A Chronological Review of Deep Learning Models for Seasonal Epidemic Forecasting

Akash Choudhuri

Department of Computer Science,
The University of Iowa,
akash-choudhuri@uiowa.edu

September 2, 2022

1 Introduction

Since the beginning of human evolution, epidemics have been synchronous with growing communities. However, of late, epidemics have emerged as the biggest threat to the human population due to a wide variety of diseases and rapid mutation of disease-causing microorganisms. In addition to this, overpopulation and globalization are the chief catalysts for the current spread of epidemics, making densely populated regions more susceptible to epidemic outbreaks. The recent devastation caused by the COVID-19 pandemic is the most relevant example of the adverse effects of epidemics. It is also a testament that social progress can only be brought about by effectively combating epidemics.

Much research has been conducted on understanding the nature of epidemics and combating them. Amongst various solutions, forecasting epidemics before they occur is a powerful tool to combat epidemic outbreaks. Good forecasts enable effective control measures and help policymakers make informed decisions regarding allocating vaccines, test kits, and other resources to combat epidemics. Also, reliable forecasts can inform and help the general population to adapt their lifestyle patterns according to the future severity of epidemics. On that note, the task of flu forecasting has been studied extensively. Reliable flu forecasts in the USA have influenced various control measures over the past two decades. However, the government still incurs a high cost through yearly flu hospitalizations. Thus, improvements in this domain are essential to minimize overall costs.

Most of the prior works for flu forecasting are classified into statistical, mechanical, and learning-based methods. While statistical approaches like Time Series models [8] and the matrix factorization approach [3] fit a predefined statistical model on historical data and use it for forecasting, mechanical models like compartmental models [16, 20, 22] and the Empirical Bayes model [5] are modelled based on domain knowledge. On the other hand, learning-based approaches are fairly recent, with EpiDeep [1] being the first proposed flu forecasting model in 2019. Despite that, results from learning-based approaches have been promising as they have incorporated both domain knowledge and data ingestion from multiple sources. Also, these deep learning approaches provide interpretable results which can be analyzed to generate deeper insights into the spread of flu.

This report discusses effective learning-based epidemic forecasting methods, associated challenges, and future research directions in this domain. The report describes and compares three papers that built state-of-the-art models for epidemic forecasting [1, 18, 10]. Amongst the papers discussed later, EpiDeep [1], published before COVID-19, was the first work on deep learning for flu forecasting. The poor performance of EpiDeep in the contamination of flu values with COVID-19 signals led to a modified model CALI-NET [18], which does flu forecasting in the context of COVID-19. On the other hand, EpiFNP [10] tries to provide a principled approach for uncertainty quantification of predictions made by Deep Learning models.

The following sections describe the problem statement and associated challenges, the model architectures, the results, and the direction of future works in this domain.

2 Data set and Objectives

To encourage research into making more accurate forecasts about Influenza-like Illnesses (ILI), the US national Centers of Disease Control and Prevention (CDC) has been hosting the 'FluSight' challenge for seasonal influenza forecasting at both national as well as regional levels much before the advent of the COVID-19 pandemic. Models generate weekly predictions of various metrics of the current influenza season based on prior data, which is a representation of a time series data containing values of percentages of weighted Influenza-like Illness (wILI) reported by various healthcare providers all across the USA.

According to CDC, Influenza-like-Illnesses (ILI) is defined as "fever (temperature of 100°F [37.8°C] or greater) and a cough and/ or a sore throat without a known cause other than influenza.". The wILI data is obtained through the reports from ILINet. ILINet consists of a network of hospitals, healthcare facilities and clinics in the USA that reports the proportion of outpatient doctor visits at their facilities where the patient had an influenza-like illness (ILI) to the CDC. The CDC compiles the data, weights the numbers by state populations, and releases one wILI percentage count for each Health and Human Services (HHS) region. In addition, the national wILI count is also released, which is weighted by the national population.

The Flusight challenge generally starts from week 40 of the calendar year and lasts until week 20 of the following year, when influenza is prevalent in human societies. The generalized objectives of this challenge are that, given the wILI data, one needs to build effective models which can effectively predict Future Incidences, Seasonal Peak Intensity, Seasonal Peak Time, and Onset Week.

Specifically, the historical wILI incidence reports are available till week t-1, while the task is to make predictions for weeks t, t+1, t+2, ..., t+k where $k \in \mathbb{N}$. The main prediction tasks are, as follows:

Given: A time series $\mathcal{Y}_c$=\{y_c^1,y_c^2,...,y_c^t,\} representing the current season c till week t and CDC baseline b_c.

Prediction Tasks:

- **Task 1:** Future Incidence Prediction: This is the short term (1-4 weeks) predictions of the model using the historical wILI data. Mathematically, it is given by $\forall_{i=t+1}^{t+4} y_c^i$.

- **Task 2:** Peak Intensity Prediction: It is the maximum intensity of influenza in the given season (highest wILI value for the current season). Mathematically, it is given by: $max_i y_c^i \forall_{i=1}^{T}$, where T is the last week of the season.

- **Task 3:** Peak Time Prediction: It is the surveillance week when the wILI value rounded to one decimal point is the highest. Mathematically, it is given by: $argmax_i\ y_c^i \forall_{i=1}^{T}$, where T is the last week of the season.

- **Task 4:** Onset Prediction: It is the first surveillance week when the percentage of visits for ILI reaches or exceeds a pre-defined baseline value for three consecutive weeks. Mathematically, it is the week j such that $\forall_{i=j}^{j+3}\ y_c^i \geq b_c$.

