underlying drift, there is explicit drift in the fact that this set provides a new population size. We use this population size in our methods to scale the mortality numbers. This means that when scaled values are the same for both data sets, they actually denotes a growth in the number of daily deaths when the population has grown. In this scenario we use a train set with data from February the 15th, 2020 until December the 14th, 2020. After ensembles are created, we use the data set from March the 1st, 2021 until June the 10th, 2021 as additional train set for the drift adaptation mechanism and evaluate on a test set of the next 30 days, until July the 10th, 2021. The data between December 15th 2020 and February 28 2021 were not available, resulting in a gap between data sets. We limit the countries in this scenario to the 26 European collection of countries, as the others are not available in the new data set.

5.2 Experimental setup

Our framework is built on version 0.12.1 of auto-sklearn. All our ensembles are run for 3 hours . For each iteration we limit the runtime to a maximum of 10% of the total runtime, which comes down to 18 minutes. The majority of iterations, however, finish much faster. This amount of time ensures hundreds of models are compared to create the resulting ensembles. We run auto-sklearn in parallel on 8 cores, of an Intel(R) Xeon(R) CPU of 2.1 GHz with 10 GB of RAM. As mentioned before in Chapter 4 we use a holdout set as validation strategy, and make sure not to shuffle the data. As internal performance metric we use the mean squared error. We use the full default regressor and pre-processor search space, and extend the feature pre-processor search space by adding the variable window size pre-processor. We limit the window pre-processor to a minimum of three days and a maximum of 30 days.

Bootstrapping. As the Bayesian optimisation used by auto-sklearn is stochastic, one run of the framework may optimise towards a locally optimal configuration, thus not yielding the actual optimal configuration. We perform bootstrapping to gain confidence in our predictions. For each estimator we make, we run our framework 25 times. Repeating 1,000 times, we sample with replacement five ensembles from the 25 runs, of which we select the one with the lowest validation error. These 1,000 selected models form our bootstrap distribution used to analyse results. For each day within the forecasting horizon, we report on the mean forecast and the 95% confidence interval. We use our bootstrapping approach not only for our methods, but also for the deep learning baselines.

Performance metrics. To evaluate our methods we use the mean squared error as defined in Equation 5.1. Here Y denotes true target values of our time series and $\hat{Y}$ the prediction of the model. As our ensemble creates multiple predictions for each day, the daily average is used for $\hat{Y}$. When analysing how the methods train, we compute the mean squared error of forecasts scaled by the population size of each country. This way, the error resembles the scaled values as they are used when training the models. To gain more insight in the quality of our prediction we perform error decomposition, splitting the MSE in a bias term and a variance term as shown in Equations 5.2 and 5.3. The bias is the difference between the expected value of the prediction and the truth label. The variance is the average of the squared deviation of the mean. Ideally, both the bias and the variance are low. In this case predictions are accurate. If the bias is high and the variance is low, it means that the estimator is quite confident, but misses the target consistently. In this case the ensemble has learned false information. If the bias is low and the variance high, the estimate makes predictions around the truth label, but with a wide range. In this case estimators may benefit from more training data, to narrow down the prediction interval. Analysing the error decomposition gives us the ability to decide whether or not the methods are capable to learn useful structures, allowing for improvement by adding more data or changing the method completely. Finally, we present the performance of all methods in form of root mean squared error, shown in Equation 5.4, over unscaled forecasts. The unscaled forecast is the prediction of the number of actual deaths in a country. We argue that these forecasts may be more useful for policy makers interested in individual countries. By computing the root of the mean squared error we translate the size of the error back to the unit.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{5.1}$$

$$Bias^2 = (Y - \overline{\hat{Y}})^2 \tag{5.2}$$

$$Variance = \frac{1}{m}\sum_{j=1}^{m}(\hat{Y}_j - \overline{\hat{Y}_j})^2 \tag{5.3}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2} \tag{5.4}$$

5.3 Methods and baselines

In this section, we discuss our ensembles and the baselines we compare them with. Our proposed methods are the *multi-output* ensemble and the *repeated single-output* ensemble. For the repeated single-output ensemble we use only deaths data. For the multi-output ensemble we try different combinations of deaths and mobility data, as well as additions in the form of spatial weights and train data from countries not used in the test set. In the late forecasting scenario we compare three drift adaptation techniques – fully refitting on all data, partially refitting on only new data and retraining on only new data – for both our methods. We compare our methods with the *persistence*, *Bayesian inference*, and *ARIMA wavelet*, *GRU*, *LSTM* and *Bi-LSTM* baselines. Details of these methods are found below.

Our methods. The *repeated single-output* ensemble uses single-output regression, enabling meta-learning for warm starting the algorithm configuration search. As this ensemble uses its own predictions as input for future predictions, it is not possible to use external time series data like mobility data. For each of the scenarios, this methods uses only death incidence data as features. The *multi-output* ensemble does not have this limitation, which is why we vary the data sets used in combination with this method. For each of the scenarios we use this method with and without mobility data. This data comes in the form of percentual increase of the use of modes of transportation, like walking, or the percentual increase of visits to place categories, like grocery stores. When these values are negative we speak of a percentual decrease. In our intermediate and late forecasting scenarios we split the mobility data set to see if Apple, Google or the combined mobility data set is most informative.

As the spread of diseases does not happen only over time, but also over space, we experiment with adding spatial lags to the data input sequences. Countries that are closer to each other may have patterns that are more similar to each other than to countries farther away. Spatial weights allow us to take spatial correlations into account. To this end, we use the PySAL module, a Python library for spatial analysis [48]. Using a shapefile, we define neighbors as countries that share a border. Using the Queens weighting scheme, we do not apply limitations on the amount of neighbors that are considered. The spatial weights have a smoothing effect over neighbors. When multiplied with the daily new deaths, this will be smoothed across countries. In the combined data set used with the other ensembles, we have only 58 countries. To be able to create a spatial weights matrix where all countries have neighbours we need more countries. In the original ECDC data set there are however more countries available. By combining the ECDC data set with the shapefile we are able to implement spatial weights for 212 countries worldwide. The spatial weights are only applicable in the intermediate forecasting scenario, as only in this scenario there are enough countries available in the data set.

In an effort to reduce the variance of the error, we compare the performance of the multi-output ensemble in the intermediate scenario using death incidence data of the 58 countries in the global set with the multi-output ensemble using the death incidence data of the 212 countries worldwide used for creating spatial weights. This way we can feed the ensemble more observations, bringing the variance of the error down. Similar to the spatial weights, this addition of countries is only applicable to the intermediate forecasting scenario, as only in this scenario there are enough countries available in the data set.

In the late forecasting scenario, we experiment with drift adaptation techniques. Both our methods use the techniques *full refit*, where the full combined data set is used for refitting the old ensembles, *partial refit*, where only the new part of the data set is used for refitting the old ensembles, and *retrain*, where only the new part of the data set is used for training new ensembles.

Baselines. The *persistence* baseline is a naive baseline that forecasts the last seen observation from the train data for all days in the forecasting window. When forecasting the window $[x_{t+1}, ..., x_{t+h}]$ each predicted value will have x_t, disregarding all previous values x_i with $i < t$.

We compare against the *ARIMA wavelet* model [19] proposed by Chakraborty and Gosh, as it showed improvement on the widely used ARIMA models. The ARIMA wavelet model is the combination of an ARIMA model and a wavelet based forecasting model. It fits an ARIMA model on the mortality data and then models the residuals via the wavelet model. We use the model as implemented by Chakraborty and Ghosh, but adjust the number of forecasting days to align with the scenario. The parameters of the ARIMA model controlling the order of autoregression, the order of differencing and the moving average are automatically configured using a grid search and the Akaike Information Criterion. As the drift adaptation techniques are not applicable to this model, we use only the new data in the late forecasting scenario as train set.

The *Bayesian inference model* [14] proposed by Flaxman et al. was highly influential in estimating the effects of government measures in containing the disease. It is an hierarchical model, including a model for the daily number of deaths and one for the daily number of infections. The forecasted number of deaths depends on the number of infections of the previous day. The forecasted number of infections depends on the number of infections of the previous day and the reproduction number, which in turn is a function of the effect of each intervention. Through Bayesian inference the effects of the interventions are estimated, which in turn is used to construct the prediction models. By chaining predictions multiple days of forecasting can be achieved. We use the model as implemented by Flaxman et al. but use an updated mortality data set, as the data set they used was a previous version published by the ECDC. It is important to note that for this baseline only 11 countries are available. This is due to their dependency on detailed data like containment strategies and the infection fatality for each country. These countries are Austria, Belgium, Denmark, France, Germany, Italy, Norway, Spain, Sweden, Switzerland, United Kingdom. Additionally, for this model the necessary data was only available until May the 5th, 2020, making the model applicable to only the early forecasting scenario. The model provides a 95% confidence interval but does not give insights into its error decomposition.

To compare our framework with recurrent neural networks, we reproduce the *GRU*, *LSTM* and *Bi-LSTM* as studied by Shahid et al. [27]. The different algorithms differ from each other only by the type of neurons they consist of. The GRU neuron has two gates: the reset and update gates. The LSTM neuron has three gates: the input, output and forget gates. The Bi-LSTM has the same neurons as the LSTM, but approaches the time series in two ways, once in time order and once in reversed order. In our comparison, all three architectures share the same architectures, as chosen by [27] and shown in Table 5.1. We did however enlarge the batch size from 10 to the number of countries in the scenario: 58 for the early and intermediate forecasting scenarios and 26 for the late forecasting scenario. This allows the models to train for each country simultaneously without them being able to see future time steps. Additionally we increased the number of time steps used as input to 30, to match the other ensembles and baselines in our comparison. To give these methods a fair comparison to our own, we experiment with adding mobility data features in all scenarios by concatenating mobility feature columns for each day in the input sequence, similar to our own methods, as well as the retrain drift adaptation technique in the late forecasting scenario.

Table 5.1: Hyperparameter settings for the GRU, LSTM and Bi-LSTM.

Hyperparameters	Values
No. of neurons	{16, 32, 64, 128}
Learning rate	0.001
Optimiser	Adam
Batch size	No. of countries: 58 or 26
Epochs	300
Time steps	30

Chapter 6

Results

In this chapter we present the results of our experiments. Each section presents one of the three scenarios. The first section shows the early forecasting scenario. We use a limited amount of data and answer **Q1**. In the second section we show the intermediate forecasting scenario. Here we use a longer time period and answer **Q2**. The third scenario is the late forecasting scenario. It introduces concept drift and answers **Q3**. In each of the figures our methods are indicated with an M and all baselines with a B.

6.1 Early pandemic forecasting

Because we want to compare the predictive performance of our methods for many different countries, we create rankings of their performance based on MSE relative to each other over all countries. When one method is consistently better than the others over most countries, it gets assigned a lower average rank. The average ranks of all methods give insight on how well the methods perform compared to each other. When average ranks are close to each other, it is not immediately obvious if the methods are significantly different. Using the Nemenyi test [49] we can visualise the significant difference between average ranks. This test defines a critical distance between average ranks. Any methods within critical distance to another method is not significantly different. As a preliminary result we compare the deep learning baselines with each other, each including or excluding mobility features. We show this in Figure 6.1. The value indicated by the line next to a method name is the average rank. A method is better when it is more to the left of others. The horizontal lines linking methods together indicate that these methods fall within critical distance and are thus not significantly different. The LSTM and Bi-LSTM without mobility features are the left-most methods linked by a bar, indicating they are the best methods and on par with each other. For both the LSTM and Bi-LSTM it shows that the models without mobility features are better than their counterparts with mobility features. For the GRU the model with mobility features seems a bit better than the one without mobility features, but for this model the difference is not signficant. For the rest of the results of the early forecasting scenario we will consider the LSTM and Bi-LSTM without mobility features and the GRU with mobility features.

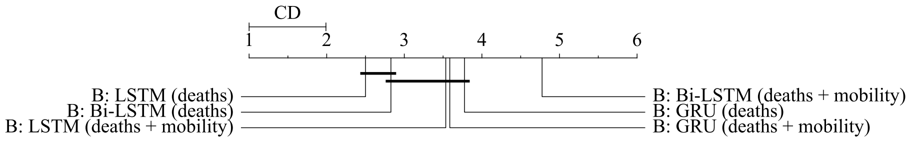
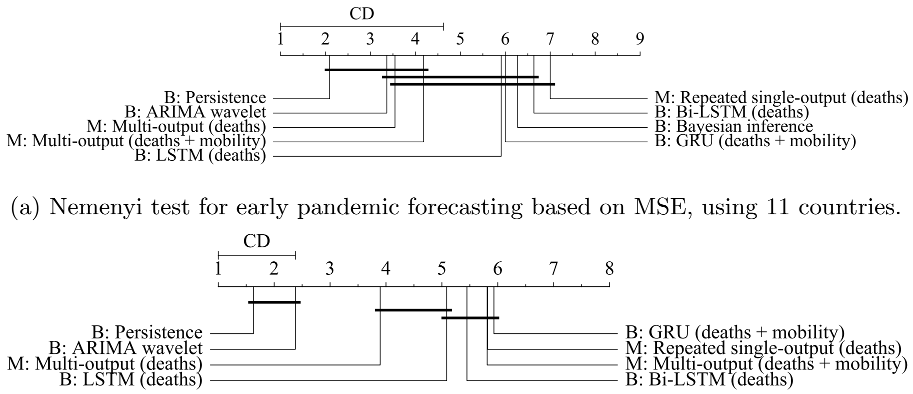


Figure 6.1: Nemenyi test for the deep learning baselines based on MSE in the early pandemic forecasting scenario, using 58 countries. Methods with lower average rank (left) are better than those with higher rank (right). Connected methods are not significantly different.



(a) Nemenyi test for early pandemic forecasting based on MSE, using 11 countries.

(b) Nemenyi test for early pandemic forecasting based on MSE, using 58 countries.

Figure 6.2: Nemenyi tests for early pandemic forecasting based on MSE. Methods with lower average rank (left) are better than those with higher rank (right). Connected methods are not significantly different.

We show the comparative performance of our methods and baselines in Figure 6.2a, which shows comparison using the 11 countries for which the Bayesian inference model is suitable., and Figure 6.2b, for which we use all 58 countries of the global collection.. From the figures we see that in both cases the persistence and ARIMA wavelet baselines perform best. In the comparison using 11 countries (Figure 6.2a) our multi-output ensembles are on par with the more simple persistence and ARIMA wavelet baselines. Our repeated single-output ensemble gets clearly outperformed by these baselines. The deep learning and Bayesian inference baselines are on par with our best models but are outperformed by the persistence baseline. If we look at the comparison with 58 countries (Figure 6.2b) the differences are more pronounced. Here our multi-output ensemble without mobility features is again our best method, but is outperformed by both the persistence and the ARIMA wavelet baselines. It is on par with the LSTM baselines but outperforms the other methods. Our other methods perform worst together with the GRU, LSTM and Bi-LSTM baselines. It is interesting to note that in this comparison our method with mobility features performs worse than in the comparison with 11 countries. A possible explanation for this is that in many countries the pandemic had not started yet, and restrictions of mobility would not yet be in place. For the reader interested in the model performance with unscaled data, we show the results based on RMSE using unscaled deaths in Table 8.1 in the Appendix.

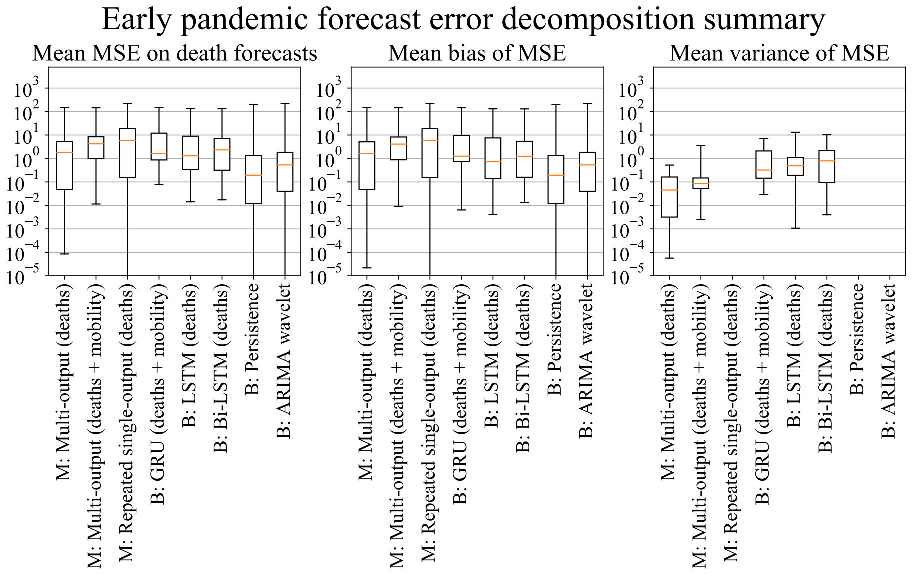


Figure 6.3: Early error decomposition summary. Each box shows the mean MSE and its decomposition in bias and variance on log scale over 58 countries.

Figure 6.3 shows a summary of the error decomposition of MSE into its bias and variance components. Each box represents the mean MSE, mean bias and mean variance of the 58 countries in the global collection. We do not show the Bayesian inference baseline in this figure, as this model reported on its mean predictions and 95% credible interval without showing the distribution of predictions. The figure shows that for each of the methods the bias on average is the main component of the MSE. The persistence and ARIMA wavelet methods produce exactly one prediction, which is why for these models the MSE equals the bias. The relatively high bias in combination with low variance may indicate some underfitting. This is the case for both the deep learning baselines and our own methods. Especially the repeated single-output ensemble shows no variance at all.

Overall, the most simple baselines – the persistence and ARIMA wavelet baselines – dominate the early forecasting scenario. There are two main reasons accountable for this. Firstly, because we use only a short forecasting horizon of 14 days, the persistence baseline is quite strong if the change in the truth values is low. Especially early in the pandemic this is the case in many countries, where the disease still had to pick up pace. To visualise this, we show the predictions for Austria in Figure 6.4, but many other countries – like Denmark, Egypt, Hungary or Sweden (Figures 8.1 to 8.4 shown in the Appendix) – show similar behaviour. In the figure we see that the truth label is relatively steady. Here the persistence and ARIMA wavelet baselines fit right through the average of the truth, resulting in low MSE. In countries where COVID-19 was not yet widely spread – for instance New Zealand, which is shown in Figure 6.5 (but again, similar countries exist and are shown in the Appendix, like Argentina, Chile, Malaysia and South Africa, Figures 8.5 to 8.8) – the baselines have a similar advantage. For these countries our mobility ensemble tends to forecast deaths that transcend the truth vastly. It may be that in this scenario mobility patterns are a bad indicator of incidence, as high mobility in countries with lots of contagion may have a different effect than high mobility in countries with little to no contagion.

A second reason why our ensembles fair badly against the baselines is that due to the frequency of data collection there is not much training data available. This may especially be disastrous for the multi-output ensemble with mobility features. This ensemble may have too many features to account for. Where the ensembles without mobility features have one feature per day, the mobility ensembles have nine. Combined with the lack of data in the early forecasting scenario these ensembles may not be trained well. The error bars of the Bayesian inference baseline get quite wide when time progresses, as consecutive predictions use previous predictions as base. For our multi-output ensembles as well as the deep learning baselines this is not the case. These are trained specifically to forecast multiple steps ahead. We expected that our repeated single-output ensemble would have error bars growing over time, similar to the Bayesian inference baseline. However, it has extremely slim error bars, which may indicate a case of strong underfitting.