3 Challenges

The task of flu predictions has different challenges, which makes building models that make accurate predictions very difficult. Amongst the various challenges associated with this task them, some of them are:

- Frequent data revision causes a phenomenon known as backfill [11]. Backfill is caused due to human or technical constraints, which leads to a revision of wILI values over time until they reach a stable value. This poses a significant challenge to the design of the models participating in the FluSight challenge, as they need to account for the uncertainty in revisions of the training data.

- Data sparsity is another major challenge in this task. This is because the data is calculated every week, which means that each year will have 52 data points. Also, data collection started reasonably recently (in the 1997-98 season), which means that constructing deep learning frameworks using sparse data is a significant challenge as traditional deep learning methods mainly give optimal performance with large data sets.

- Due to the similarity between the symptoms of COVID-19 and influenza [21], the need for effective classification of traditional influenza and COVID-19 is paramount. However, due to the current unavailability of enough core data on COVID-19, filtering actual influenza cases from COVID cases becomes even more challenging. So, a practical model must combat data scarcity and contamination to predict future dynamics of the disease effectively. On that note, this paper provides a chronological review of the steps taken to tackle these challenges.

The following papers introduce and solve these challenges to generate effective predictive flu forecasts.

4 EpiDeep: Exploiting Embeddings for Epidemic Forecasting

Epideep was the first deep learning model that leveraged the benefits of deep clustering. It generated time series embeddings for historical seasons, which in turn gave predictions of the spread of disease in the current season. It essentially compares the similarity in characteristics of the current season with the previous seasons to enhance predictive power. Figure 1 shows the schematic representation of EpiDeep.

Prior works before EpiDeep can be broadly classified into statistical, mechanistic, and time-series models. Statistical models used historical data to perform model fitting exercises, which failed to incorporate domain knowledge or were too simple. Some statistical models extensively used feature engineering, which reduced the interpretability of results. On the other hand, mechanistic models incorporated domain knowledge and failed to address additional factors such as human mobility. This led to requirements of extensive calibration to work on varied problems. On the other hand, traditional time series models do not possess the flexibility to deal with outlier phenomena in the data and predict extreme cases. This led to the motivation behind designing a Neural Network based framework. The model also needed to address the above-mentioned issues and work better than the traditional RNN-based sequence prediction models, which generally fail to give optimal performance with sparse data.

The important contributions of EpiDeep are as follows:

- Its architecture is an amalgamation of various deep learning models with appropriate tricks to deal with data sparsity.

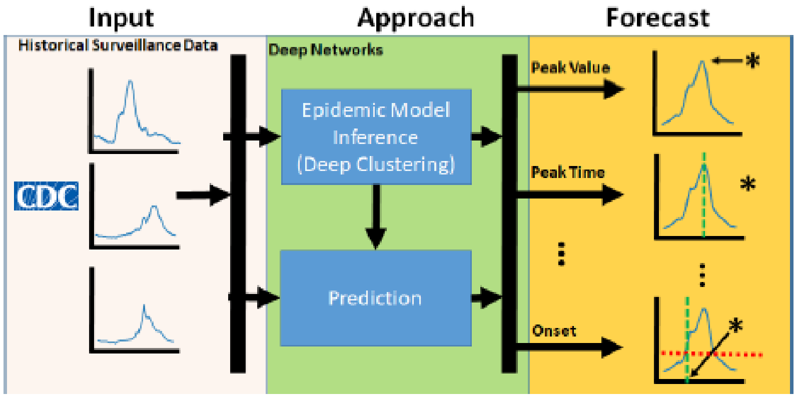

Figure 1: Schematic Representation of EpiDeep.

- Experiments showed that the model was robust to backfill as there was consistent performance when the data was delayed by up to 8 weeks.
- The idea of generating embeddings (explained later) in the latent space helped in enhancing the interpretability of forecasts, which effectively made an overall improvement in the explainability of the model.

4.1 Model Description

As previously mentioned, given the historical wILI time series, the model performs the 4 tasks previously mentioned. This follows by training the model by using historical incidence data Y= $\{\mathcal{Y}_1,\mathcal{Y}_2,\mathcal{Y}_3,\ldots,\mathcal{Y}_{c-1}\}$ and using the trained model for predictions. The idea of 'evolving similarities for prediction' for the current season (which is partially observed) with historical seasons (which is fully observed) is implemented by the 'Query Length Data Clustering' module, which learns embeddings to capture similarities between the current season and historical seasons restricted till time t. Further, embeddings are generated for full-length historical seasons (except the current season) and a mapping function maps between these representations and uses the resultant embeddings for forecasting. This is referred to as the 'Full-Length Data Clustering Module'. The overall model architecture is shown in Figure 2.

We will explain the different components of EpiDeep briefly.

4.1.1 Input Encoding

Given the sequential time series data in the form of $\mathcal{Y}_c$=$\{y_c^1$,y_c^2,...,y_c^t,$\}$, the data is converted to a matrix Y $\in$N$^{l\times(t-l-1)}$ given by:

$$Y = \begin{bmatrix} y_c^1 & y_c^2 & y_c^3 & \cdots & y_c^{t-l+1} \\ y_c^2 & y_c^3 & y_c^4 & \cdots & y_c^{t-l} \\ y_c^3 & y_c^4 & y_c^5 & \cdots & y_c^{t-l-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_c^l & y_c^{l+1} & y_c^{l+2} & \cdots & y_c^t \end{bmatrix}$$

This matrix is fed (column-wise) into a LSTM with the LSTM equations for the j^{th} input are, as follows:

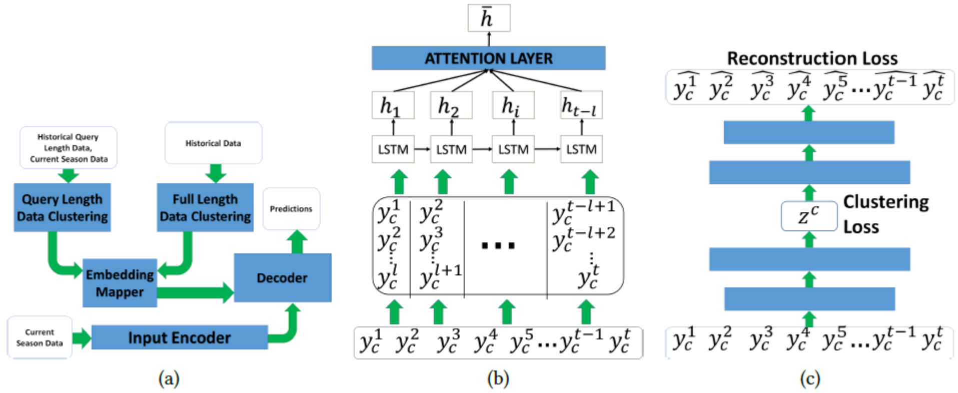
Historical Query Length Data, Current Season Data
Historical Data
Query Length Data Clustering
Full Length Data Clustering
Predictions
Embedding Mapper
Decoder
Current Season Data
Input Encoder
(a)
$\bar{h}$
ATTENTION LAYER
h_1
h_2
h_i
h_{t-l}
LSTM
LSTM
LSTM
LSTM
y_c^1 y_c^2 ⋮ y_c^l
y_c^2 y_c^3 ⋮ y_c^{l+1}
...
y_c^{t-l+1} y_c^{t-l+2} ⋮ y_c^t
y_c^1 y_c^2 y_c^3 y_c^4 y_c^5...y_c^{t-1} y_c^t
(b)
Reconstruction Loss
$\widehat{y_c^1}$ $\widehat{y_c^2}$ $\widehat{y_c^3}$ $\widehat{y_c^4}$ $\widehat{y_c^5}$...$\widehat{y_c^{t-1}}$ $\widehat{y_c^t}$
z^c
Clustering Loss
y_c^1 y_c^2 y_c^3 y_c^4 y_c^5...y_c^{t-1} y_c^t
(c)

Figure 2: (a) Overall Architecture of EpiDeep. (b) The overall architecture of the Encoder Module. (c) The Architecture of the Deep Clustering Module.

$$i_j = \sigma(\boldsymbol{W_i} Y[:j] + U_i h_{i-1} + b_i) \tag{1}$$

$$f_j = \sigma(\boldsymbol{W_f} Y[:j] + U_f h_{i-1} + b_f) \tag{2}$$

$$C_j = i_j \odot \tanh(\boldsymbol{W_c} Y[:j] + U_c h_{i-1} + b_c) + f_j \odot C_{j-1} \tag{3}$$

$$o_j = \sigma(\boldsymbol{W_o} Y[:j] + U_o h_{i-1} + b_o) \tag{4}$$

$$h_j = o_j \odot \tanh(C_j) \tag{5}$$

Here, i represents the input gate while f represents the forget gate, C represents the state and o represents the output gate.

The vector W $\in \mathbb{R}^{h\times 1}$ and matrix U $\in$R$^{h\times h}$ are the weights and h is the size of hidden units.

Further, domain knowledge (dealing with the backfill phenomenon) is incorporated by applying the attention mechanism, which produces the overall output of the LSTM as a weighted sum of the previous hidden states. Mathematically, it can be written as:

$$\alpha_{js}^{a} = \frac{exp(u_{jx}^{T} u_{\alpha})}{\sum_z exp(u_{zx}^{T} u_a)} \tag{6}$$

$$u_{js} = tanh(W_a^T h_{js} + b_a) \tag{7}$$

Then, the context vector vector h_j^{-} for j^{th} input is computed by:

$$h_j^{-} = \sum_z \alpha_{jz}^{a} h_{jz} \tag{8}$$

4.1.2 Deep Clustering

Simply feeding the context vector obtained from the previous encoding phase is not a proper technique as the model will not be able to generalize well. So, there is the need to capture similarities between seasons to give more information to the model to guide its predictions. But due to the sparsity of data, there was a need for a holistic approach to identifying common seasons.

To deal with this, the authors propose the idea of learning an embedding of the partially observed current season in the latent space such that the distance between the embedding and the similar historical season is minimized. This is done in a two-fold step, which are, as follows:

- **Query Length Clustering:** This module tries to cluster the embeddings of historical seasons. Given the current season $\mathcal{Y}_c$={y_c^1,y_c^2,...,y_c^t,}, deep clustering is performed on the union of the current season and the historic seasons (each sequence only taken up to time t) to learn meaningful embeddings of $\mathcal{Y}_c$, which is particularly done using Improved Deep Embedding Clustering (IDEC) [7]. IDEC mainly clusters given input by augmenting clustering loss to reconstruction loss. This leads to the formation of embeddings z_c^t.

- **Full-Length Clustering:** This module deals with whole season-level clustering. The process is similar to Query length clustering, except the clustering is done for entire seasons (T). Thus, for each season $y_i \in T$, full-length embedding z_i^T is obtained.

The partial length embeddings z_c^t are now mapped to the full-length embeddings z_c^T by learning a mapping function f_{emd}, which is implemented by a feed-forward neural network. The following unsupervised loss function is optimized:

$$L_{emd} = \sum_t ||z_i^T - f_{emd}(z_c^t)||_2^2 \tag{9}$$

4.1.3 Predictions

The previously mentioned tasks are computed by the model are as follows:

Task 1: Future Incidence Prediction: This is implemented by using a feed-forward neural network (FFN) f_{next} which maps the encoding learned to the output $\widehat{y} \in \mathbb{R}$. So,

$$\widehat{y} = f_{next}(h_j^-, z_c^T) \tag{10}$$

The loss function is $L_{pred} = \sum_{k \in Y} ||y_k^{t+1} - \widehat{y}||_2^2$.