This scenario shows when data is still sparsely available, our methods are outperformed by more simpler methods like the ARIMA wavelet model and the persistence baseline. From error decomposition we see signs of underfitting. This means that our methods do not capture the more complex behaviour of the data in this early stage of the pandemic. Of our methods, the multi-output ensemble with only death incidence data fairs best. This ensemble is on par with the LSTM baseline but is more accurate than the others.

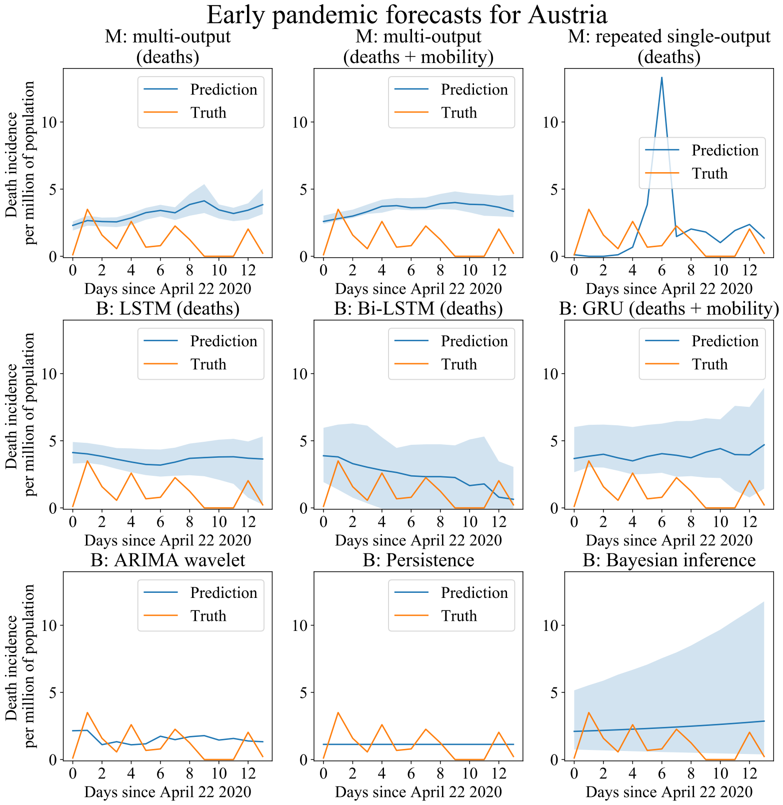


Figure 6.4: Early forecasting for Austria.

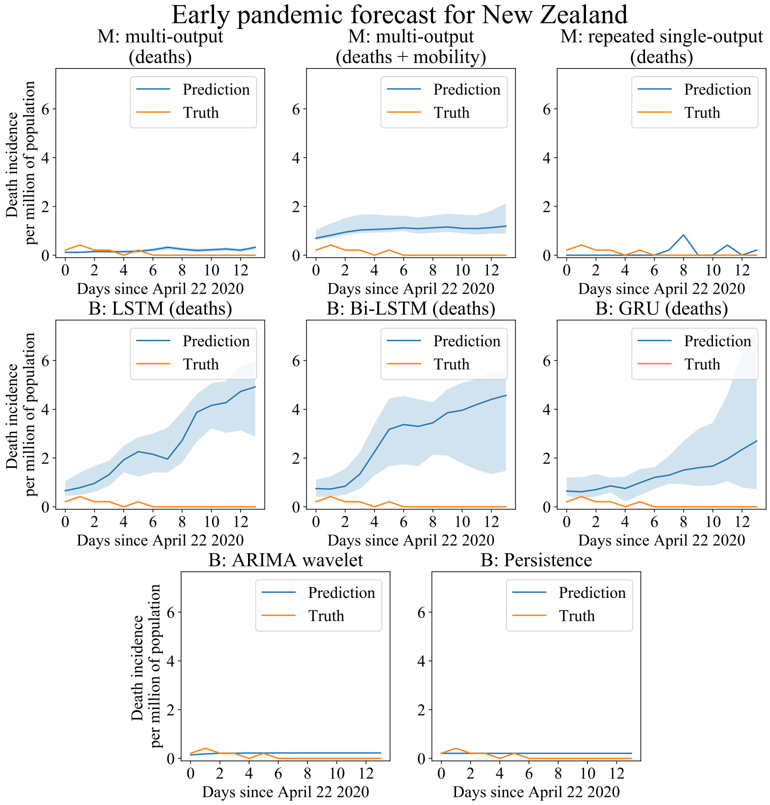
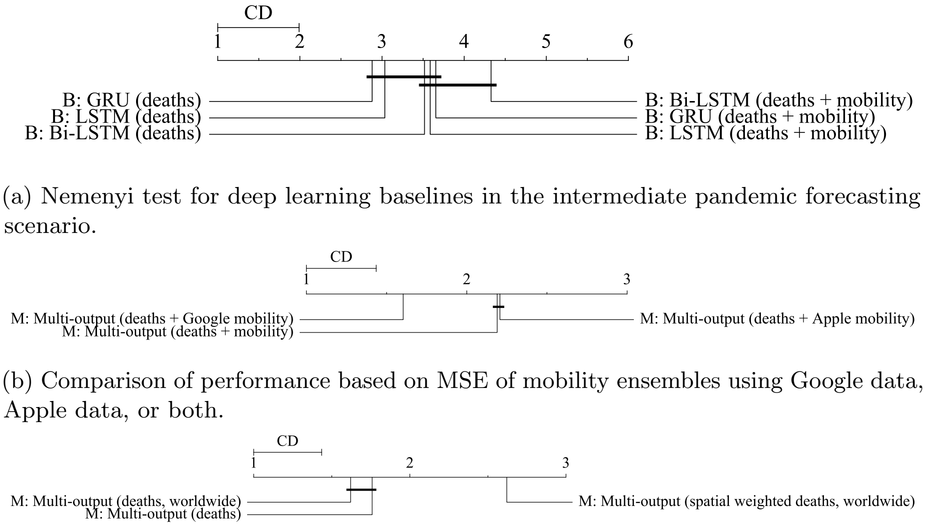


Figure 6.5: Early forecasting for New Zealand.

6.2 Intermediate pandemic forecasting

In this scenario COVID-19 has taken a grip on more countries. Most countries now have some number of death incidence and the variance of the true observations is on average larger than that of the early forecasting scenario. As also the time series have become longer it is interesting to see whether we can improve our predictions. As a preliminary study we look into using different combinations of our data sets. First we compare the deep learning baselines with and without mobility. Then, we compare the mobility data sets, split into Apple, Google and the combination of the two to see which set is most useful. After that, we experiment with spatial weights as features, and additional countries worldwide for training, to see if we can improve the multi-output ensemble without using mobility data. We use the best of these results in our subsequent experiment where we compare each ensemble with other baselines.



(a) Nemenyi test for deep learning baselines in the intermediate pandemic forecasting scenario.

(b) Comparison of performance based on MSE of mobility ensembles using Google data, Apple data, or both.

(c) Comparison of performance based on MSE of multi-output ensembles not using mobility data, extended with additional countries for training or spatial weights.

Figure 6.6: Nemenyi test for intermediate pandemic forecasting using different mobility data sets in our ensembles based on MSE, using 58 countries. The subfigures compare differences in performance when selecting certain mobility data sets or when using additional countries for training or spatial weights as features. Methods with lower average rank (left) are better than those with higher rank (right). Connected methods are not significantly different.

Figure 6.6a shows a Nemenyi plot for the deep learning baselines deployed for the intermediate forecasting scenario. The differences between the baselines are small, as most are connected to each other. The GRU, LSTM and Bi-LSTM baselines without mobility features seem to be better than their counterparts with mobility features, but for none of them this is on a significant level. For the subsequent comparisons we consider the baselines without mobility features.

It is interesting to know which mobility data set is the most useful aid for predicting COVID-19 death incidence. To this end, we split it into three different mobility ensembles: one using Apple mobility data, one using the Google mobility data and the original mobility ensemble using both. We show the relative performance of these mobility ensembles according to the Nemenyi in Figure 6.6b. From this figure we see that using Google mobility data increases performance as compared to using Apple data or both.

Because the spread of diseases is not only a process depending on time, but also on space, we use this scenario also to see how using spatial weights would affect the predictive performance. Additionally, we test if we can improve performance decreasing the variance by providing data from additional countries as train set. We still use the 58 countries from the global collection for testing. The use of these additional countries excludes the use of mobility features. From Figure 6.6c we see that using spatial weights does not yield better performance. Adding extra countries in the training phase seems slightly better, not on a significant level. In the subsequent experiment we use the multi-output ensembles with Google mobility features and the multi-output ensemble trained on additional countries worldwide.

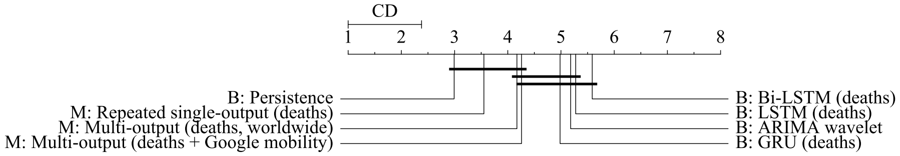


Figure 6.7: Nemenyi test for intermediate pandemic forecasting using the best configured ensembles based on MSE, using 58 countries. Methods with lower average rank (left) are better than those with higher rank (right). Connected methods are not significantly different.

We show our full comparison with the best ensembles from the preliminary results in Figure 6.7. In this scenario the persistence, our repeated single-output ensemble, our multi-output ensemble using Google mobility data and our multi-output ensemble trained on 212 countries worldwide are on par as the methods with the lowest error. The repeated single-output ensemble is better than all baselines other than the persistence baseline on a significant level. In general, there is no clear winner for this scenario, as average ranks are quite close to each other. For the reader interested in the model performance with unscaled data, we show the results based on RMSE using unscaled deaths in Table 8.2 in the Appendix.

We show the error decomposition of the intermediate forecast scenario in Figure 6.8. Similar to the decomposition of the first scenario is that the bias

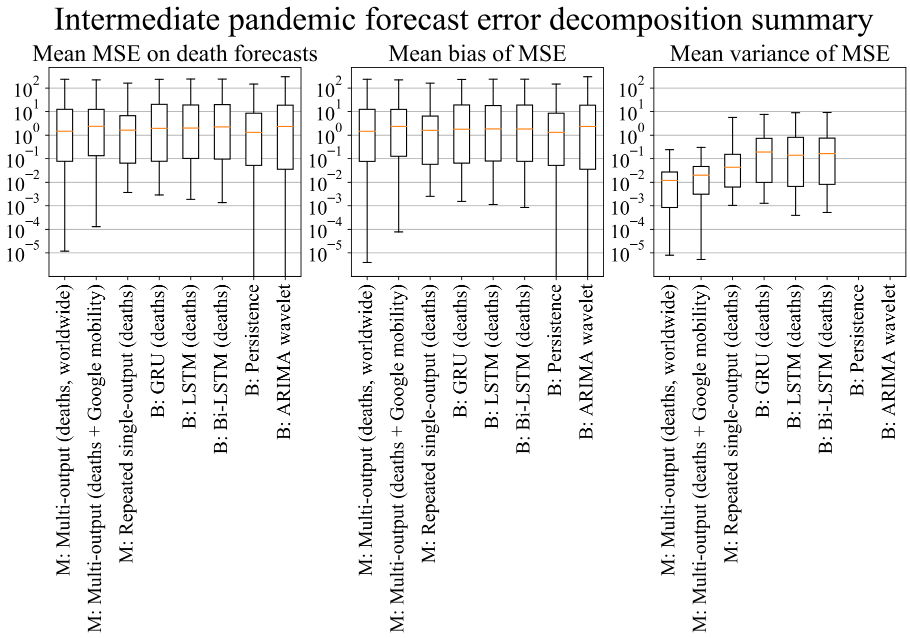


Figure 6.8: Intermediate error decomposition summary.

component of the methods is relatively larger than the variance component. A difference is that in this scenario the repeated single-output ensemble now has a variance component. The MSE of our methods and the deep learning baselines look roughly similar to the MSE in the previous scenario. However, as the variance of the true observations is larger in this scenario, this means that the models are somewhat better in capturing complex data. The persistence and ARIMA wavelet models suffer more for this increase in variance. While the persistence is still among the best estimators for this scenario, the MSE drops in comparison to the early forecasting scenario.

When we look at the true observations of the countries, we can distinguish a few different patterns. Some countries show clear weekly cycles in the data. Some countries have a consistent upward or downward trend. Others have an upward trend first and downward later. For some countries the weekly upward and downward trends and weekly cycles are mixed. Another group has little to no variation in new daily deaths, and some countries fall outside of all of these. To give an example of a country with weekly cycles, we show the country of Switzerland in Figure 6.9, as the cycles are quite consistent for this country. Other examples with cycles include Mexico, the Netherlands, Poland and are shown in the Appendix in Figures 8.9 to 8.11. In these countries the baselines do not forecast well. Our multi-output ensembles also tag behind, but our repeated single-output ensemble is able to capture weekly cycles quite well. To give an example of a country with clear trend, we show the forecast of Belgium in Figure

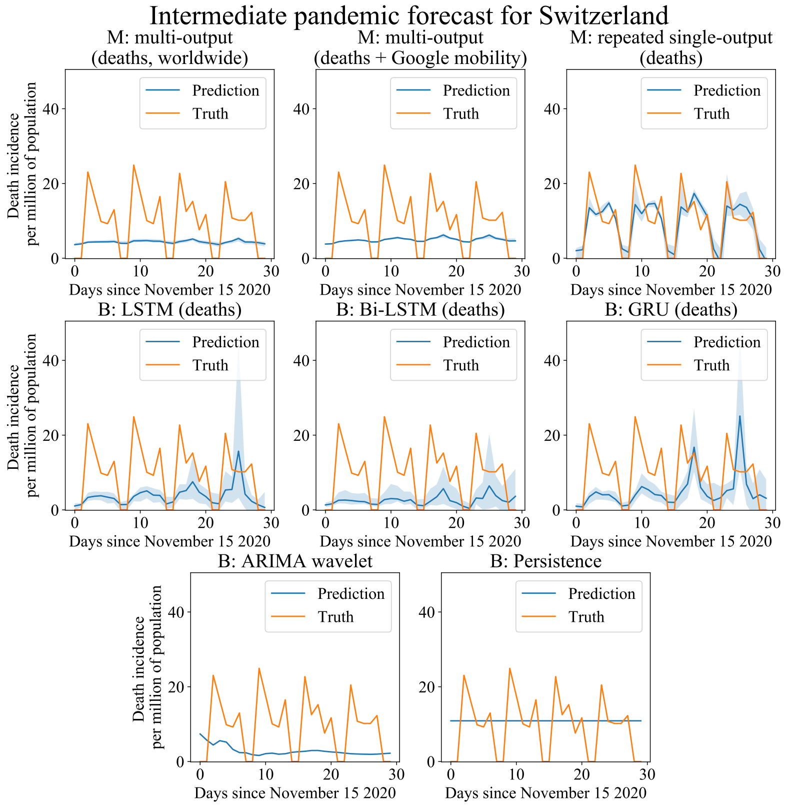


Figure 6.9: Intermediate forecasting for Switzerland.

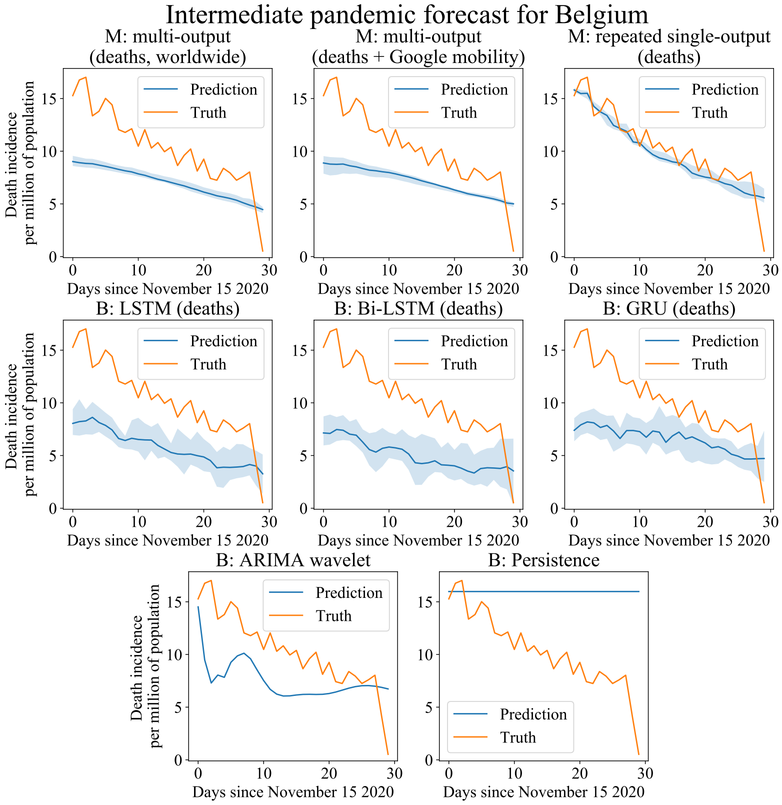


Figure 6.10: Intermediate forecasting for Belgium.

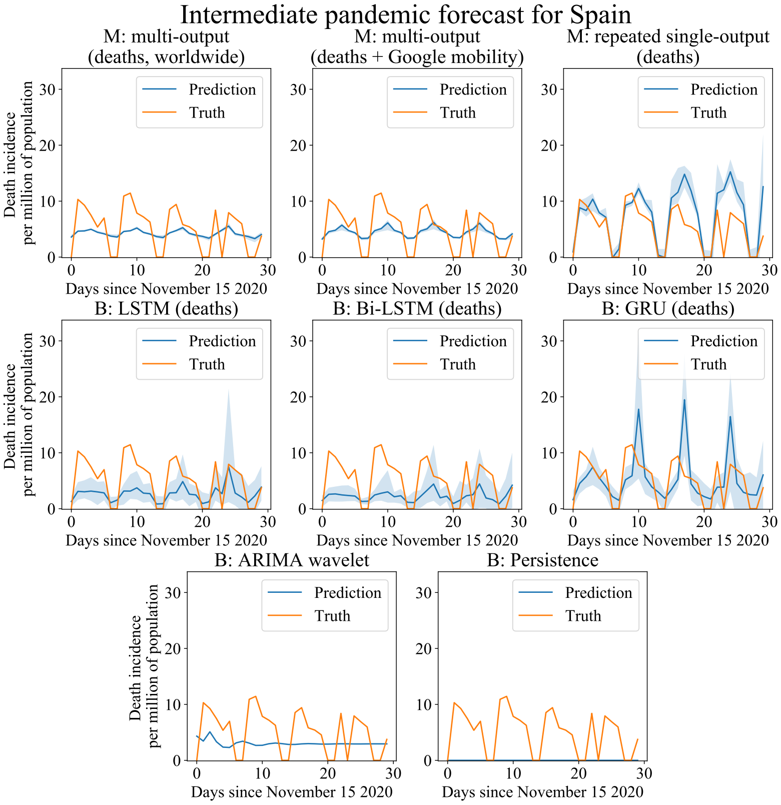


Figure 6.11: Intermediate forecasting for Spain.

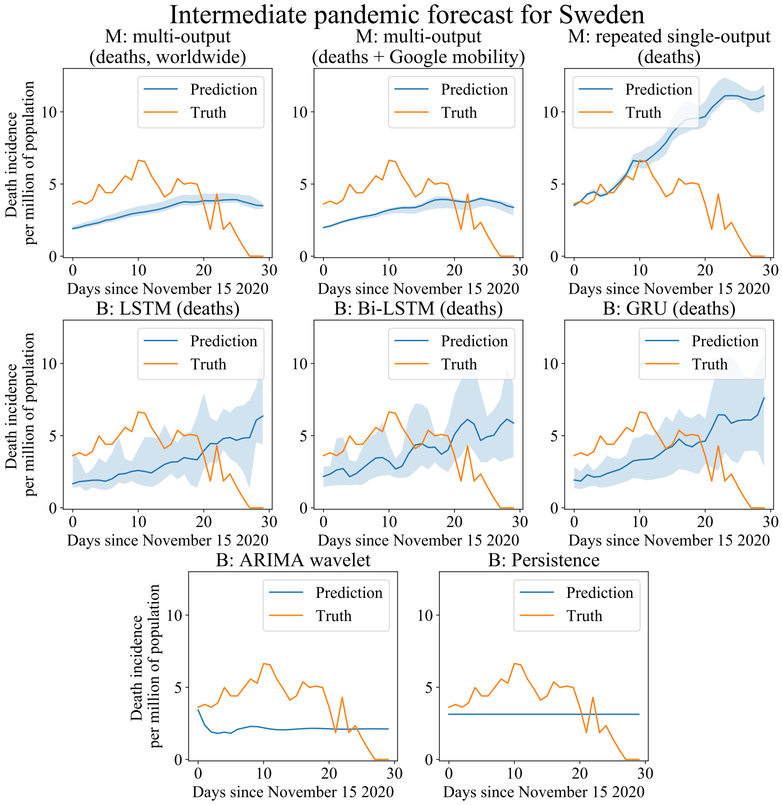


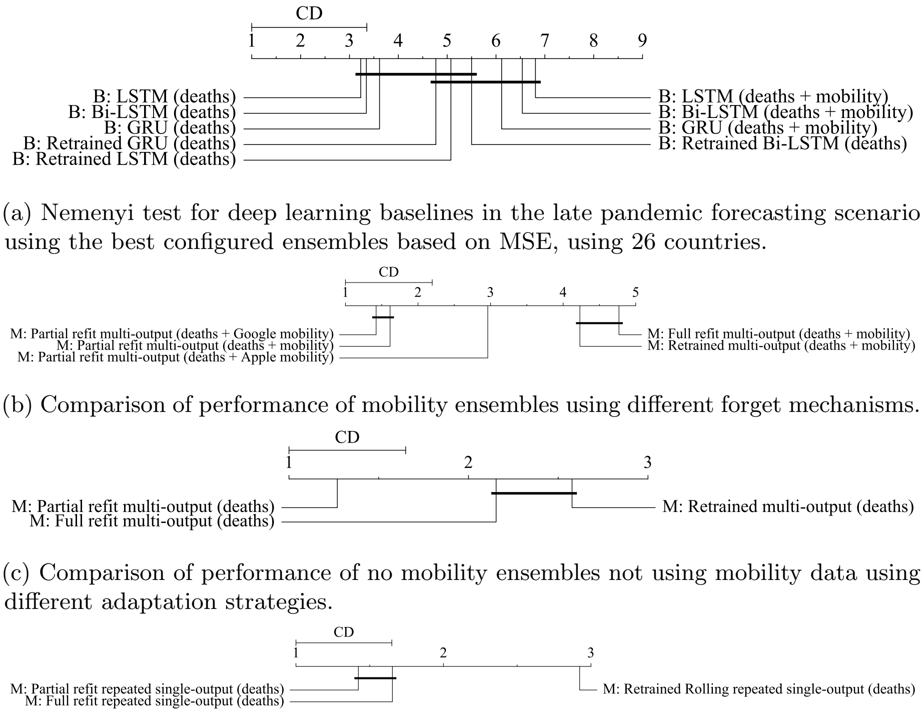
Figure 6.12: Intermediate forecasting for Sweden.

6.10. Other examples include Indonesia and Latvia and are shown in Figures 8.12 and 8.13 the Appendix. In this figure the weakness of the persistence baseline is apparent, as it is not able to cope with harsh declines. Our ensembles do this better. Whereas the multi-output ensembles capture the trend but estimate the daily deaths too low, our repeated single-output ensemble predicts them quite well. Some countries have a mix of trends and weekly cycles. Examples of these include Russia and the United States of America, shown in Figures 8.14 and 8.15 in the Appendix. For these countries our ensembles are able to capture a weekly cycle, but it becomes harder to capture the magnitude of each cycle. To show an example of a country with clear cycles and a mix of upward and downward trend, we show Spain in Figure 6.11. In this case the second cycle is larger than the first, and the third and fourth both smaller. The persistence baseline got initiated in a low point, vastly forecasting too low. The multi-output ensembles capture the cycles without interference of the trend. The repeated single-output ensemble predicts the increasing trend at the start and keeps enlarging their prediction, thus getting off the mark. Something similar happens when a trend suddenly changes. This is the case for instance in Hungary and Morocco, both shown in Figures 8.16 and 8.17 in the Appendix, but is most clear for the case of Sweden, which we show in this chapter in Figure 6.12. Here the sudden change in trend eludes the repeated single-output ensemble. The multi-output ensembles have a more smooth effect and predict closer to the truth.

This experiment has shown that while persistence is still a strong baseline, our ensembles gain value as the pandemic progresses. As extreme values become more extreme the ARIMA wavelet method that smooths signals is less able to predict the true values. Our methods particularly shine in countries with stronger cycles. Overall taken, the variance is quite low as compared to the bias, indicating some underfitting. If it is known that cycles exist for a country, our repeated single-output ensemble may be a good choice as forecaster. However, caution should be taken when using the predictions as a basis of policy making, as it fails to predict sudden shifts in trend. As there is more variance in the true observations, it is more challenging for the estimators to achieve a low MSE. Even though the MSE is similar to that of the previous scenario, an increase of data means that the methods were able to learn more complex patterns. This experiment has also shown that the use of Google mobility data is more beneficial than the use of Apple mobility data or the combination of the two sets, but an ensemble with these features is not necessarily better than one without mobility features.

6.3 Late pandemic forecasting

The third scenario is interesting as it represents a shift in data concept. We use two separate death incidence data sets. Both data sets were collected in the same manner by the ECDC. An important difference between the sets is the change in population size of each country. As we use this value as scaling factor for the incidence data, this is a form of concept drift to account for. To this end we combine three drift adaptation approaches with our ensembles. In this scenario the death incidence goes down, and the variance of true observations is a lot smaller than in the intermediate scenario. First we perform a preliminary study to compare the deep learning baselines with and without mobility features and with the retrain drift adaptation method. Then we compare the three strategies for our methods: the repeated single-output ensemble without mobility data, and the multi-output ensembles with Google mobility data and without mobility data.



(a) Nemenyi test for deep learning baselines in the late pandemic forecasting scenario using the best configured ensembles based on MSE, using 26 countries.

(b) Comparison of performance of mobility ensembles using different forget mechanisms.

(c) Comparison of performance of no mobility ensembles not using mobility data using different adaptation strategies.

(d) Comparison of performance of repeated single-output ensembles not using mobility data using different adaptation strategies.

Figure 6.13: Nemenyi test for late pandemic forecasting based on MSE using 26 countries. The subfigures compare differences in performance when using different adaptation strategies. Methods with lower average rank (left) are better than those with higher rank (right). Connected methods are not significantly different.

For the late pandemic forecast scenario Figure 6.13a shows that the deep learning baselines with mobility features are all outperformed by their counterparts without mobility features. We implemented the retrain drift adaptation strategy for these algorithms, but these did not yield an improvement over their counterparts not adapting to a change in normalising factor. We use the deep learning baselines without mobility features in our full comparisons. For each ensemble variant we compared the different adaptation strategies in a preliminary experiment. The resulting Nemenyi plot is shown in Figure 6.13b to 6.13d. In Figure 6.13b we show the comparison with different combinations of mobility data. Similar to the intermediate forecasting scenario the ensemble with Google mobility data is better than the others, although the difference is smaller now. As adaptation strategy the partial refit dominates. This strategy involved training the ensembles on the old data and updating the regressor weights on the new data. The same holds for the comparison of methods not using mobility features, shown in Figure 6.13c. For the repeated single-output ensemble the difference is less pronounced. From all three figures it follows that the retrain strategy performs worst. In this strategy most of the data is discarded, which would make the amount of data used by the methods in this scenario similar to the early forecasting scenario.

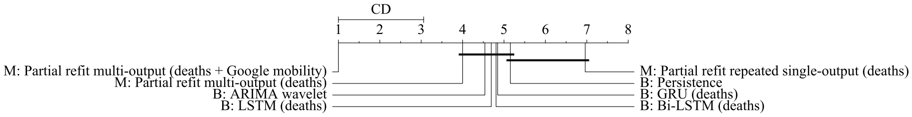


Figure 6.14: Nemenyi test for late pandemic forecasting using the best configured ensembles based on MSE, using 26 countries. Methods with lower average rank (left) are better than those with higher rank (right). Connected methods are not significantly different.

We use the ensembles with the refit strategy in our comparison to the baselines shown in Figure 6.14. From this figure it can be seen that our multi-output ensemble using Google mobility features now outperform all the other baselines on a significant level. Our repeated single-output ensemble performs worst here and is on par with the persistence baseline. For the reader interested in the model performance with unscaled data, we show the results based on RMSE using unscaled deaths in Table 8.3 in the Appendix.

In Figure 6.15 we show the error decomposition summary. The figure shows that compared to the previous scenarios, the MSE on average dropped considerably, except for our repeated single-output ensemble, which has a similar MSE to the previous scenario. This drop may mean that the ensembles were able to learn better, but it could also be partially explained by the lower variance in true observations for this scenario. For most methods the most important component is the bias, except for our repeated single-output ensemble. This ensemble has substantial variance. The MSE for our multi-output ensemble using mobility features is lower than all other methods, attributed to its lower bias and lower variance.

When we look at the following forecast plots we see that all our ensembles are able to fit the truth value to some extent. The credibility bounds of the repeated single-output ensemble grow as time progresses. This is to be expected, as predictions later on depend on earlier predictions. In countries with clear

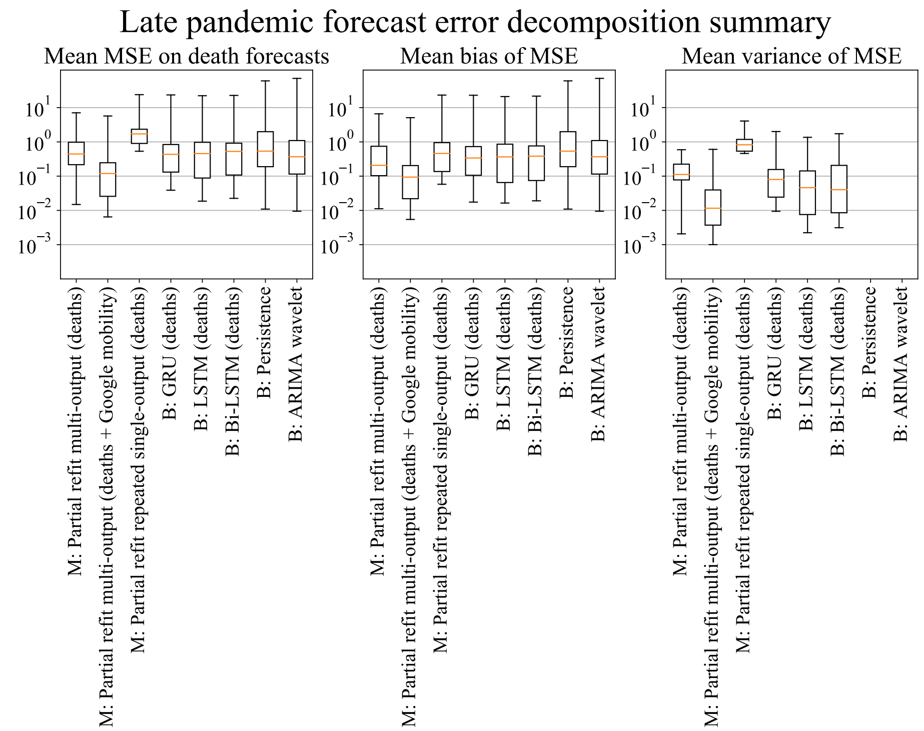


Figure 6.15: Late error decomposition summary.

cycles, like Germany, shown in Figure 6.16, the repeated single-output ensemble is still capable of closely forecasting the truth, although it does so with more uncertainty over time. The multi-output ensemble using Google mobility features smooths out the extreme values slightly, but also predicts the truth generally close. In countries with clear trends, like Greece, shown in Figure 6.17, but also Croatia or Lithuania, shown in Figures 8.18 and 8.19 in the Appendix, all of our ensembles forecast quite well. The mobility ensemble does so with the slimmest confidence interval. In cases where sudden increases or drops in daily new deaths occur, it becomes more difficult to make a good prediction, but even in these cases the extreme values are smoothed by the ensembles and predictions are pretty close to the truth. This behavior is seen for example in the countries Bulgaria, Latvia and Romania, shown in Figures 8.20 to 8.22 in the Appendix and Slovakia as shown in Figure 6.18.

This scenario has shown that updating the weights of the regressors in the ensembles is beneficial when concept drift has occurred. Especially the multi-output ensemble using mobility data makes strong predictions in this scenario. A possible explanation for this is that it uses the additional mobility data for which there was no abrupt concept drift. The results of this scenario suggest that it may be beneficial to check for concept drifts more often, as partially

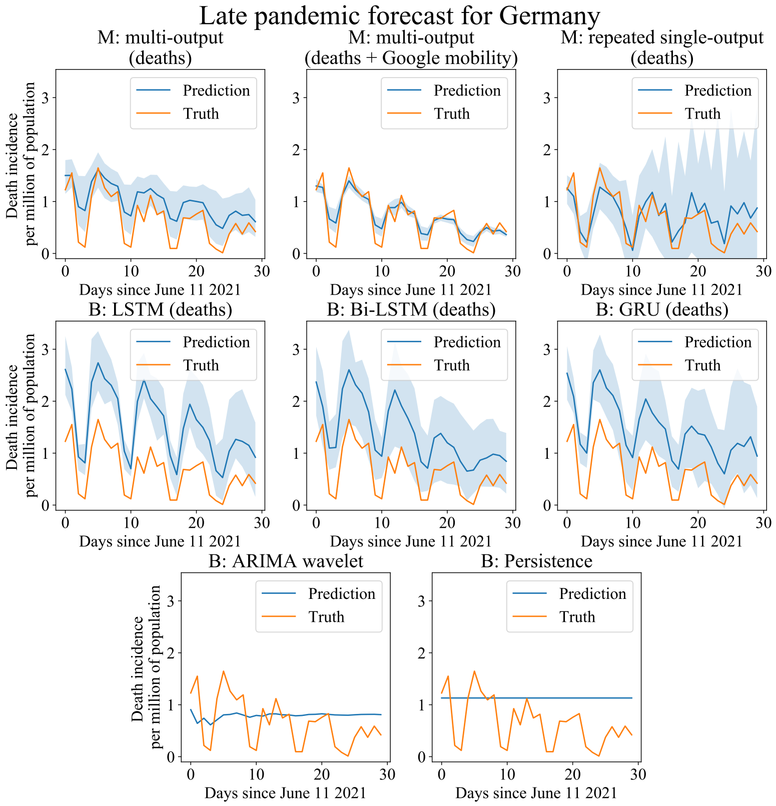


Figure 6.16: Late forecasting for Germany.

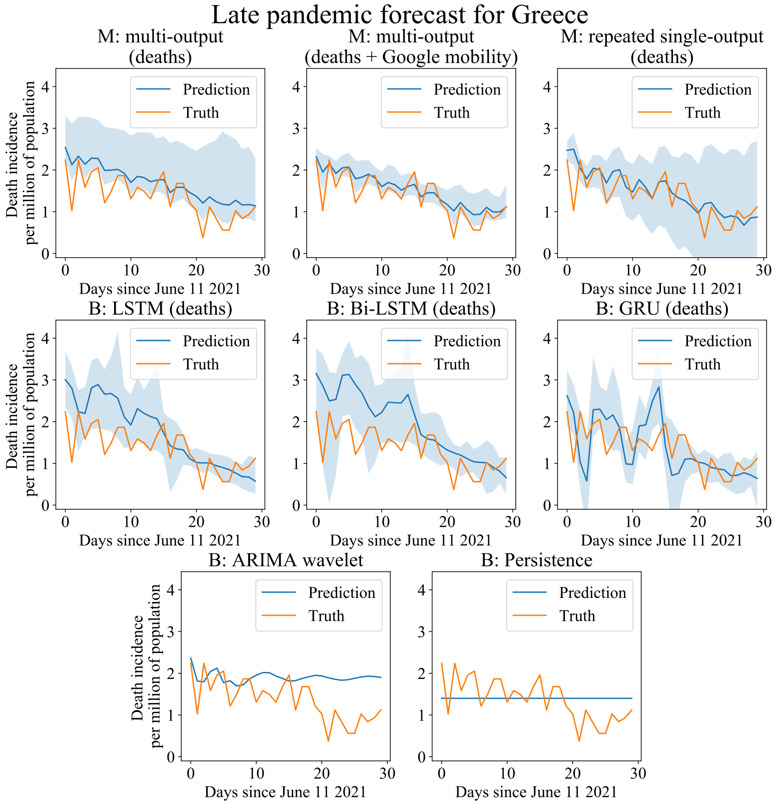


Figure 6.17: Late forecasting for Greece.

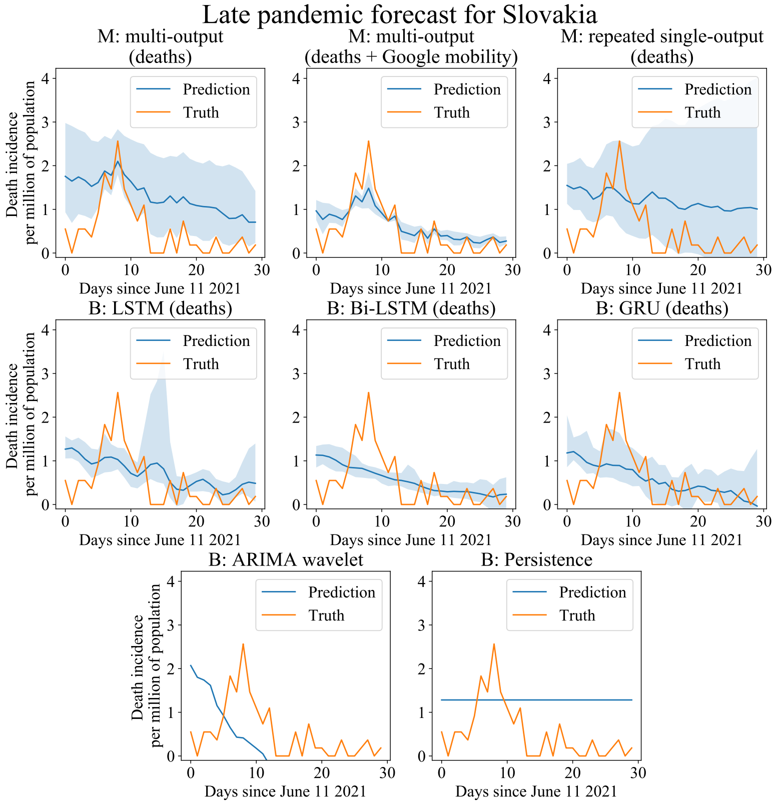


Figure 6.18: Late forecasting for Slovakia.

forgetting old data patterns improves prediction performance.