Task 2: Peak Intensity Prediction: This is done with a FFN as well, except with the training objective being to predict the peak intensity directly.

Task 3: Peak Time Prediction: This metric is also computed using a FFN. But the prediction task is slightly different as the objective is to leverage the encoding to predict peak time (in weeks).

$$x_t = W f_{next}(h_j^-, z_c^T) \tag{11}$$

and

$$P(t|x_t) = \frac{exp(x_t)}{\sum_i exp(x_i)} \tag{12}$$

Here, f_{next} is a FFN, and $P(t|x_t)$ is the probability that the peak occurs at time t. The peak time, with cross-entropy loss function, is given by:

$$\widehat{t} = argmax P(t|x_t) \tag{13}$$

Task 4: Onset Prediction: It is similar to the prediction process of Peak Time Prediction. The overall objective function is:

$$\theta^* = argmin_\theta[L_{emd} + L_C^t + L_r^t + L_C^T + L_r^t + L_{pred}] \tag{14}$$

Here L_C^t and L_r^t are the clustering loss and reconstruction loss of the Query Length Data Clustering Process, and L_C^T and L_r^T are the clustering loss and reconstruction loss of the Full-Length Data Clustering Process. L_{pred} is the prediction loss and is different for each task. The model parameters are inferred using the Adam optimizer.

4.2 Results

The paper used Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE) and Logarithmic Score (for probabilistic predictions) as performance comparison metrics. For the experiments, the baseline models that were chosen are, as follows:

- **Hist:** The traditional approach for flu forecasting, that takes the historical average from previous seasons to make predictions as an average.
- **ARIMA:** This is the Autoregressive Integrated Moving Average Model, which is popular in time series predictions.
- **KNN:** The k-Nearest Neighbours is a popular Supervised Machine Learning Algorithm.
- **LSTM:** This is a traditional Recurrent Neural Network Model called Long Short Term Memory Model.
- **EB:** This is the Empirical Bayes framework which works with the idea of fitting the translation of a historical season to make predictions for the current season.

The results of predictions showed that EpiDeep outperformed all the baseline models in all metrics. Experiments showed that EpiDeep outperformed the LSTM model in all aspects, highlighting the importance of generating embeddings.

Another set of experiments was conducted to observe the effect of delayed data arrival, where forecasts were made using EpiDeep with simulated delays of 2,4 and 8 weeks and observed the predictions for 2014/15, 2015/16, and 2016/17 seasons. The results showed that the model flexibly adapts itself to perform well during the case of delayed data arrival. It can also handle backfill. Further experiments were also conducted regarding the interpretability of the forecasts made by the model, which showed mixed results, one of which was that EpiDeep embeddings have different meaningful clusters of wILI trends for both national and regional, and seasonal embeddings.

5 Cali-NET: Steering a Historical Disease Forecasting Model Under a Pandemic

At the onset of the COVID-19 pandemic, state-of-the-art flu forecasting models were performing very poorly. This was because of the definition of ILI by the CDC. As COVID-19 had symptomatic similarities with influenza, this contaminated the wILI curves during the onset of the COVID-19 pandemic as it was hard to differentiate the two diseases by symptoms alone. Traditional statistical and mechanistic models did not provide good predictions in this case. Epideep, which generally provided much better forecasts than traditional models before COVID, could not perform well during this period. This was due to the vast amount of changes in the wILI seasonal progression (shown in the figure for season 2019-20), which required further domain knowledge about COVID-19 to effectively model dynamics in such a situation.

The important contributions of this work are as follows:

- It uses the idea of heterogeneous transfer learning [15], by which it leverages the benefits of EpiDeep into the current model, which will also work for COVID-19.
- It utilizes a Knowledge Distillation scheme [19] to transfer relevant important information to the target model, reducing the effect of insufficient COVID-ILI Data.

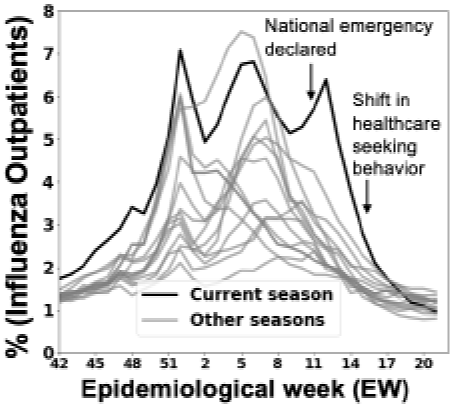

Figure 3: Difference in the wILI curves for a COVID-19 contaminated season and previous historical seasons.

- The authors have performed rigorous experiments to validate the improved performance of CALI-NET over the state-of-the-art models.

5.1 Modified Objective

Here, the given data set is slightly modified, although the original objective remains the same. As previously mentioned, the objective of the flu forecasting task is, given data up to t-1 weeks, the model needs to predict activity for the next k weeks in the future. The provided historical data is the same as in the case of EpiDeep. However, additionally, an external data set containing COVID-19-related exogenous signals for all regions and up to t-1 weeks is provided, which provides additional information to the model regarding the contamination of ILI by COVID-19. The deep learning architecture proposed (elaborated in the latter sections) incorporates the concept of Knowledge distillation [15], which uses EpiDeep as the base model.

5.2 Model Description

The following section will delve into the model architecture and elaborate the different components and attributes of the overall model. Figure 4 shows the overall model architecture.