Chapter 7

Conclusion and future work

In this work we adapted the AutoML framework auto-sklearn to COVID-19 forecasting. We used mortality data and used mortality data and mobility data collected from 58 countries to construct automatically configured ensembles. We compared the performance of a multi-output ensemble and a repeated single-output ensemble and combine these with concept drift adaptation strategies.
Using three scenarios separated by time, we compared our ensembles with root mean squared error to a persistence baseline, an ARIMA wavelet method, a Bayesian inference method, and deep learning methods from literature. We found that when the pandemic is still novel, our methods are outperformed by baselines as simple as persistence. However, when the pandemic has progressed for just shy of a year in many countries, our ensembles are on par with the best baselines. Even later, when concept drift occurs due to a shift in data normalisation and possibly mutation of the virus, our methods significantly outperform the baselines, especially when using mobility data along mortality data. Our work has shown that our modified version of auto-sklearn does not perform as well as simple baselines within the first few months of the pandemic, but gains importance as time progresses. After a little less than a year we have gained enough data to be able to capture most cycles and trends occurring in the time series. Only when trends suddenly change, our predictions are eluded. Additionally, we discovered that when concept drift occurs by a change of data normalisation or possibly a mutation of the virus, refitting the models trained on the older data enables a major performance boost, especially when (unchanged) mobility data is used alongside the mortality data. Our methods create ensembles of pipelines of which the hyperparameters are automatically optimised, making them easy to use. The drift adaptation strategies are straightforward and the code of our framework is publicly available[1].

Still, caution should be taken when these forecasts are at the basis of policy making. As we used highly aggregated data, these forecasts do not tell the full tale. As Ioannidis et al. argue in [50], it is important to incorporate age

[1] https://github.com/jacotetteroo/AutoML4COVID-19

groups in decision making. Mortality rates per age group are however not readily available for the majority of countries. Oliver et al. [51] reinforce their view on the need for age specific data and argue more efforts must be taken to make more mobility data available.

In this work we used only simple features, which were the lags of observed data. It is interesting to see how much our results may improve when features are domain specific. Expert knowledge in the field of epidemics could greatly help in this regard. Nevertheless, in the late scenario where our ensembles were adapted to drift, these simple features worked splendidly. The open mobility data sources were a great asset, boosting the confidence of our forecasters. The intermediate and late scenarios have shown that the Google mobility set has more impact on the predictive performance than the Apple mobility set. The spatial features we used did not yield any improvement in prediction quality. The most probable reason for this is that we used data on the national level. The smoothing effect spatial lags have may be way more accurate if smaller areas are taken into consideration. Another reason for this may be that people from different countries do not interact with each other on the same level. It may well be that for certain countries the borders are often transgressed, while for others these remain closed. Given more detailed mobility data, this could be modeled. It would also enable the creation of realistic contact networks.

Our best performing ensembles utilised the concept drift adaptation strategy of refitting the ensembles once the drift has occurred. We applied this strategy once, at the point where the drift in concept was most clear. It may however be possible that multiple, smaller drifts occur during the time period captured in the time series. Finding the best moments to adapt the ensembles over time is another interesting future direction. Current AutoML systems use large batches of data at the same time to train their models. If these batches are too large, however, chances are the concept drift slips in undetected. A proper trade-off should be made between how much data is used in order to learn the data patterns sufficiently and to be able to detect concept drift within the used data. In this work we found that the Google mobility set provided useful characteristics for improving the accuracy of forecasts. This set contained the percentual increase or decrease of visits to a category of places. For future work it is interesting to find out which (combination) of these places are indispensable for the enhancement of the forecasts.

Bibliography

[1] World Health Organization. Statement on the first meeting of the international health regulations (2005) emergency committee regarding the outbreak of novel coronavirus (2019-ncov). Retrieved November 3, 2021 from `https://www.who.int/news-room/detail/23-01-2020-statement-on-the-meeting-of-the-international-health-regulations-(2005)-emergency-committee-regarding-the-outbreak-of-novel-coronavirus-(2019-ncov)`, 2020.

[2] Kaitano Dube, Godwell Nhamo, and David Chikodzi. Covid-19 cripples global restaurant and hospitality industry. *Current Issues in Tourism*, 24(11):1487–1490, 2021.

[3] Padam Bahadur Poudel, Mukti Ram Poudel, Aasish Gautam, Samiksha Phuyal, Chiran Krishna Tiwari, Nisha Bashyal, and Shila Bashyal. Covid-19 and its global impact on food and agriculture. *Journal of Biology and Today's World*, 9(5):221, 2020.

[4] William Ogilvy Kermack, A. G. McKendrick, and Gilbert Thomas Walker. A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 115(772):700–721, 1927.

[5] Romualdo Pastor-Satorras, Claudio Castellano, Piet Van Mieghem, and Alessandro Vespignani. Epidemic processes in complex networks. *Reviews of modern physics*, 87(3):925, 2015.

[6] Leon Danon, Ashley P Ford, Thomas House, Chris P Jewell, Matt J Keeling, Gareth O Roberts, Joshua V Ross, and Matthew C Vernon. Networks and the epidemiology of infectious disease. *Interdisciplinary perspectives on infectious diseases*, 2011, 2011.

[7] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[8] Kenneth L. Cooke. Stability analysis for a vector disease model. *The Rocky Mountain Journal of Mathematics*, 9(1):31–42, 1979.

[9] Quan-Hui Liu, Marco Ajelli, Alberto Aleta, Stefano Merler, Yamir Moreno, and Alessandro Vespignani. Measurability of the epidemic reproduction number in data-driven contact networks. *Proceedings of the National Academy of Sciences*, 115(50):12680–12685, 2018.

[10] Duygu Balcan, Vittoria Colizza, Bruno Gonçalves, Hao Hu, José J. Ramasco, and Alessandro Vespignani. Multiscale mobility networks and the spatial spreading of infectious diseases. *Proceedings of the National Academy of Sciences*, 106(51):21484–21489, 2009.

[11] Alberto Aleta, David Martín-Corral, Ana Pastore y Piontti, Marco Ajelli, Maria Litvinova, Matteo Chinazzi, Natalie E. Dean, M. Elizabeth Halloran, Ira M. Longini Jr, Stefano Merler, Alex Pentland, Alessandro Vespignani, Esteban Moro, and Yamir Moreno. Modelling the impact of testing, contact tracing and household quarantine on second waves of covid-19. *Nature Human Behaviour*, 4(9):964–971, Sep 2020.

[12] Chuang Yang, Zhiwen Zhang, Zipei Fan, Renhe Jiang, Quanjun Chen, Xuan Song, and Ryosuke Shibasaki. Epimob: Interactive visual analytics of citywide human mobility restrictions for epidemic control, 2021.

[13] Trevelyan J McKinley, Joshua V Ross, Rob Deardon, and Alex R Cook. Simulation-based bayesian inference for epidemic models. *Computational Statistics & Data Analysis*, 71:434–447, 2014.

[14] Seth Flaxman, Swapnil Mishra, Axel Gandy, H. Juliette T. Unwin, Thomas A. Mellan, Helen Coupland, Charles Whittaker, Harrison Zhu, Tresnia Berah, Jeffrey W. Eaton, Mélodie Monod, Pablo N. Perez-Guzman, Nora Schmit, Lucia Cilloni, Kylie E. C. Ainslie, Marc Baguelin, Adhiratha Boonyasiri, Olivia Boyd, Lorenzo Cattarino, Laura V. Cooper, Zulma Cucunubá, Gina Cuomo-Dannenburg, Amy Dighe, Bimandra Djaafara, Ilaria Dorigatti, Sabine L. van Elsland, Richard G. FitzJohn, Katy A. M. Gaythorpe, Lily Geidelberg, Nicholas C. Grassly, William D. Green, Timothy Hallett, Arran Hamlet, Wes Hinsley, Ben Jeffrey, Edward Knock, Daniel J. Laydon, Gemma Nedjati-Gilani, Pierre Nouvellet, Kris V. Parag, Igor Siveroni, Hayley A. Thompson, Robert Verity, Erik Volz, Caroline E. Walters, Haowei Wang, Yuanrong Wang, Oliver J. Watson, Peter Winskill, Xiaoyue Xi, Patrick G. T. Walker, Azra C. Ghani, Christl A. Donnelly, Steven Riley, Michaela A. C. Vollmer, Neil M. Ferguson, Lucy C. Okell, Samir Bhatt, and Imperial College COVID-19 Response Team. Estimating the effects of non-pharmaceutical interventions on covid-19 in europe. *Nature*, 584(7820):257–261, Aug 2020.

[15] R.J. Hyndman and G. Athanasopoulos. *Forecasting: principles and practice.* OTexts: Melbourne, Australia, 2 edition, 2008.

[16] Pavan Kumar, Himangshu Kalita, Shashikanta Patairiya, Yagya Datt Sharma, Chintan Nanda, Meenu Rani, Jamal Rahmani, and Akshaya Srikanth Bhagavathula. Forecasting the dynamics of covid-19 pandemic in top 15 countries in april 2020: Arima model with machine learning approach. *MedRxiv*, 2020.

[17] Saleh I. Alzahrani, Ibrahim A. Aljamaan, and Ebrahim A. Al-Fakih. Forecasting the spread of the covid-19 pandemic in saudi arabia using arima prediction model under current public health interventions. *Journal of Infection and Public Health*, 13(7):914–919, 2020.

[18] Hirotogu Akaike. *Information Theory and an Extension of the Maximum Likelihood Principle*, pages 199–213. Springer New York, New York, NY, 1998.

[19] Tanujit Chakraborty and Indrajit Ghosh. Real-time forecasts and risk assessment of novel coronavirus (covid-19) cases: A data-driven analysis. *Chaos, Solitons & Fractals*, 135:109850, 2020.

[20] Yuexin Wu, Yiming Yang, Hiroshi Nishiura, and Masaya Saitoh. Deep learning for epidemiological predictions. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 1085–1088, New York, NY, USA, 2018. Association for Computing Machinery.

[21] Emily L. Aiken, Andre T. Nguyen, Cecile Viboud, and Mauricio Santillana. Toward the use of neural networks for influenza prediction at multiple spatial resolutions. *Science Advances*, 7(25):eabb1237, 2021.

[22] Bofeng Fu, Yaodong Yang, Yu Ma, Jianye Hao, Siqi Chen, Shuang Liu, Tiegang Li, Zhenyu Liao, and Xianglei Zhu. Attention-based recurrent multi-channel neural network for influenza epidemic prediction. In *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1245–1248, 2018.

[23] Vinay Kumar Reddy Chimmula and Lei Zhang. Time series forecasting of covid-19 transmission in canada using lstm networks. *Chaos, Solitons & Fractals*, 135:109864, 2020.

[24] Abdelhafid Zeroual, Fouzi Harrou, Abdelkader Dairi, and Ying Sun. Deep learning methods for forecasting covid-19 time-series data: A comparative study. *Chaos, Solitons & Fractals*, 140:110121, 2020.

[25] Sourabh Shastri, Kuljeet Singh, Sachin Kumar, Paramjit Kour, and Vibhakar Mansotra. Time series forecasting of covid-19 using deep learning models: India-usa comparative case study. *Chaos, Solitons & Fractals*, 140:110227, 2020.

[26] İsmail Kırbaş, Adnan Sözen, Azim Doğuş Tuncer, and Fikret Şinasi Kazancıoğlu. Comparative analysis and forecasting of covid-19 cases in various european countries with arima, narnn and lstm approaches. *Chaos, Solitons & Fractals*, 138:110015, 2020.

[27] Farah Shahid, Aneela Zameer, and Muhammad Muneeb. Predictions for covid-19 with deep learning models of lstm, gru and bi-lstm. *Chaos, Solitons & Fractals*, 140:110212, 2020.

[28] Frank Hutter, Holger H. Hoos, Kevin Leyton-Brown, and Kevin P. Murphy. An experimental investigation of model-based parameter optimisation: Spo and beyond. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, GECCO '09, page 271–278, New York, NY, USA, 2009. Association for Computing Machinery.

[29] Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In Carlos A. Coello Coello, editor, *Learning and Intelligent Optimization*, pages 507–523, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[30] Chris Thornton, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, page 847–855, New York, NY, USA, 2013. Association for Computing Machinery.

[31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[32] Randal S. Olson and Jason H. Moore. Tpot: A tree-based pipeline optimization tool for automating machine learning. In Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, editors, *Proceedings of the Workshop on Automatic Machine Learning*, volume 64 of *Proceedings of Machine Learning Research*, pages 66–74, New York, New York, USA, 24 Jun 2016. PMLR.

[33] Erin LeDell and Sebastien Poirier. H2o automl: Scalable automatic machine learning. In *Proceedings of the AutoML Workshop at ICML*, volume 2020, 2020.

[34] Haifeng Jin, Qingquan Song, and Xia Hu. Auto-keras: An efficient neural architecture search system. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 1946–1956, New York, NY, USA, 2019. Association for Computing Machinery.

[35] Hector Mendoza, Aaron Klein, Matthias Feurer, Jost Tobias Springenberg, Matthias Urban, Michael Burkart, Maximilian Dippel, Marius Lindauer, and Frank Hutter. *Towards Automatically-Tuned Deep Neural Networks*, pages 135–149. Springer International Publishing, Cham, 2019.

[36] Tao Han, Francisco Nauber Bernardo Gois, Ramsés Oliveira, Luan Rocha Prates, and Magda Moura de Almeida Porto. Modeling the progression of covid-19 deaths using kalman filter and automl. *Soft Computing*, Jan 2021.

[37] Joao Alexandre Lobo Marques, Francisco Nauber Bernardo Gois, José Xavier-Neto, and Simon James Fong. *Artificial Intelligence Prediction for the COVID-19 Data Based on LSTM Neural Networks and H2O AutoML*, pages 69–87. Springer International Publishing, Cham, 2021.

[38] ECDC. Historical data (to 14 december 2020) on the daily number of new reported covid-19 cases and deaths worldwide. Retrieved November 3, 2021 from `https://www.ecdc.europa.eu/en/publications-data/download-todays-data-geographic-distribution-covid-19-cases-worldwide`, 2021.

[39] ECDC. Data on the daily number of new reported covid-19 cases and deaths by eu/eea country, 2021.

[40] Apple. Mobility trends reports. Retrieved November 3, 2021 from `https://covid19.apple.com/mobility`, 2021.

[41] Google. Covid-19 community mobility reports. Retrieved November 3, 2021 from `https://www.google.com/covid19/mobility`, 2021.

[42] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. Openml: Networked science in machine learning. *SIGKDD Explor. Newsl.*, 15(2):49–60, jun 2014.

[43] Can Wang, Mitra Baratchi, Thomas Bäck, Holger H. Hoos, Steffen Limmer, and Markus Olhofer. Automated machine learning with enhanced feature engineering for time series forecasting. *Under review*, 2021.

[44] João Gama, Indrundefined Žliobaitundefined, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Comput. Surv.*, 46(4), mar 2014.

[45] Bilge Celik and Joaquin Vanschoren. Adaptation strategies for automated machine learning on evolving data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(9):3067–3078, 2021.

[46] Kamalich Muniz-Rodriguez, Gerardo Chowell, Chi-Hin Cheung, Dongyu Jia, Po-Ying Lai, Yiseul Lee, Manyun Liu, Sylvia K Ofori, Kimberlyn M Roosa, Lone Simonsen, et al. Doubling time of the covid-19 epidemic by province, china. *Emerging infectious diseases*, 26(8):1912, 2020.

[47] Jeffrey C Schlimmer and Richard H Granger. Incremental learning from noisy data. *Machine learning*, 1(3):317–354, 1986.

[48] Sergio J Rey and Luc Anselin. Pysal: A python library of spatial analytical methods. In *Handbook of applied spatial analysis*, pages 175–193. Springer, 2010.

[49] Peter Bjorn Nemenyi. *Distribution-free multiple comparisons.* Princeton University, 1963.

[50] John P.A. Ioannidis, Sally Cripps, and Martin A. Tanner. Forecasting for covid-19 has failed. *International Journal of Forecasting*, 2020.

[51] Nuria Oliver, Bruno Lepri, Harald Sterly, Renaud Lambiotte, Sébastien Deletaille, Marco De Nadai, Emmanuel Letouzé, Albert Ali Salah, Richard Benjamins, Ciro Cattuto, Vittoria Colizza, Nicolas de Cordes, Samuel P. Fraiberger, Till Koebe, Sune Lehmann, Juan Murillo, Alex Pentland, Phuong N Pham, Frédéric Pivetta, Jari Saramäki, Samuel V. Scarpino, Michele Tizzoni, Stefaan Verhulst, and Patrick Vinck. Mobile phone data for informing public health actions across the covid-19 pandemic life cycle. *Science Advances*, 6(23):eabc0764, 2020.

Chapter 8

Appendix

We show the result tables and the referenced figures that were not shown in the main body of this work on the following pages. The full collection of figures for each scenario and country can be found externally[1].

[1] https://github.com/jacotetteroo/AutoML4COVID-19

Table 8.1: RMSE of the early forecasting scenario. To save space, we abbreviated our methods (M) and the baselines (B). MO is multi-output, RSO is repeated single-output, P is persistence, AW is ARIMA wavelet and BI is Bayesian inference. We present the data used for features as d for deaths and d+m for deaths together with mobility. Bold results are the lowest for corresponding countries on a 5% significance level.