5.2.1 COVID- Augmented Exogenous Model (CAEM)

As already mentioned earlier, this model works on top of the EpiDeep Architecture to distill COVID contamination from the ILI Data. Information about the COVID cases was found by using an external data set representing COVID-related signals. The simple way to process this data would be to build a simple Feedforward Network. However, as this model was proposed in the early stages of COVID-19, there was a sufficient lack of proper data that could help the FFN perform well. This called for the need to exploit regional interplay characteristics.

The base EpiDeep model is also modified for effective knowledge transfer (elaborated later). While the base design of EpiDeep requires a different model to be trained for each predicted week, the modification (EpiDeep-CN) incrementally re-trains the same model for each week in the season.

Thus, a graph $G = (V, E)$ was constructed with vertices V being the 11 regions (10 regional and 1 national) and E being the set of edges with the condition that an edge will exist between 2 vertices if they border each other. This graph representation is passed into an autoencoder which

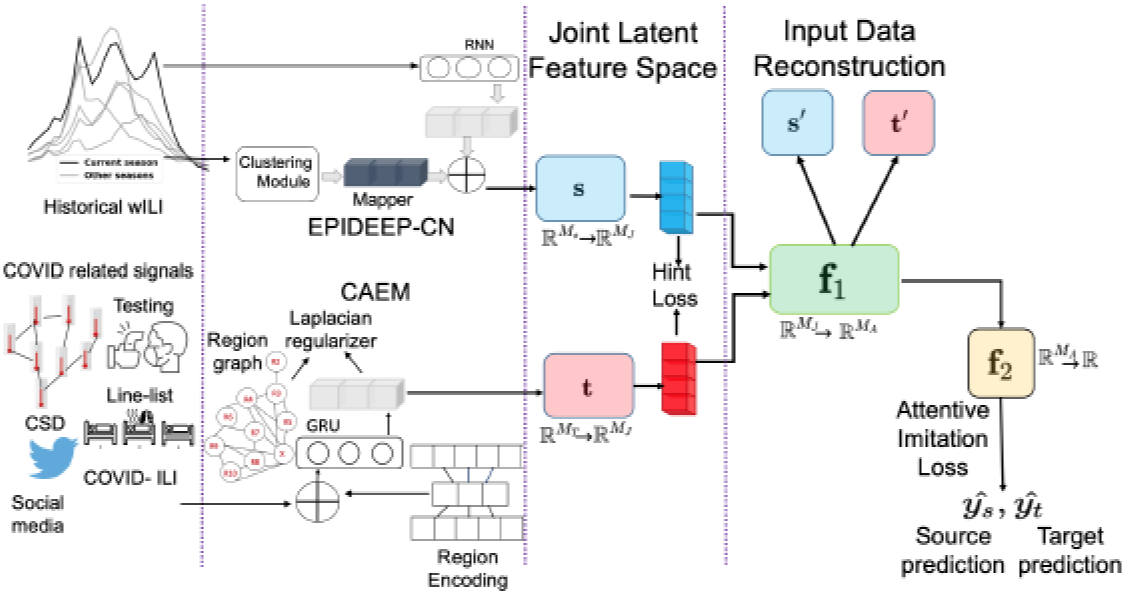

Figure 4: Overall Model architecture. Note that a modified version of EpiDeep is used as the source model and a feature model is designed on top of that.

reconstructs the one hot vectors (columns/ rows of the adjacency matrix) for each region to generate regional embedding $r \in R^{1 \times h_r}$. This is then concatenated with the inputted exogenous data signal of that particular region for that particular week.

The concatenated value is then passed in a sequential manner through the GRU recurrent neural architecture[4]. The GRU is trained to encode temporal dependencies using data from week t-W to t-1 to predict values up to week t+k. Thus, mathematically, if the exogenous data signals input is represented by $x_i^{t-\lambda} \in R^{1 \times l}$ for week t-λ, then the concatenation of $x_i^{t-\lambda}\&r_i$ can be represented as $x_i^{t-\lambda} \in R^{1 \times l+h_r}$, which is the input to the GRU.

The optimization objective for CAEM is:

$$\min_{\Theta_{RE}\Theta_F} ||F(X^{t-W:t-1}; \Theta_F) - Y^t||_2^2 + RE(E; \Theta_{RE}) + Tr(h^T L h) \tag{15}$$

In the equation, Θ_{RE} represents the model parameters for regional embedding, and Θ_F represents the model parameters for the recurrent forecasting function (F). F represents the GRU, whose input, as previously mentioned, is the concatenated historical COVID signals and regional embeddings. The RE or regional auto encoder accepts an 11 x 11 adjacency matrix (E) and h $\in R^1 X h_r$ is the hidden representation of the input sequence generated by the GRU at the end of the recurrence. L represents the normalized Laplacian Matrix of Graph G. L is given by:

$$L = 1 - \mathcal{A} \tag{16}$$

Where $\mathcal{A}$ represents the normalized Adjacency Matrix, which is given by:

$$\mathcal{A} = D^{-1/2} A D^{-1/2} \tag{17}$$

Where A represents the adjacency matrix of G and D represents the diagonal degree matrix of G. Both RE, and F modules are jointly trained coupled with Laplacian Regularization.

5.2.2 Using Learned Representations from EpiDeep

To incorporate the representations of EpiDeep into the current COVID-aware model architecture, the authors incorporate the Heterogenous Transfer Learning with Attention-based Framework proposed in [15] with modified Epideep (EpiDeep-CN) as the source model to the proposed CAEM module that will incorporate the COVID related exogenous signals. However, it is understandable that the outputs of the 2 different modules will not be comparable in their own individual feature spaces. So, the model has modules s and t to transform the embeddings of the EpiDeep module and the CAEM module respectively to a latent space, where the embeddings are comparable. Mathematically, we can consider s and t to be functions such that s: $R^{M_S} \rightarrow R^{M_J}$ and t: $R^{M_T} \rightarrow R^{M_J}$.

After the embeddings have been projected into the latent feature space, a sequence of shared transformations $f_1 : R^{M_J} \rightarrow R^{M_A}$ and $f_2 : R^{M_J} \rightarrow R$ is applied on them, transforming them to the one-dimensional space (thus, the output of f_2 will be the predicted value in the regression problem). On top of this, a denoising autoencoder is applied for input data reconstruction, which improves the latent representations.

5.2.3 Attentive Knowledge Distillation Loss

As the model architecture figure shows, the model performs effective knowledge transfer from EpiDeep-CN. However, it was evident that EpiDeep was not performing well in contaminated ILI data. Thus, Knowledge Transfer would have a control measure on the amount of knowledge EpiDeep (teacher) sends to CAEM (student) to prevent the negative transfer and improve predictions.

The work applied a modification of the attentive knowledge distillation technique proposed previously [19] in deep pose estimation to perform this operation.

The modification of EpiDeep to EpiDeep-CN allows the Knowledge Distillation (KD) loss function to be applied to the same set of EpiDeep-CN parameters, enabling efficient knowledge transfer to CAEM. Notably, the training datasets for the teacher and student models are not the same. However, the KD loss is applied to the overlapped (i.e., from Jan 2020) dataset (when the COVID pandemic started).

The KD loss consists of 2 parts, i.e., imitation loss $\mathcal{L}_{Im}$ and hint loss $\mathcal{L}_{Hint}$. Suppose we define student loss and teacher loss as the error of student and teacher networks concerning the ground truth. Imitation Loss is then the error of the student network concerning the teacher network. However, Epideep-CN (teacher) model's predictions are necessarily not reliable due to a multitude of factors, one of which is the backfill process of ILI data over time. In that case, we would want to model the uncertainty of the teacher model by not a parametric distribution but by using empirical error of the teacher model's prediction concerning the ground truth. Attentive imitation loss amalgamates the loss of both the student model concerning ground truth and the teacher model concerning ground truth and the loss of the student model concerning the teacher model by also modeling the uncertainty of the teacher model's predictions. In hint loss, as used in hint training [19] where the intermediate representation of the student model is trained so that it can mimic the teacher model's latent representation, Saputra et al. [19] argues that the teacher model is not a perfect estimator and may give wrong information to student model at times. So, hint loss considers the output embeddings of student and teacher models, but attentive imitation loss considers the predictions of student and teacher models.

Mathematically, the overall loss function is given by:

$$\mathcal{L}_{KD} = \alpha \frac{1}{n} \sum_{i=1}^{n} \Phi_i ||\hat{y_s} - \hat{y_t}||_i^2 + \Phi_i ||\Psi_s - \Psi_t||_i^2 \tag{18}$$

where $\Phi_i = (1 - \frac{||\hat{y_s} - y||_i^2}{\eta})$ and Ψ_s and Ψ_t are the output embeddings of s and t respectively. $i \in \{1, ..., n\}$ is the index for each training observation and n is the batch size and $\eta = \max e_s - \min e_s$ is the normalizing factor and $e_s = \{||y - \hat{y_s}||_j^2 : j = 1, ..., N\}$ is the actual set of squared errors between the source predictions and the ground truth. N is the total number of overlapping observations in the training data, and Φ_i is the attention weight assigned to the i^{th} training observation. The overall goal of KD is to have an effective and efficient unidirectional knowledge transfer from the teacher (EpiDeep-CN) model to the student (CAEM) model. Because of this unidirectional knowledge transfer process, the KD losses do not affect the teacher model but only the student model (CAEM). The overall architecture is referred to as CALI-NET.

Thus this model provides spatial consistency (through the Laplacian losses) and captures hidden dynamics of interactions between COVID-19 and general ILI, which enables better predictions.

5.3 Results

As this was a novel situation, very few baseline models were available. Also, none of them were specifically focused on COVID-19, which gives this model an advantage. Some of the baseline models were:

- **Epideep** [1]: This has been elaborated before.
- **Delta Density** [2]: This is one of the state of the art performing method in recent CDC influenza forecasting challenges.
- **SARIMA** [17]: This was also a seasonal auto-regressive method.
- **EB:** This has been elaborated on before.
- **Hist:** This has been elaborated on before.

As CALI-NET is more suited to predicting using exogenous COVID-related signals, it is obvious that it will work well. However, the authors focused on the explainability of the model, particularly on the topic of positive knowledge transfer, where they stressed on the importance of preventing negative knowledge transfer. Further, an ablation study was also performed to ascertain the importance of the regional reconstructed graph, Laplacian regularization, and the GRU (recurrent model).

Further experiments are also performed to ascertain the effect of knowledge distillation, particularly the attentive KD loss, after which it was concluded that the loss function provides structure and balance to the transferred knowledge. Also, a study was performed to observe the effect of exogenous COVID-related signals as well as parameter sensitivity experiments.

6 EPIFNP: Neural Non-Parametric Uncertainty Quantification for Epidemic Forecasting

Even though the previously discussed models (EpiDeep and CALI-NET) were significant breakthroughs in Epidemic Forecasting, little prior work has been performed to incorporate the uncertainty quantification in flu predictions. The main contributions of this work are as follows:

- This work introduces a deep generative Gaussian process framework for epidemic forecasting, which automatically learns stochastic correlations between query sequences and historical data sequences (something EpiDeep also did) for non-parametric uncertainty quantification.