Country	M: MO:d	M: MO:d+m	RSO:d	B: GRU:d+m	B: LSTM:d	B: Bi-LSTM:d	B: P	B: AW	B: BI
Argentina	41.354 ± 4.52	127.35 ± 15.63	139.271 ± 0.0	45.895 ± 17.78	12.768 ± 5.83	19.317 ± 6.01	**5.581 ± 0.0**	33.553 ± 0.0	NA
Australia	2.993 ± 0.35	12.012 ± 5.84	4.575 ± 0.0	15.817 ± 6.85	5.748 ± 2.14	8.91 ± 7.89	1.648 ± 0.0	**1.602 ± 0.0**	NA
Austria	23.5 ± 2.3	24.294 ± 1.73	33.166 ± 0.0	28.73 ± 11.27	22.832 ± 5.99	20.243 ± 5.59	**9.584 ± 0.0**	10.658 ± 0.0	15.669 ± 0.0
Belgium	89.93 ± 3.99	**87.77 ± 1.45**	170.513 ± 0.0	94.806 ± 8.57	**86.043 ± 12.05**	96.303 ± 8.0	113.53 ± 0.0	123.404 ± 0.0	169.111 ± 0.0
Brazil	274.133 ± 88.1	493.795 ± 43.18	504.519 ± 0.0	196.315 ± 63.36	242.716 ± 56.22	241.136 ± 124.58	203.327 ± 0.0	**173.594 ± 0.0**	NA
Bulgaria	8.998 ± 0.88	13.29 ± 2.46	44.439 ± 0.0	6.223 ± 2.38	5.977 ± 4.24	3.855 ± 3.28	**2.79 ± 0.0**	6.581 ± 0.0	NA
Cambodia	0.013 ± 0.13	4.344 ± 1.6	**0.0 ± 0.0**	13.281 ± 6.14	1.956 ± 0.52	1.428 ± 1.02	0.0 ± 0.0	0.0 ± 0.0	NA
Canada	79.209 ± 5.52	83.785 ± 7.03	107.839 ± 0.0	65.287 ± 16.98	81.182 ± 18.55	87.747 ± 15.33	**65.144 ± 0.0**	69.162 ± 0.0	NA
Chile	25.738 ± 6.42	52.691 ± 5.76	54.27 ± 0.0	17.614 ± 7.86	8.564 ± 3.63	9.161 ± 5.57	**3.91 ± 0.0**	14.301 ± 0.0	NA
Colombia	53.342 ± 9.49	171.324 ± 16.51	145.253 ± 0.0	51.454 ± 20.43	20.53 ± 15.81	27.391 ± 8.58	**5.503 ± 0.0**	16.675 ± 0.0	NA
Croatia	6.869 ± 0.76	9.499 ± 1.08	14.895 ± 0.0	4.248 ± 2.02	3.537 ± 1.35	3.373 ± 2.65	**2.405 ± 0.0**	2.72 ± 0.0	NA
Czechia	23.26 ± 1.6	26.208 ± 2.83	76.303 ± 0.0	37.052 ± 17.17	10.751 ± 6.08	25.973 ± 9.1	**5.663 ± 0.0**	11.106 ± 0.0	NA
Denmark	10.801 ± 1.18	12.638 ± 1.35	8.86 ± 0.0	28.148 ± 6.89	12.551 ± 5.03	15.1 ± 2.62	**6.141 ± 0.0**	6.358 ± 0.0	7.153 ± 0.0
Egypt	10.229 ± 0.98	47.918 ± 23.89	24.466 ± 0.0	89.7 ± 34.18	78.193 ± 38.36	75.835 ± 30.34	9.543 ± 0.0	**9.431 ± 0.0**	NA
Estonia	1.989 ± 0.12	2.585 ± 0.23	2.053 ± 0.0	3.534 ± 1.8	4.266 ± 1.01	3.964 ± 1.72	1.282 ± 0.0	**1.171 ± 0.0**	NA
Finland	12.127 ± 0.09	11.927 ± 0.26	13.657 ± 0.0	14.04 ± 7.97	**10.572 ± 0.95**	12.31 ± 3.61	12.021 ± 0.0	11.368 ± 0.0	NA
France	**305.694 ± 11.62**	307.635 ± 12.29	587.62 ± 0.0	500.964 ± 66.49	454.916 ± 116.9	574.072 ± 64.59	318.652 ± 0.0	512.985 ± 0.0	711.119 ± 0.0
Germany	183.202 ± 28.42	219.541 ± 27.76	200.004 ± 0.0	382.52 ± 121.69	186.651 ± 73.65	198.462 ± 37.18	**98.849 ± 0.0**	115.643 ± 0.0	253.118 ± 0.0
Greece	11.762 ± 1.64	17.349 ± 2.92	25.657 ± 0.0	15.734 ± 6.35	8.509 ± 4.46	12.684 ± 6.86	4.567 ± 0.0	**3.97 ± 0.0**	NA
Hungary	19.559 ± 2.71	24.164 ± 1.98	53.403 ± 0.0	18.261 ± 12.23	12.209 ± 5.12	16.532 ± 10.18	9.067 ± 0.0	**8.93 ± 0.0**	NA
India	134.605 ± 27.76	1065.688 ± 368.25	474.053 ± 0.0	955.525 ± 422.27	294.434 ± 176.72	152.52 ± 82.39	**47.454 ± 0.0**	120.014 ± 0.0	NA
Indonesia	29.061 ± 8.44	110.263 ± 83.3	57.054 ± 0.0	225.487 ± 93.33	165.987 ± 78.5	136.06 ± 41.04	**14.375 ± 0.0**	14.501 ± 0.0	NA
Ireland	60.13 ± 0.54	58.953 ± 0.36	67.687 ± 0.0	60.418 ± 1.66	56.505 ± 0.7	**56.087 ± 0.91**	68.658 ± 0.0	72.254 ± 0.0	NA
Israel	14.208 ± 1.93	21.143 ± 1.54	14.303 ± 0.0	15.561 ± 4.43	6.025 ± 3.79	15.347 ± 8.17	**3.273 ± 0.0**	6.593 ± 0.0	NA
Italy	243.085 ± 18.38	264.188 ± 16.93	419.14 ± 0.0	526.614 ± 111.56	366.857 ± 153.24	610.103 ± 52.11	**238.636 ± 0.0**	251.035 ± 0.0	465.03 ± 0.0
Japan	26.528 ± 0.41	26.058 ± 3.59	28.968 ± 0.0	61.139 ± 70.54	47.833 ± 21.03	38.959 ± 12.23	**25.505 ± 0.0**	25.912 ± 0.0	NA
Latvia	**1.121 ± 0.02**	2.258 ± 0.31	3.082 ± 0.0	2.077 ± 0.57	1.473 ± 0.62	1.163 ± 0.34	1.363 ± 0.0	1.14 ± 0.0	NA
Lithuania	4.035 ± 0.35	5.505 ± 0.6	7.305 ± 0.0	6.533 ± 1.8	2.146 ± 0.92	4.027 ± 1.95	**1.0 ± 0.0**	1.257 ± 0.0	NA
Luxembourg	2.258 ± 0.18	2.578 ± 0.19	2.33 ± 0.0	2.706 ± 0.97	3.721 ± 0.82	2.134 ± 1.02	**1.604 ± 0.0**	1.611 ± 0.0	NA
Malaysia	4.504 ± 0.59	23.852 ± 13.56	4.629 ± 0.0	27.526 ± 11.08	5.798 ± 9.71	11.411 ± 3.6	**1.626 ± 0.0**	1.65 ± 0.0	NA
Mexico	155.161 ± 43.82	318.095 ± 33.64	373.485 ± 0.0	99.241 ± 30.37	61.92 ± 69.23	**60.043 ± 6.51**	80.603 ± 0.0	96.987 ± 0.0	NA
Morocco	14.872 ± 2.52	51.79 ± 12.31	30.998 ± 0.0	29.696 ± 14.76	24.43 ± 17.1	15.662 ± 10.53	**3.919 ± 0.0**	7.246 ± 0.0	NA
Netherlands	72.801 ± 3.51	73.322 ± 3.7	115.061 ± 0.0	67.139 ± 15.58	125.142 ± 37.52	175.253 ± 17.32	**62.217 ± 0.0**	64.733 ± 0.0	NA
New Zealand	1.087 ± 0.09	4.578 ± 1.04	1.464 ± 0.0	5.207 ± 1.49	14.078 ± 1.98	15.4 ± 2.6	**0.845 ± 0.0**	0.914 ± 0.0	NA
Norway	7.411 ± 0.45	8.982 ± 0.77	7.392 ± 0.0	19.225 ± 7.06	6.271 ± 2.38	13.889 ± 5.1	5.682 ± 0.0	**4.878 ± 0.0**	5.444 ± 0.0
Philippines	20.472 ± 4.83	96.462 ± 44.01	25.761 ± 0.0	97.345 ± 43.75	44.579 ± 21.57	49.754 ± 13.62	**11.174 ± 0.0**	13.828 ± 0.0	NA
Poland	45.283 ± 6.87	75.907 ± 8.71	169.435 ± 0.0	34.857 ± 16.52	21.79 ± 8.37	37.742 ± 23.62	**12.884 ± 0.0**	21.551 ± 0.0	NA
Portugal	37.619 ± 4.41	41.711 ± 4.74	30.716 ± 0.0	73.055 ± 25.83	23.994 ± 14.92	21.638 ± 5.85	**16.0 ± 0.0**	16.148 ± 0.0	NA
Romania	34.486 ± 6.78	45.292 ± 4.33	57.221 ± 0.0	23.667 ± 12.32	20.902 ± 8.51	42.861 ± 17.44	**11.139 ± 0.0**	13.742 ± 0.0	NA
Russia	60.174 ± 14.06	194.137 ± 26.04	169.118 ± 0.0	37.238 ± 10.02	121.799 ± 71.22	91.637 ± 33.32	**31.65 ± 0.0**	43.394 ± 0.0	NA
Saudi Arabia	5.142 ± 2.46	26.029 ± 11.2	17.688 ± 0.0	19.958 ± 9.73	18.066 ± 10.17	26.088 ± 10.19	**3.151 ± 0.0**	4.398 ± 0.0	NA
Singapore	0.661 ± 0.02	1.619 ± 1.02	0.707 ± 0.0	4.221 ± 2.08	1.753 ± 0.71	1.081 ± 0.63	**0.655 ± 0.0**	0.656 ± 0.0	NA
Slovakia	3.955 ± 0.44	9.951 ± 1.12	9.58 ± 0.0	7.096 ± 2.33	7.319 ± 2.72	10.285 ± 3.03	**0.845 ± 0.0**	2.633 ± 0.0	NA
Slovenia	6.126 ± 0.5	6.277 ± 0.49	16.406 ± 0.0	4.087 ± 1.34	7.605 ± 1.85	6.797 ± 2.19	**1.69 ± 0.0**	2.565 ± 0.0	NA
South Africa	26.406 ± 3.0	107.707 ± 14.94	76.287 ± 0.0	34.929 ± 17.03	16.413 ± 5.31	10.717 ± 4.0	**3.566 ± 0.0**	19.513 ± 0.0	NA
South Korea	5.495 ± 0.49	10.553 ± 6.12	2.752 ± 0.0	17.855 ± 7.13	6.161 ± 18.1	13.365 ± 6.03	1.626 ± 0.0	**1.534 ± 0.0**	NA
Spain	**269.582 ± 13.66**	281.48 ± 15.63	595.522 ± 0.0	377.805 ± 57.87	305.219 ± 93.04	452.705 ± 68.32	272.154 ± 0.0	309.906 ± 0.0	560.574 ± 0.0
Sweden	37.616 ± 1.67	37.054 ± 0.82	67.105 ± 0.0	**34.177 ± 3.48**	59.62 ± 8.69	48.865 ± 12.21	40.448 ± 0.0	40.732 ± 0.0	87.128 ± 0.0
Switzerland	31.878 ± 1.98	33.36 ± 3.33	42.893 ± 0.0	35.04 ± 8.92	74.653 ± 12.71	46.545 ± 17.64	**24.949 ± 0.0**	25.054 ± 0.0	27.469 ± 0.0
Taiwan	0.571 ± 0.19	1.824 ± 1.14	**0.0 ± 0.0**	5.531 ± 3.19	3.949 ± 1.77	2.765 ± 1.66	0.0 ± 0.0	0.0 ± 0.0	NA
Thailand	1.861 ± 0.32	23.962 ± 10.99	1.89 ± 0.0	59.248 ± 24.08	10.858 ± 5.84	9.64 ± 2.2	1.488 ± 0.0	**1.284 ± 0.0**	NA
Turkey	102.581 ± 19.34	149.829 ± 12.39	79.585 ± 0.0	96.104 ± 39.44	135.759 ± 48.3	210.977 ± 72.03	**62.767 ± 0.0**	65.106 ± 0.0	NA
Ukraine	30.311 ± 3.44	76.138 ± 11.95	78.34 ± 0.0	29.641 ± 16.01	20.649 ± 8.51	25.078 ± 5.59	**5.825 ± 0.0**	15.339 ± 0.0	NA
United Arab Emirates	**3.803 ± 0.16**	4.39 ± 2.23	5.05 ± 0.0	9.317 ± 3.94	8.172 ± 3.1	9.718 ± 3.9	4.432 ± 0.0	4.402 ± 0.0	NA
United Kingdom	468.509 ± 12.97	464.586 ± 11.03	729.752 ± 0.0	419.615 ± 35.6	446.803 ± 71.56	583.2 ± 59.99	**393.923 ± 0.0**	457.096 ± 0.0	810.513 ± 0.0
United States	1261.142 ± 98.79	1225.59 ± 68.22	2023.314 ± 0.0	**957.563 ± 293.67**	1507.987 ± 402.61	1664.455 ± 591.83	1028.931 ± 0.0	1201.339 ± 0.0	NA
Uruguay	0.779 ± 0.01	1.984 ± 1.29	0.886 ± 0.0	2.71 ± 1.33	5.589 ± 2.46	3.673 ± 3.46	0.845 ± 0.0	**0.758 ± 0.0**	NA
Vietnam	0.233 ± 0.74	25.172 ± 8.6	0.535 ± 0.0	61.656 ± 26.6	11.445 ± 3.05	8.356 ± 5.99	**0.0 ± 0.0**	0.103 ± 0.0	NA

Table 8.2: RMSE of the intermediate forecasting scenario. To save space, we abbreviated our methods (M) and the baselines (B). MO is multi-output, RSO is repeated single-output, P is persistence and AW is ARIMA wavelet. We present the data used for features as d for deaths, d+Gm for deaths together with Google mobility and dw for deaths trained on additional countries worldwide. The deep learning baselines use only deaths features. Bold results are the lowest for corresponding countries on a 5% significance level.

Country	M: MO:dw	M: MO:d+Gm	M: RSO:d	LSTM	Bi-LSTM	GRU	Persistence	ARIMA wavelet
Argentina	**66.978 ± 9.02**	132.72 ± 6.13	71.224 ± 12.32	81.043 ± 10.77	106.665 ± 7.34	111.14 ± 10.95	104.163 ± 0.0	113.392 ± 0.0
Australia	2.072 ± 0.33	3.726 ± 1.31	1.653 ± 0.69	1.386 ± 0.65	1.337 ± 0.32	0.768 ± 0.33	**0.183 ± 0.0**	1.169 ± 0.0
Austria	65.684 ± 2.46	68.149 ± 3.19	**31.384 ± 0.62**	61.559 ± 2.18	66.199 ± 2.27	67.219 ± 3.07	47.052 ± 0.0	88.662 ± 0.0
Belgium	47.36 ± 2.42	47.665 ± 2.93	**18.248 ± 0.73**	53.395 ± 3.69	58.499 ± 5.11	69.618 ± 6.77	76.276 ± 0.0	46.515 ± 0.0
Brazil	239.393 ± 9.55	413.655 ± 55.03	**159.886 ± 4.99**	256.533 ± 44.43	177.092 ± 48.35	210.016 ± 63.43	238.741 ± 0.0	226.523 ± 0.0
Bulgaria	96.829 ± 1.63	75.533 ± 2.66	66.24 ± 1.58	92.185 ± 2.88	99.703 ± 3.4	97.31 ± 3.2	**61.91 ± 0.0**	112.458 ± 0.0
Cambodia	0.021 ± 0.05	0.113 ± 0.11	1.081 ± 0.47	1.527 ± 0.69	1.834 ± 0.39	1.613 ± 0.93	**0.0 ± 0.0**	0.0 ± 0.0
Canada	**19.978 ± 0.81**	35.479 ± 2.64	22.733 ± 5.28	32.413 ± 5.61	25.238 ± 2.88	28.127 ± 6.68	34.042 ± 0.0	38.337 ± 0.0
Chile	22.787 ± 0.96	28.242 ± 2.13	21.631 ± 1.72	**19.341 ± 4.56**	23.722 ± 2.98	24.41 ± 3.59	22.029 ± 0.0	28.134 ± 0.0
Colombia	18.139 ± 2.37	18.461 ± 2.12	37.756 ± 9.76	36.469 ± 11.08	65.748 ± 27.6	72.034 ± 30.26	**13.733 ± 0.0**	20.206 ± 0.0
Croatia	36.427 ± 0.47	33.915 ± 0.68	21.968 ± 1.29	37.474 ± 3.13	41.186 ± 3.02	39.38 ± 2.02	**18.214 ± 0.0**	44.293 ± 0.0
Czechia	64.93 ± 4.32	51.139 ± 4.7	**28.585 ± 1.99**	63.597 ± 2.87	71.5 ± 3.33	76.238 ± 3.98	59.95 ± 0.0	38.404 ± 0.0
Denmark	**2.756 ± 0.18**	5.163 ± 0.79	5.614 ± 0.37	4.401 ± 0.51	3.695 ± 0.59	4.033 ± 0.49	5.908 ± 0.0	4.155 ± 0.0
Egypt	5.252 ± 0.98	156.058 ± 18.68	19.453 ± 5.5	6.364 ± 1.19	**3.692 ± 1.43**	4.528 ± 0.85	6.199 ± 0.0	4.832 ± 0.0
Estonia	2.434 ± 0.02	**1.878 ± 0.04**	2.206 ± 0.19	2.704 ± 0.16	2.764 ± 0.05	2.795 ± 0.04	2.415 ± 0.0	2.538 ± 0.0
Finland	**3.742 ± 0.04**	**3.613 ± 0.08**	4.126 ± 0.33	4.376 ± 0.18	4.125 ± 0.08	4.068 ± 0.13	3.907 ± 0.0	3.932 ± 0.0
France	327.012 ± 4.1	307.682 ± 1.77	**246.63 ± 10.1**	348.501 ± 23.55	322.253 ± 21.44	318.322 ± 15.58	549.614 ± 0.0	340.719 ± 0.0
Germany	153.397 ± 11.42	124.367 ± 5.83	170.97 ± 39.81	**105.301 ± 15.5**	134.05 ± 19.48	138.829 ± 27.43	205.695 ± 0.0	296.785 ± 0.0
Greece	56.726 ± 1.18	33.947 ± 2.59	**26.908 ± 3.79**	56.176 ± 3.47	55.67 ± 3.92	56.811 ± 4.66	52.971 ± 0.0	78.198 ± 0.0
Hungary	93.659 ± 1.57	86.261 ± 1.97	64.936 ± 3.55	87.224 ± 3.33	97.269 ± 5.26	97.306 ± 3.68	**45.008 ± 0.0**	98.564 ± 0.0
India	167.822 ± 34.65	473.874 ± 55.56	156.634 ± 40.38	250.148 ± 153.11	220.584 ± 150.57	346.538 ± 121.74	76.175 ± 0.0	**72.221 ± 0.0**
Indonesia	**28.45 ± 0.8**	39.192 ± 2.04	91.937 ± 22.59	33.238 ± 4.93	33.08 ± 13.98	64.922 ± 18.61	36.467 ± 0.0	33.889 ± 0.0
Ireland	5.031 ± 0.1	4.107 ± 0.06	**3.941 ± 0.04**	3.952 ± 0.28	4.206 ± 0.26	4.115 ± 0.2	4.683 ± 0.0	4.767 ± 0.0
Israel	11.664 ± 0.57	10.285 ± 0.2	**9.258 ± 0.17**	10.796 ± 0.73	10.798 ± 0.49	10.45 ± 0.47	9.265 ± 0.0	13.352 ± 0.0
Italy	388.742 ± 7.71	377.614 ± 6.23	**120.312 ± 10.63**	337.985 ± 28.13	333.81 ± 26.82	335.793 ± 20.08	180.322 ± 0.0	357.585 ± 0.0
Japan	**13.443 ± 0.61**	15.123 ± 0.91	31.725 ± 6.67	**12.679 ± 2.21**	13.494 ± 1.53	14.486 ± 2.61	16.445 ± 0.0	19.981 ± 0.0
Latvia	6.26 ± 0.14	5.079 ± 0.13	**4.334 ± 0.14**	7.384 ± 0.78	8.406 ± 0.23	8.402 ± 0.22	5.301 ± 0.0	8.057 ± 0.0
Lithuania	13.481 ± 0.16	9.477 ± 0.36	**6.71 ± 0.26**	16.711 ± 1.09	14.668 ± 1.53	12.125 ± 1.12	12.863 ± 0.0	17.772 ± 0.0
Luxembourg	4.742 ± 0.12	4.578 ± 0.06	**3.139 ± 0.04**	5.798 ± 0.25	5.699 ± 0.42	5.776 ± 0.39	3.459 ± 0.0	5.942 ± 0.0
Malaysia	6.255 ± 0.7	**3.003 ± 0.2**	5.212 ± 1.18	6.042 ± 0.79	5.551 ± 1.95	5.008 ± 1.65	3.507 ± 0.0	3.045 ± 0.0
Mexico	272.568 ± 6.94	261.954 ± 6.74	**220.226 ± 5.88**	269.197 ± 19.2	273.764 ± 16.14	257.575 ± 19.03	266.958 ± 0.0	280.753 ± 0.0
Morocco	31.197 ± 1.55	62.771 ± 4.61	45.571 ± 8.02	25.649 ± 4.21	32.99 ± 6.96	29.208 ± 7.99	**22.927 ± 0.0**	23.58 ± 0.0
Netherlands	20.574 ± 0.95	19.314 ± 1.23	**14.642 ± 1.88**	37.3 ± 37.66	33.042 ± 15.65	59.044 ± 33.14	20.926 ± 0.0	22.382 ± 0.0
New Zealand	0.154 ± 0.02	0.164 ± 0.03	0.312 ± 0.13	0.443 ± 0.2	0.532 ± 0.11	0.468 ± 0.27	**0.0 ± 0.0**	0.044 ± 0.0
Norway	5.238 ± 0.02	5.214 ± 0.03	5.083 ± 0.14	5.51 ± 0.16	5.547 ± 0.06	5.574 ± 0.07	**5.023 ± 0.0**	5.388 ± 0.0
Philippines	41.623 ± 2.34	34.37 ± 1.48	25.579 ± 0.95	29.911 ± 5.3	36.293 ± 13.87	31.069 ± 15.76	**24.354 ± 0.0**	24.766 ± 0.0
Poland	306.644 ± 2.97	292.461 ± 5.9	**176.458 ± 10.86**	249.192 ± 12.89	293.259 ± 22.52	258.842 ± 20.09	177.653 ± 0.0	336.678 ± 0.0
Portugal	30.943 ± 2.02	38.56 ± 1.03	27.444 ± 4.15	20.63 ± 2.33	23.273 ± 5.09	24.611 ± 2.47	**12.823 ± 0.0**	58.075 ± 0.0
Romania	62.98 ± 1.48	69.141 ± 2.29	50.042 ± 13.85	48.347 ± 3.75	48.273 ± 4.38	52.004 ± 3.65	**31.517 ± 0.0**	85.367 ± 0.0
Russia	**104.834 ± 16.18**	300.119 ± 41.36	290.375 ± 36.73	119.652 ± 32.09	184.501 ± 30.43	179.461 ± 26.22	105.053 ± 0.0	228.057 ± 0.0
Saudi Arabia	9.033 ± 0.71	**4.237 ± 1.16**	**2.178 ± 3.29**	14.173 ± 5.02	16.871 ± 4.64	27.705 ± 10.69	6.483 ± 0.0	4.576 ± 0.0
Singapore	0.212 ± 0.05	0.184 ± 0.0	0.449 ± 0.14	0.437 ± 0.09	0.404 ± 0.06	0.372 ± 0.22	**0.183 ± 0.0**	0.185 ± 0.0
Slovakia	11.572 ± 0.29	11.699 ± 0.27	28.148 ± 1.18	15.386 ± 2.81	16.646 ± 1.31	18.468 ± 2.21	**11.567 ± 0.0**	14.947 ± 0.0
Slovenia	31.797 ± 0.47	31.162 ± 0.26	26.694 ± 0.78	31.76 ± 0.9	32.593 ± 1.13	32.423 ± 1.5	**25.371 ± 0.0**	36.206 ± 0.0
South Africa	**48.979 ± 2.68**	**40.015 ± 2.25**	64.248 ± 4.18	83.199 ± 8.97	82.81 ± 6.24	76.28 ± 4.3	52.798 ± 0.0	51.02 ± 0.0
South Korea	**2.081 ± 0.12**	30.694 ± 2.55	6.6 ± 1.77	2.582 ± 0.53	2.631 ± 0.32	2.62 ± 0.5	2.456 ± 0.0	2.666 ± 0.0
Spain	**168.079 ± 1.9**	**158.399 ± 5.52**	195.637 ± 42.67	256.755 ± 23.31	202.185 ± 22.46	225.704 ± 26.23	297.479 ± 0.0	206.011 ± 0.0
Sweden	22.898 ± 0.56	22.039 ± 0.57	58.301 ± 1.72	32.616 ± 4.85	32.821 ± 3.0	29.065 ± 4.84	**20.231 ± 0.0**	25.512 ± 0.0
Switzerland	78.199 ± 0.99	75.478 ± 0.96	**45.287 ± 1.61**	79.577 ± 6.88	79.954 ± 5.4	86.021 ± 5.29	68.277 ± 0.0	89.256 ± 0.0
Taiwan	0.055 ± 0.09	1.368 ± 0.21	1.566 ± 0.68	2.203 ± 1.0	2.644 ± 0.57	2.326 ± 1.34	**0.0 ± 0.0**	0.0 ± 0.0
Thailand	0.764 ± 0.44	0.789 ± 0.76	4.584 ± 1.98	9.203 ± 3.22	8.832 ± 1.61	7.284 ± 4.25	**0.0 ± 0.0**	0.069 ± 0.0
Turkey	88.985 ± 2.66	**70.413 ± 8.15**	106.219 ± 6.99	120.463 ± 7.31	105.313 ± 5.26	99.52 ± 6.44	85.707 ± 0.0	98.375 ± 0.0
Ukraine	93.146 ± 4.56	108.103 ± 9.72	99.257 ± 8.93	89.113 ± 4.4	**77.413 ± 4.69**	77.417 ± 3.47	89.445 ± 0.0	115.374 ± 0.0
United Arab Emirates	2.479 ± 0.18	**1.55 ± 0.04**	2.202 ± 0.27	2.218 ± 0.65	2.482 ± 0.93	2.541 ± 0.6	2.601 ± 0.0	1.707 ± 0.0
United Kingdom	201.895 ± 5.31	258.507 ± 12.04	**109.637 ± 14.68**	132.277 ± 8.29	140.575 ± 9.42	207.986 ± 123.94	171.946 ± 0.0	165.891 ± 0.0
United States	1087.88 ± 14.37	1112.445 ± 16.34	**507.74 ± 69.26**	1030.973 ± 45.44	1041.405 ± 80.83	1091.759 ± 87.68	770.861 ± 0.0	1142.181 ± 0.0
Uruguay	1.021 ± 0.11	0.901 ± 0.02	1.229 ± 0.15	0.861 ± 0.14	0.892 ± 0.05	0.911 ± 0.05	**0.816 ± 0.0**	0.987 ± 0.0
Vietnam	0.264 ± 0.3	21.704 ± 2.11	6.349 ± 2.77	8.937 ± 4.04	10.728 ± 2.3	9.438 ± 5.44	**0.0 ± 0.0**	0.027 ± 0.0