- The relations learned for the current season and the previous seasons provide a vast degree of explainability of the model.
- Although recent works have built models like Neural Process (NP) [6] and Functional Neural Process (FNP) [13], they have only been able to incorporate stochastic processes with Deep Neural Networks (DNN) on static data. However, this work provides a DNN framework for dynamic data.
- The rigorous benchmarks and well-calibrated model are vastly superior to the other strong baselines in providing predictions using the evaluation metrics.

6.1 Need for Uncertainty Quantification

The two previously discussed models provide point estimates that might be correct to a degree (but we have no way to measure the uncertainty/ correctness). In contrast, mechanistic models with statistical approaches provide estimates which uncertainty quantities. Another problem with mechanistic models is that they cannot effectively ingest data from multiple sources, something deep learning models can easily do without rigorous feature engineering. However, the accuracy of mechanistic models is inferior compared to deep learning models. So, this work tried to marry Deep Sequential Models with Gaussian Processes to give a level of uncertainty to all its predictions.

On that front, this work models the forecasting task as a probabilistic generative process. It proposes a functional neural process model called EPIFNP, which models the probability density of the forecast value. Such a model is highly desirable as it provides accurate forecasts and a degree of uncertainty, which will help policymakers make informed decisions after getting the model's forecasts.

6.2 Model Description

As previously mentioned, for the current $(N+1)^{th}$ season has snippet of incidence time-series values up to week t denoted by $\mathcal{Y}_{N+1}^{(1,...,t)} = \{y_{N+1}^{1}, ..., y_{N+1}^{t}\}$. Let H denote the incidence data for the past N seasons denoted by H $= \{\mathcal{Y}^{i(1,...,T)}\}_{i=1}^{N}$, where T is the number of weeks per season (generally 52). Here the model not just provides an accurate prediction, it also provides a well-calibrated probability distribution $\hat{p}(Y_{N+1}^{t+1}|y_{N+1}^{1,...,t}, H)$.

In the training phase, the training set M is defined as the set of partial sequences and the forecast ground truths from the historical data H. So, M$=\{(y_i^{1,...,t}, Y_i^t) : i \leq N, t+k \leq T, Y_i^t = y_i^{t+k}\}$. Let R (reference set) be the set of sequences extracted from H that comprehensively represent H. So, R $= \{y_i^{1,...,T}\}_{i=1}^{N_R}$. Let X_D be the union of the training set M and reference set R. The overall architecture of the model is given in Figure 5.

As shown in the figure, the overall model architecture consists of the following components:

- Probabilistic Neural Sequence Encoding: This component encodes the input sequential data into a latent space embedding which helps to model complex temporal patterns within the sequences as well as capture the uncertainty in the sequence embedding.
- Stochastic Correlation Graph: This component captures the correlations between the Reference and Training Set data points in the latent embedding space (seasonal similarity between seasonal curves, something we have already seen important when we saw EpiDeep).
- Final Predictive Distribution Parameterization: This component parameterizes the predictive distribution with 3 stochastic latent variables, namely the global, the local stochastic latent variables, and the stochastic sequence embedding variables.

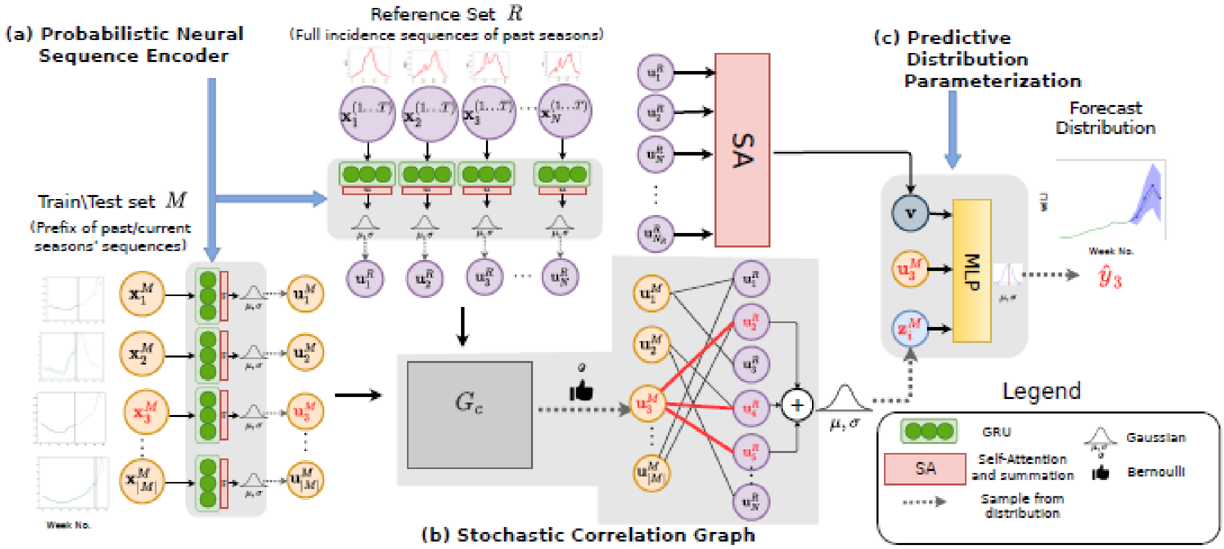
(a) Probabilistic Neural Sequence Encoder
Reference Set R
(Full incidence sequences of past seasons)
(c) Predictive Distribution Parameterization
Forecast Distribution
Train\Test set M
(Prefix of past/current seasons' sequences)
SA
MLP
G_c
μ, σ
$\hat{y}_3$
Week No.
(b) Stochastic Correlation Graph
Legend
GRU
Gaussian
SA
Self-Attention and summation
Bernoulli
Sample from distribution

Figure 5: Schematic Representation of EpiFNP.