Table 8.3: RMSE of the late forecasting scenario. To save space, we abbreviated our methods (M) and the baselines (B). MO is multi-output, RSO is repeated single-output, P is persistence and AW is ARIMA wavelet. We present the data used for features as d for deaths and d+Gm for deaths together with Google mobility. All our methods use the partial refit drift adaptation strategy. Bold results are the lowest for corresponding countries on a 5% significance level.

Country	M: MO:d	M: MO:d+Gm	M:RSO:d	B: LSTM	B: Bi-LSTM	B: GRU	B: Persistence	B: ARIMA wavelet
Austria	**4.422 ± 1.9**	**2.758 ± 0.93**	6.489 ± 14.79	5.516 ± 0.13	5.367 ± 0.11	5.111 ± 0.26	6.499 ± 0.0	5.298 ± 0.0
Belgium	**1.498 ± 4.5**	**1.249 ± 0.56**	4.98 ± 14.22	2.053 ± 0.18	1.839 ± 0.16	2.791 ± 0.53	3.578 ± 0.0	3.011 ± 0.0
Bulgaria	**7.472 ± 2.93**	**5.169 ± 1.5**	9.259 ± 3.26	15.689 ± 2.81	20.819 ± 4.71	10.365 ± 2.31	9.402 ± 0.0	11.425 ± 0.0
Croatia	**3.092 ± 1.22**	**2.413 ± 0.76**	5.442 ± 1.63	3.345 ± 0.08	3.453 ± 0.16	3.633 ± 0.24	5.037 ± 0.0	4.013 ± 0.0
Czechia	**5.202 ± 4.58**	**3.377 ± 1.07**	8.428 ± 8.47	5.609 ± 0.11	5.726 ± 0.15	5.98 ± 0.4	7.87 ± 0.0	34.826 ± 0.0
Denmark	**0.577 ± 0.32**	**0.453 ± 0.13**	2.387 ± 8.36	1.007 ± 0.24	0.953 ± 0.3	1.802 ± 0.69	0.894 ± 0.0	0.68 ± 0.0
Estonia	**0.712 ± 0.63**	**0.354 ± 0.16**	0.965 ± 1.11	1.057 ± 0.32	1.164 ± 0.19	0.878 ± 0.14	1.897 ± 0.0	1.001 ± 0.0
Finland	0.692 ± 0.19	**0.456 ± 0.15**	3.312 ± 6.24	1.066 ± 0.2	1.161 ± 0.48	1.596 ± 0.29	0.577 ± 0.0	0.538 ± 0.0
France	15.311 ± 27.73	**7.8 ± 3.31**	46.287 ± 43.61	14.783 ± 3.07	17.883 ± 5.86	19.97 ± 5.69	34.267 ± 0.0	27.228 ± 0.0
Germany	**38.147 ± 15.76**	**14.051 ± 5.01**	56.778 ± 48.63	71.963 ± 18.43	71.393 ± 25.66	68.289 ± 20.63	54.82 ± 0.0	39.678 ± 0.0
Greece	**3.519 ± 4.39**	**3.18 ± 1.62**	6.398 ± 5.56	9.158 ± 2.65	8.664 ± 3.11	7.523 ± 0.93	5.276 ± 0.0	7.744 ± 0.0
Hungary	**5.616 ± 7.08**	**4.052 ± 1.51**	11.781 ± 5.08	6.704 ± 0.81	6.568 ± 0.68	6.742 ± 0.86	5.837 ± 0.0	6.736 ± 0.0
Ireland	**7.149 ± 1.18**	**4.522 ± 1.68**	9.234 ± 2.16	8.914 ± 0.03	8.924 ± 0.03	9.048 ± 0.1	8.991 ± 0.0	9.076 ± 0.0
Italy	**13.791 ± 31.52**	**9.706 ± 3.6**	21.27 ± 35.81	14.187 ± 4.06	13.793 ± 2.12	15.193 ± 1.22	47.583 ± 0.0	30.964 ± 0.0
Latvia	**2.774 ± 0.53**	**1.898 ± 0.56**	3.347 ± 0.8	4.879 ± 0.64	6.132 ± 0.57	3.769 ± 0.85	3.873 ± 0.0	4.72 ± 0.0
Lithuania	3.587 ± 0.83	**1.224 ± 0.54**	1.866 ± 1.5	7.971 ± 1.13	7.095 ± 0.6	4.751 ± 0.64	4.923 ± 0.0	1.654 ± 0.0
Luxembourg	0.233 ± 0.18	**0.148 ± 0.06**	0.355 ± 0.33	0.415 ± 0.12	0.317 ± 0.23	0.394 ± 0.07	0.183 ± 0.0	0.197 ± 0.0
Netherlands	**2.062 ± 4.78**	**1.493 ± 0.59**	6.791 ± 10.73	9.524 ± 2.12	11.094 ± 2.75	6.822 ± 2.01	3.347 ± 0.0	3.41 ± 0.0
Norway	**0.663 ± 0.37**	**0.43 ± 0.23**	1.629 ± 3.29	0.726 ± 0.08	0.742 ± 0.14	0.932 ± 0.29	3.817 ± 0.0	0.759 ± 0.0
Poland	30.884 ± 12.08	**11.615 ± 4.83**	26.258 ± 20.74	51.821 ± 20.34	39.409 ± 9.1	117.077 ± 22.76	84.46 ± 0.0	23.468 ± 0.0
Portugal	**2.093 ± 0.38**	**1.666 ± 0.55**	4.567 ± 5.75	2.265 ± 0.42	2.314 ± 0.38	2.866 ± 0.26	4.262 ± 0.0	8.485 ± 0.0
Romania	**55.867 ± 10.61**	**36.134 ± 13.75**	91.926 ± 11.95	90.818 ± 3.26	89.342 ± 4.48	94.0 ± 1.68	150.26 ± 0.0	163.263 ± 0.0
Slovakia	4.964 ± 2.33	**1.743 ± 0.62**	3.796 ± 4.49	3.428 ± 0.55	3.164 ± 0.12	3.405 ± 0.35	5.345 ± 0.0	9.655 ± 0.0
Slovenia	**1.501 ± 0.49**	**0.918 ± 0.31**	2.029 ± 1.18	1.978 ± 0.22	1.802 ± 0.2	1.883 ± 0.19	2.041 ± 0.0	2.229 ± 0.0
Spain	**14.79 ± 13.25**	**7.995 ± 2.57**	30.488 ± 36.4	23.37 ± 7.01	40.028 ± 20.67	43.406 ± 9.38	116.214 ± 0.0	22.135 ± 0.0
Sweden	2.526 ± 2.55	**1.034 ± 0.35**	5.757 ± 7.73	3.145 ± 1.43	3.111 ± 0.61	3.383 ± 0.86	2.214 ± 0.0	1.315 ± 0.0

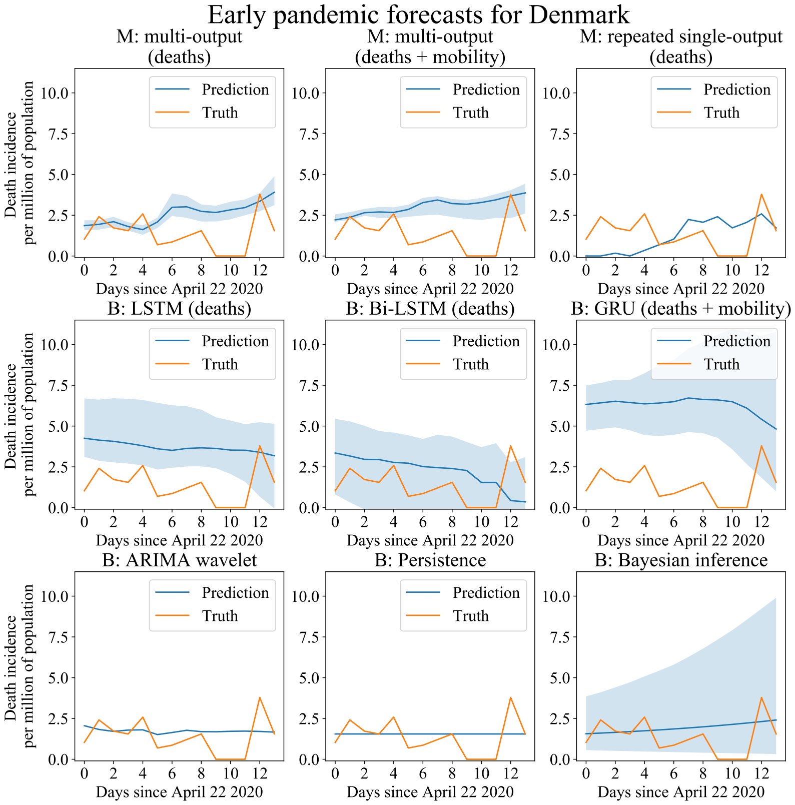


Figure 8.1: Early forecasting for Denmark.

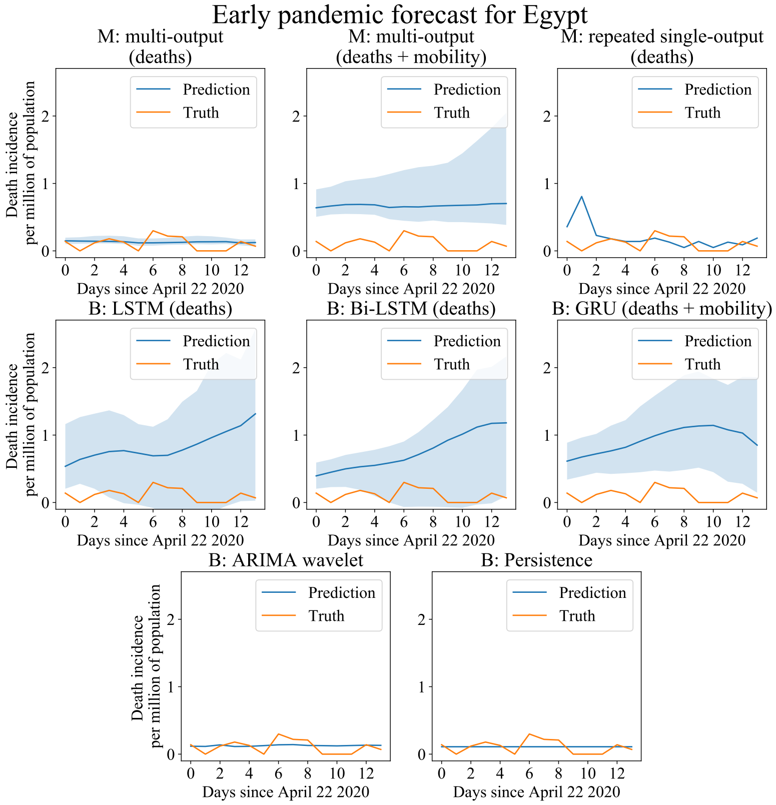


Figure 8.2: Early forecasting for Egypt.

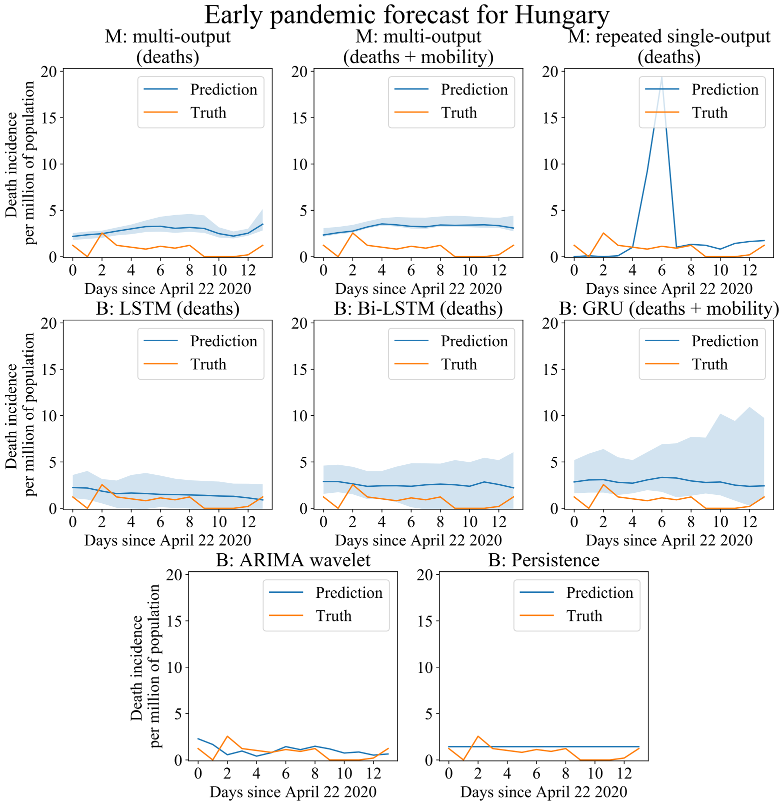


Figure 8.3: Early forecasting for Hungary.

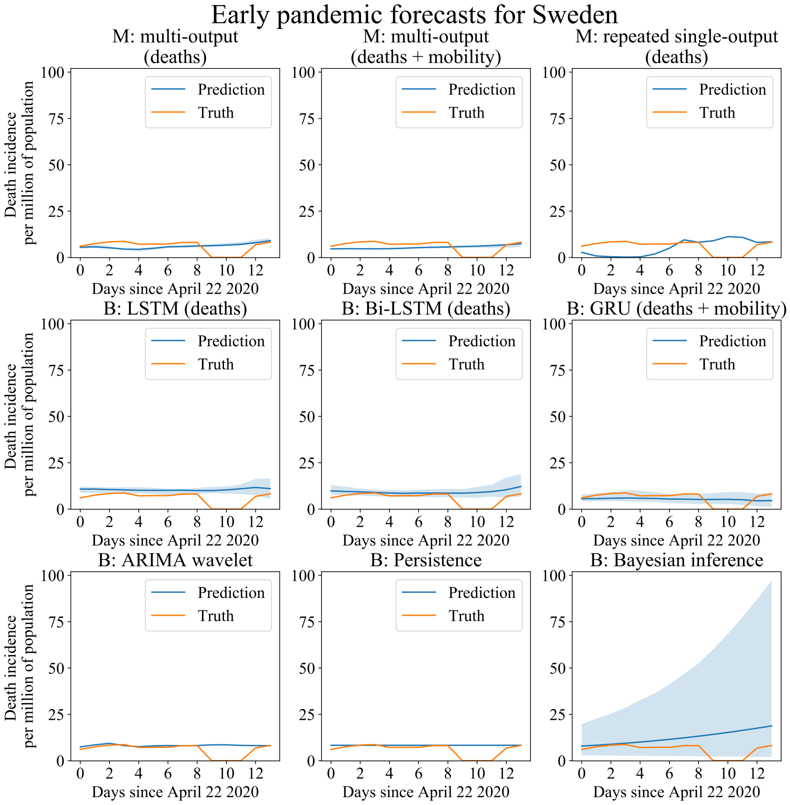


Figure 8.4: Early forecasting for Sweden.

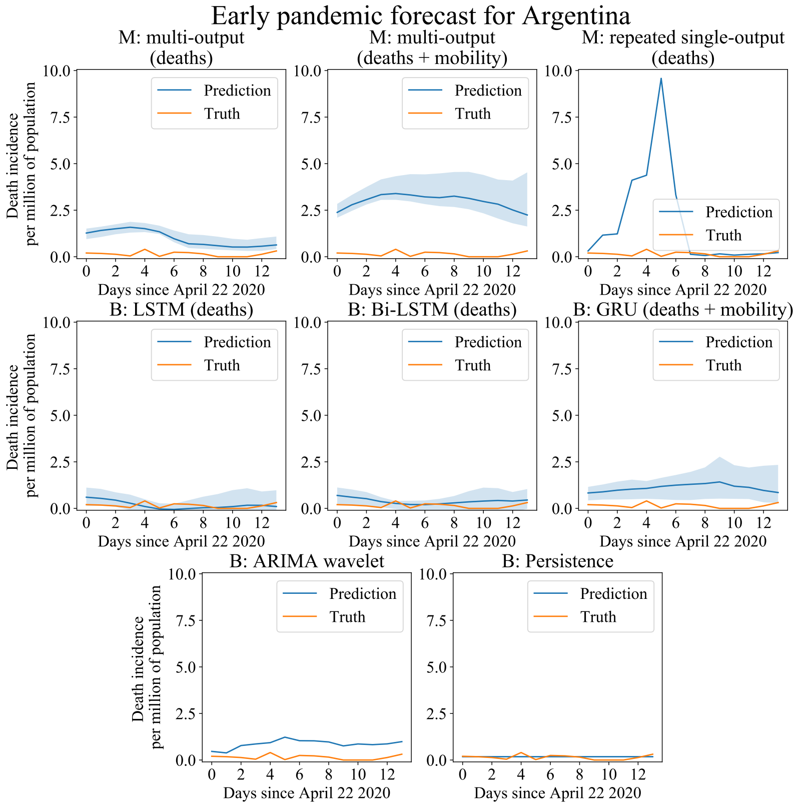


Figure 8.5: Early forecasting for Argentina.

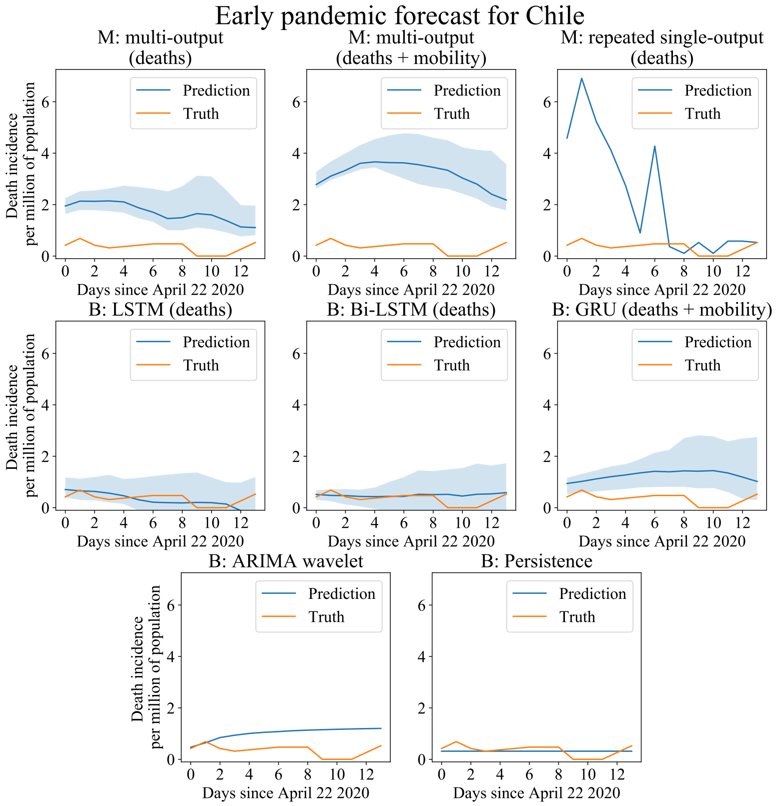


Figure 8.6: Early forecasting for Chile.

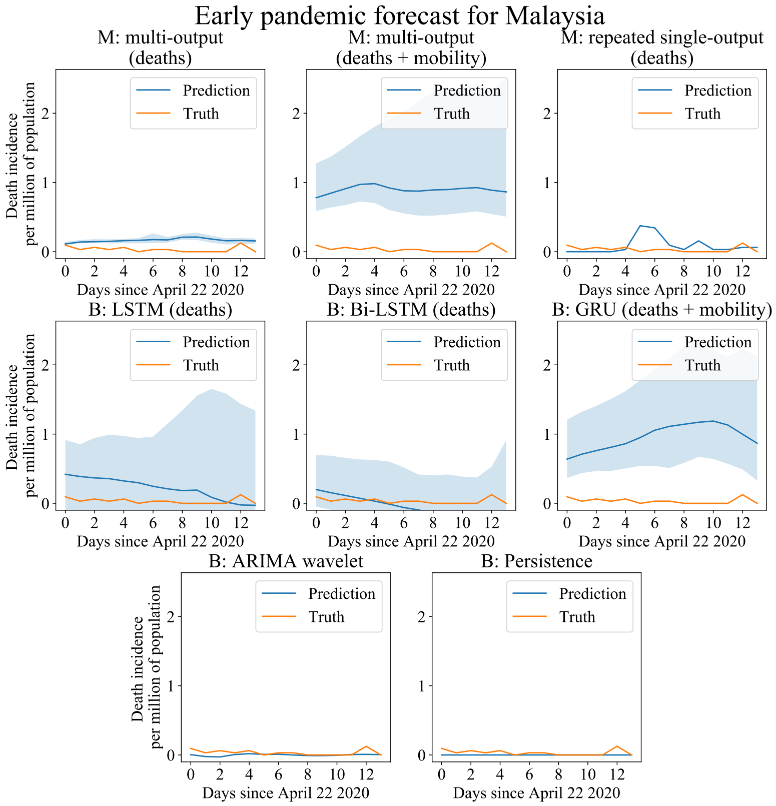


Figure 8.7: Early forecasting for Malaysia.

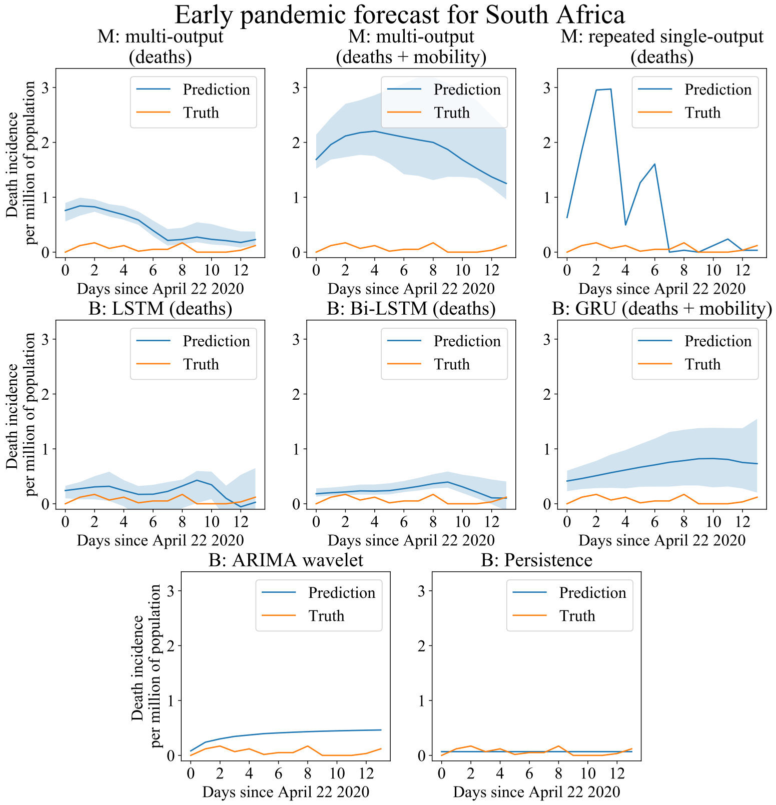


Figure 8.8: Early forecasting for South Africa.

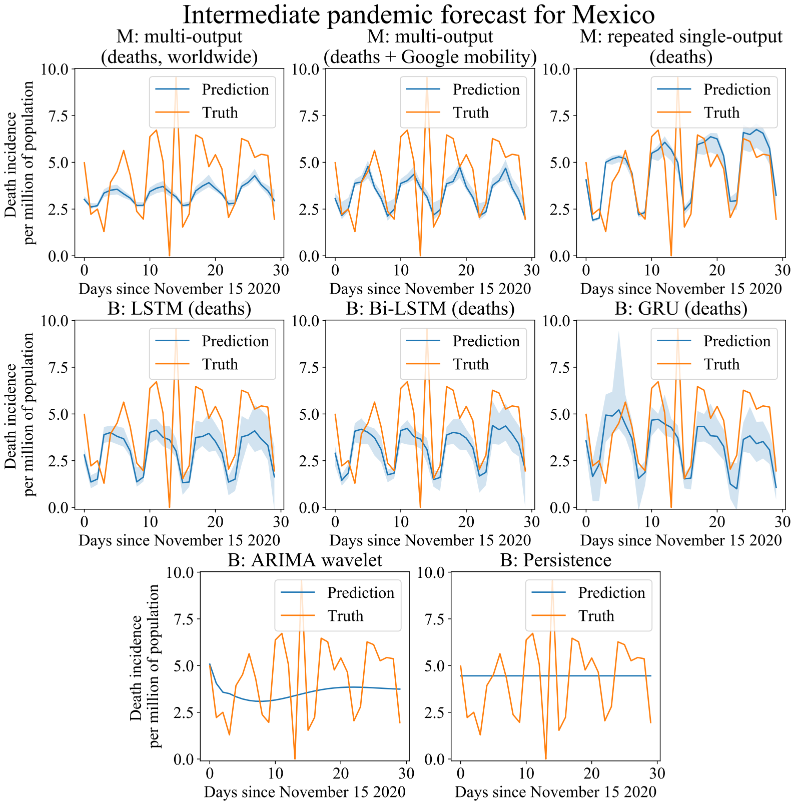


Figure 8.9: Intermediate forecasting for Mexico.

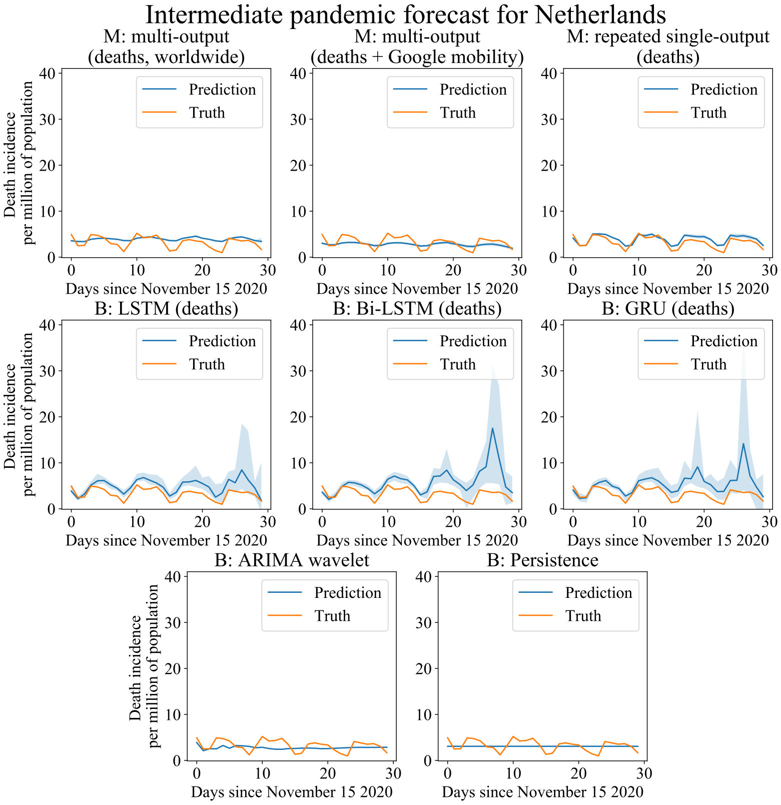


Figure 8.10: Intermediate forecasting for the Netherlands.

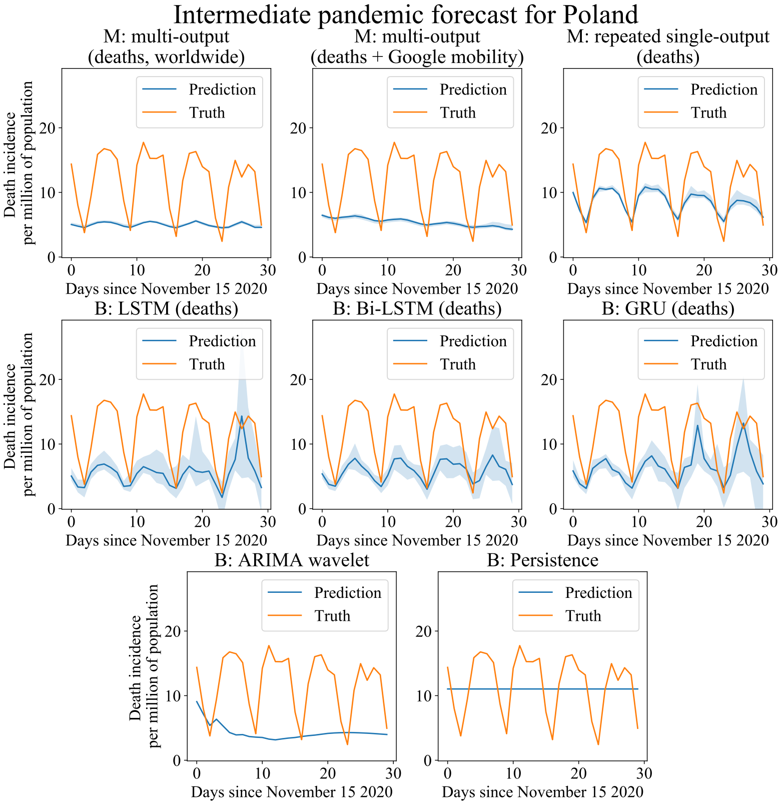


Figure 8.11: Intermediate forecasting for Poland.

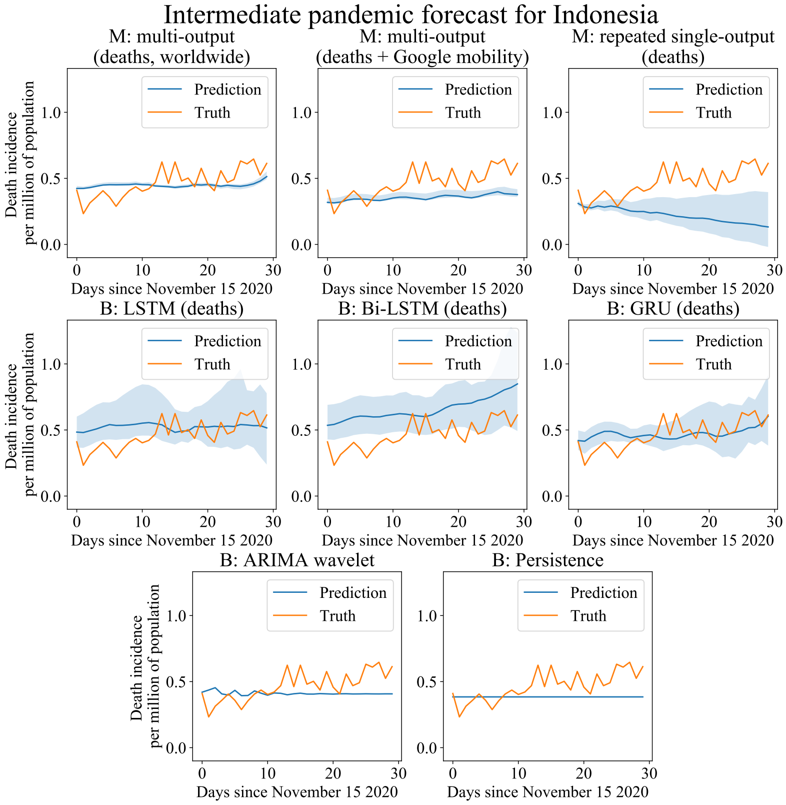


Figure 8.12: Intermediate forecasting for Indonesia.

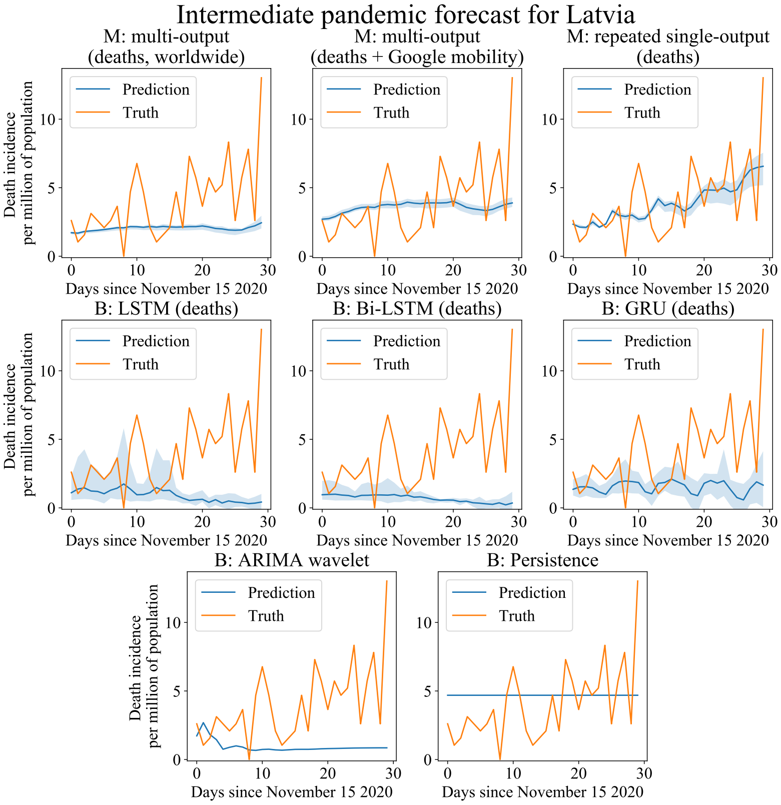


Figure 8.13: Intermediate forecasting for Latvia.

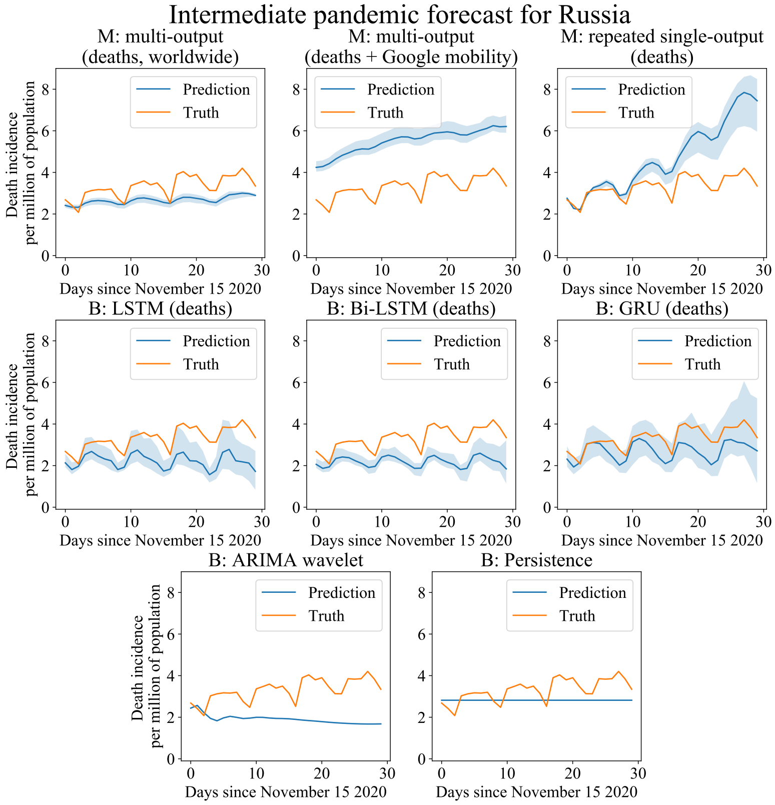


Figure 8.14: Intermediate forecasting for Russia.

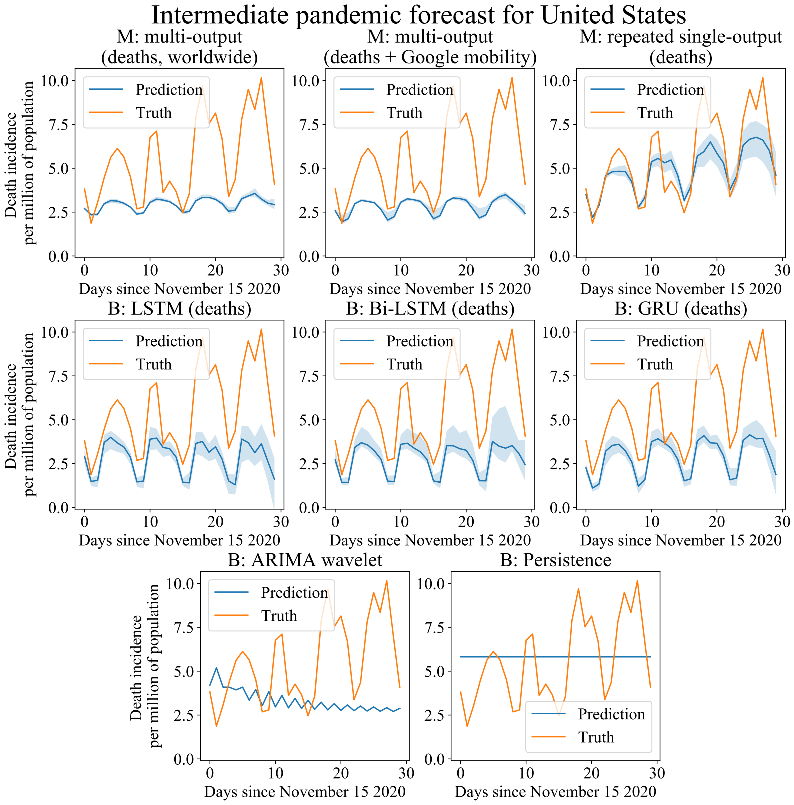


Figure 8.15: Intermediate forecasting for United States of America.

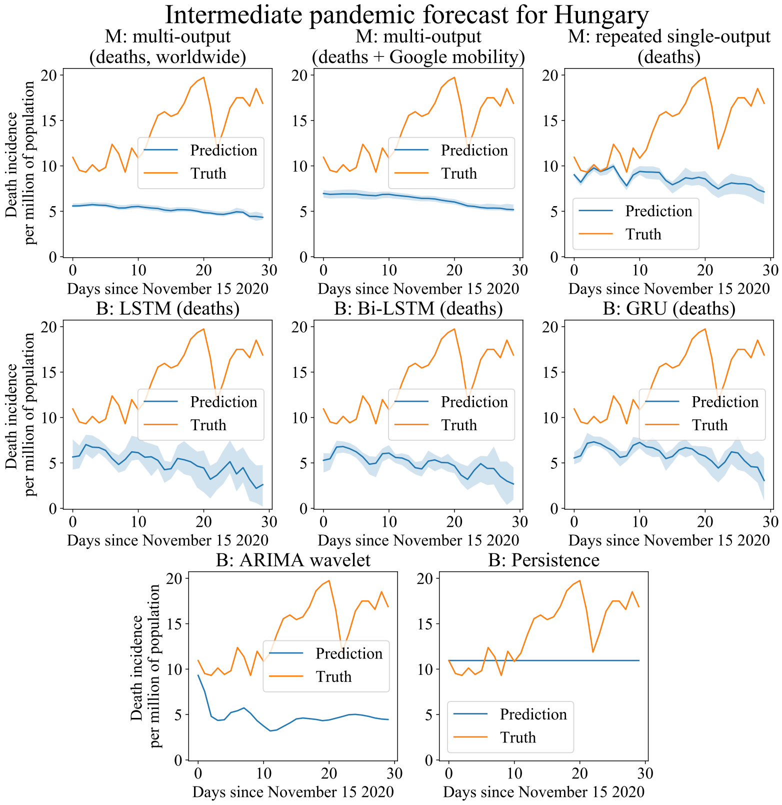


Figure 8.16: Intermediate forecasting for Hungary.

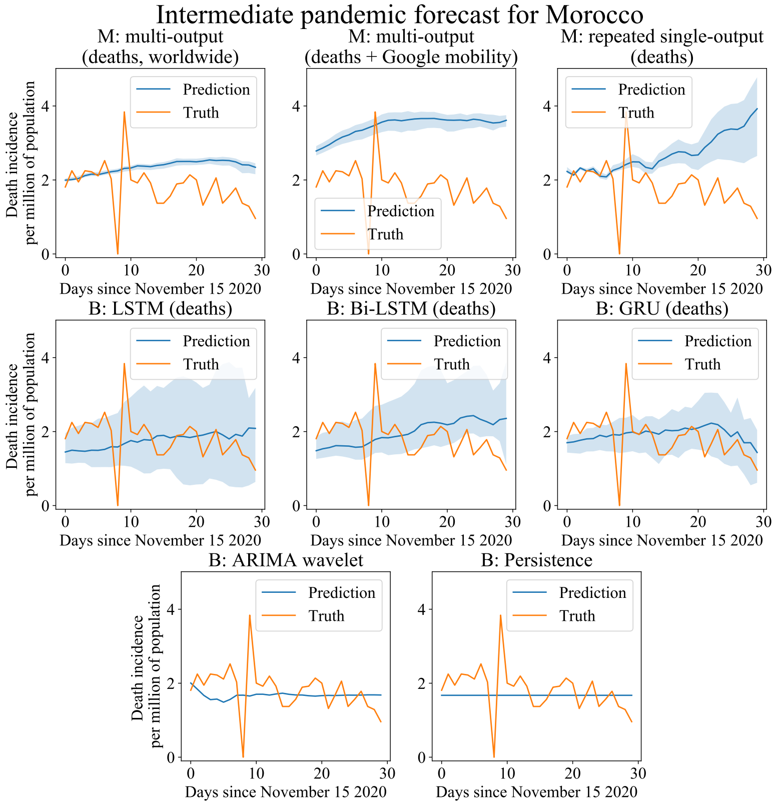


Figure 8.17: Intermediate forecasting for Morocco.

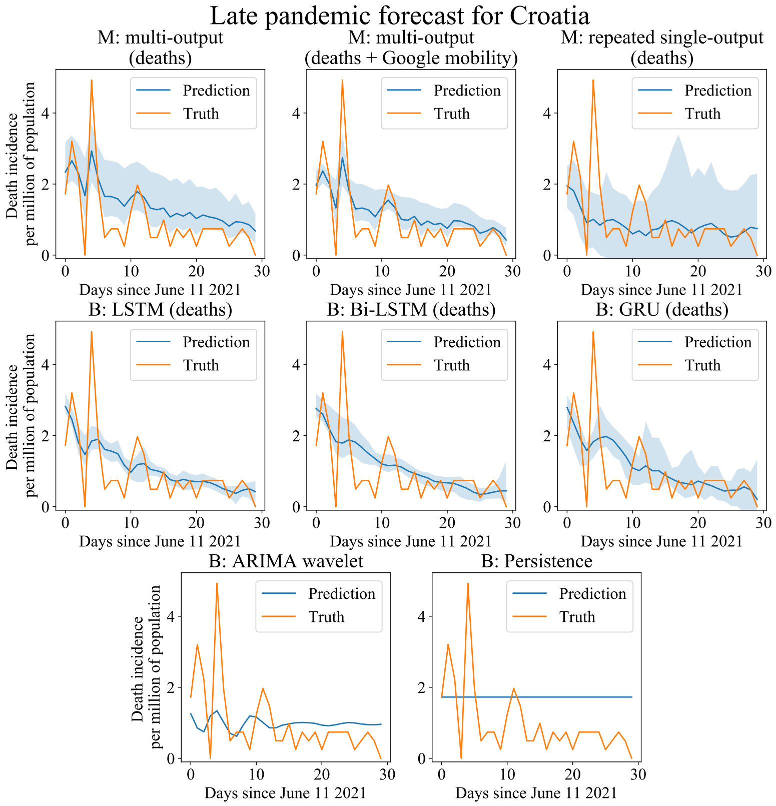


Figure 8.18: Late forecasting for Croatia.

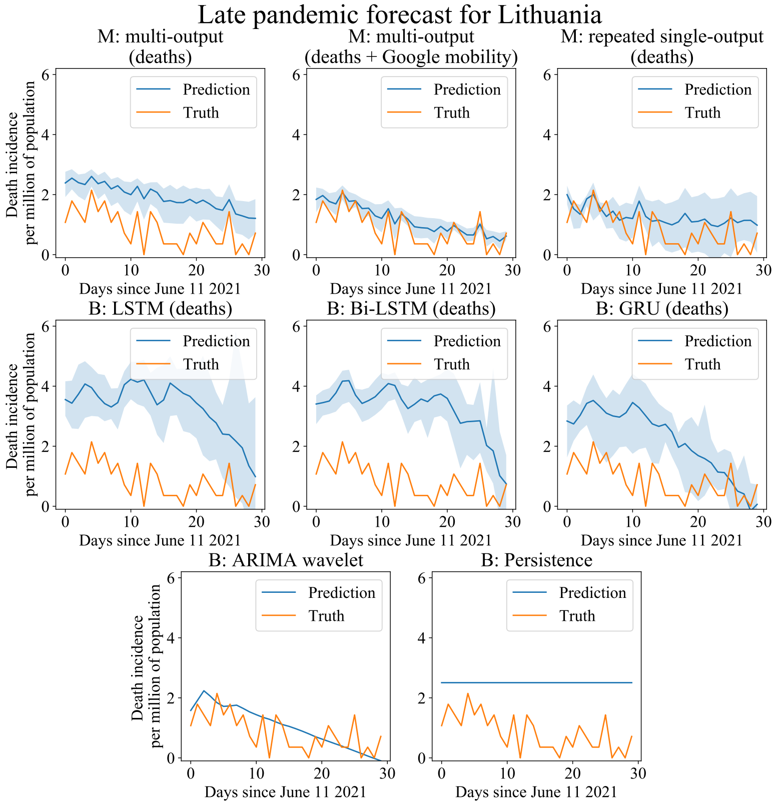


Figure 8.19: Late forecasting for Lithuania.

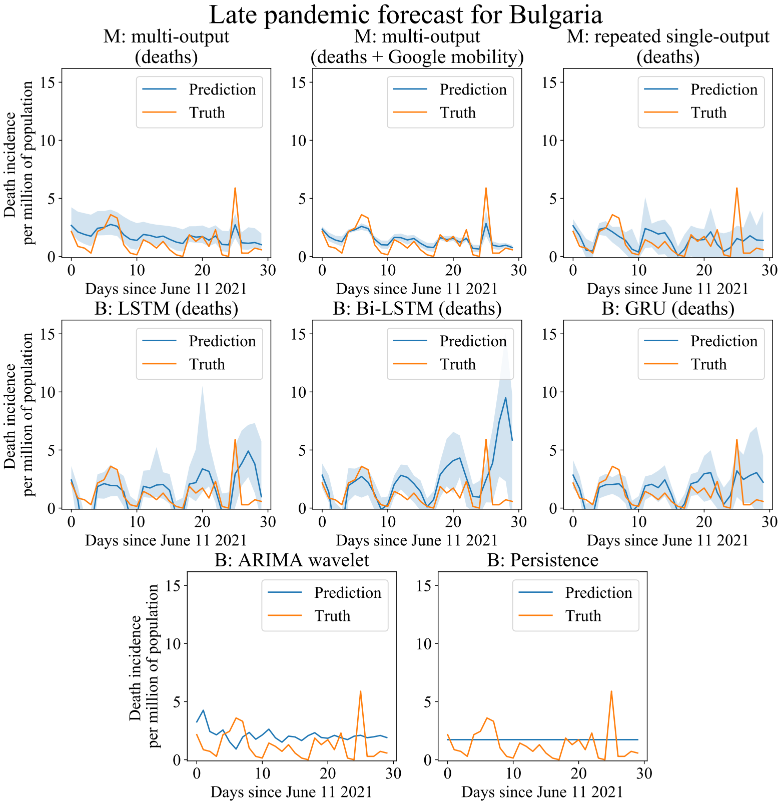


Figure 8.20: Late forecasting for Bulgaria.

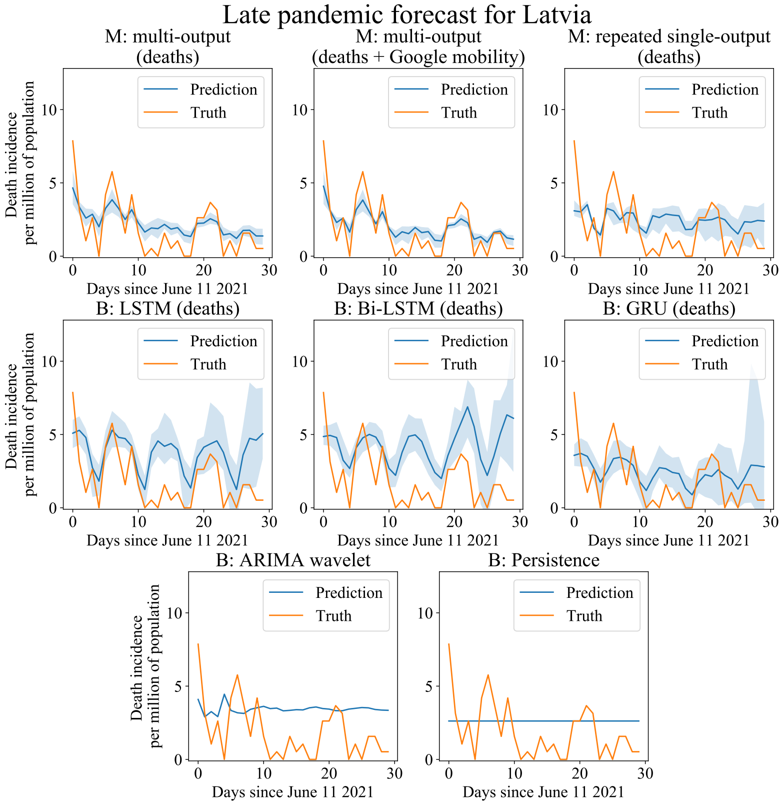


Figure 8.21: Late forecasting for Latvia.

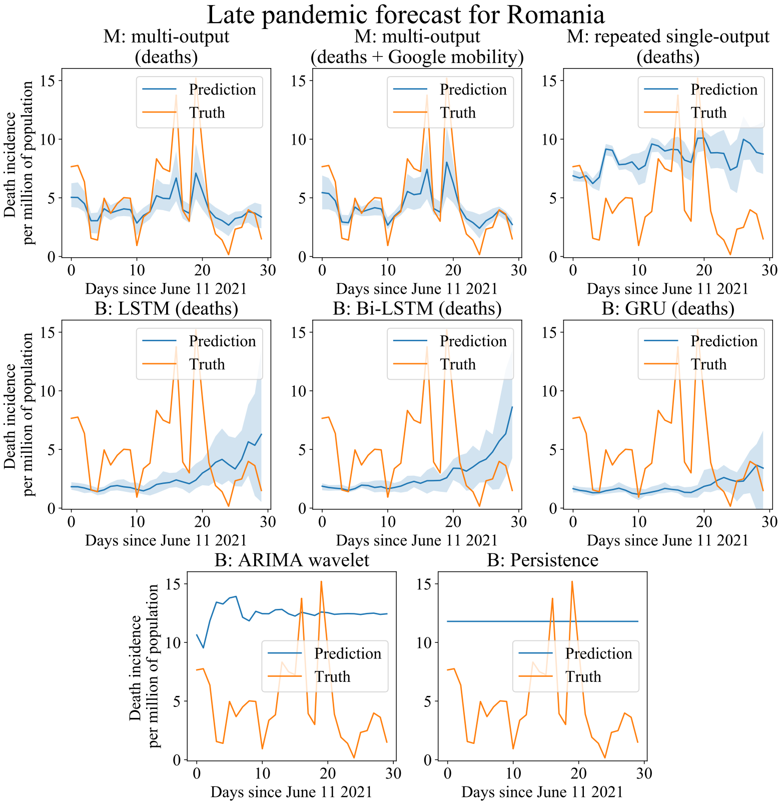


Figure 8.22: Late forecasting for Romania.