The next few subsections will elaborate on each component in more detail.

6.2.1 Probabilistic Neural Sequence Encoder

This component, as mentioned previously, models the complex temporal correlations of the sequence of inputs for accurate overall predictions while capturing the uncertainty in the sequence embedding process. This is achieved by implementing a GRU [4], which acts as a sequence encoder. The latent embedding generated (u_i) is modeled as a Gaussian random variable to capture embedding uncertainty. Mathematically the encoding of the input sequences is done by:

$$\{h_i^1, ..., h_i^t\} = GRU(\{y_i^1, ..., y_i^t\}) \tag{19}$$

h_i^t denotes the hidden state at time step t. As mentioned in the case of EpiDeep, the backfill phenomenon brings about the need for self-attention to construct the embeddings of the input data. So, the attention and aggregation method is the same as EpiDeep:

$$\{\alpha_i^1, ..., \alpha_i^t\} = Self - Attention(\{h_i^1, ..., h_i^t\}) \tag{20}$$

$$h_i^- = \sum_{t'=1}^{t} \alpha_i^{t'} h_i^{t'} \tag{21}$$

So, h_i^- is the summarized hidden state vector. Inspired by Variational Auto-Encoder, each dimension of the latent embedding u_i is parameterized as a Gaussian random variable.

$$p_\theta([u_i]_k | y_i) = \mathcal{N}([g_1(h_i^-)]_k, \exp{[g_2(h_i^-)]_k}) \tag{22}$$

Here g_1 and g_2 are 2 multilayer perceptrons and $[.]_k$ is the k-th dimension of the variable.

6.2.2 Stochastic Data Correlation Graph

This component models correlations between sequences. It constructs a bipartite graph from the reference set R to the training set M based on the similarity between their sequence embeddings. The complete bipartite graph G_c from R to M is constructed by keeping the sequences as nodes. The weight of each edge is the similarity between 2 sequences in the embedding space using the radial basis function kernel [5].

$$\mathcal{K} = (u_i^R, u_j^M) = \exp(-\gamma||u_i^R - u_j^M||^2) \tag{23}$$

u_i^R and u_j^M are the latent embeddings we got from the previous encoder step.

However, the construction of G_C is computationally expensive. So, a sample of G_c is chosen to get a stochastic binary bipartite graph G, which is represented as a binary adjacency matrix where $G_{i,j} = 1$ is when the reference sequence y_i^R is the parent of the training sequence y_j^M. The binary adjacency matrix is then parameterized using Bernoulli distributions:

$$p(G|U_D) = \prod_{i \in R} \prod_{j \in M} Bernoulli(G_{i,j}|\mathcal{K}(u_i^R, u_j^M)) \tag{24}$$

As sampling is done, we can understand that edges in G_c with higher weights will probably remain. Thus, the sampling leads to the creation of sparse correlations for each sampled graph, which speeds up training. This makes the Data Correlation Graph the most important component of the model.

6.2.3 Parameterizing Predictive Distribution

This component captures the functional uncertainty from different perspectives. This is invoked by 3 variables, as mentioned earlier. They are:

- **Local Latent Variable z_i^M:** It summarizes the information of the correlated reference points for each training point and captures the uncertainty of data correlations. This is based on the structure of the data correlation graph shown earlier and each dimension k follows a Gaussian distribution:

$$z_{i,k}^M \sim \mathcal{N}(C_i \sum_{j:G_{j,i}=1} h_1(u_j^R)_k, \exp{(C_i \sum_{j:G_{j,i}=1} h_2(u_j^R)_k)}) \tag{25}$$

  Here h_1 and h_2 are 2 Multi-Layer Perceptrons and $C_i = \sum_j G_{i,j}$ is for normalization.

- **Global Latent Variable v:** It summarizes the information in all the reference points. It summarizes the overall information of the underlying function and thus captures functional uncertainty at a global level. It is computed as:

$$\beta_1, ..., \beta_{N_R} = Self - Attention(u_1^R, ..., u_{N_R}^R) \tag{26}$$

$$v = \sum_{i=1}^{N_R} \beta_i u_i^R \tag{27}$$

- **Sequence embedding u_i^M:** As the 2 prior variables are constructed from the embeddings of the reference sequences, they may lose novel information present in the training sequences. So, a direct path is added from the latent embeddings to the training sequence to the final prediction to allow the architecture to extrapolate beyond the distribution of the reference sequence. This is done by the concatenation of the 2 latent variables and the embeddings u_i, given by:

$$e_i = concat(z_i, v_i, u_i) \tag{28}$$

Then, the final predictive distribution is obtained by:

$$p(Y_i|z_i^M, v, u_i^M) = \mathcal{N}(d_1(e_i), \exp{(d_2(e_i))}) \tag{29}$$

Here, d_1 and d_2 are Multi-Layer Perceptrons.

6.2.4 Learning the Distribution

The overall loss function can be written as:

$$p(Y_M|y_M, R) = \sum_G \int p_\theta(U_D|y_D)p(G|U_D)p_\theta(Z_M|G, U_R)p_\theta(v|U_R)p_\theta(Y_M|U_M, Z_M, v)dU_D dZ_M dv \tag{30}$$

In this equation, directly maximizing the data likelihood is intractable due to the summation and integral in the above equation. Thus, the authors choose amortized variational inference to approximate the true posterior $p(U_D, G, Z_M, v|R, M)$ with a function $q_\phi(U_D, G, Z_M, v|R, M)$ similar to [13] as:

$$q_\phi(U_D, G, Z_M, v|R, M) = p_\theta(U_D|y_D)p(G|U_D)p(v|U_R)q_\phi(Z_M|M) \tag{31}$$

q_ϕ is a single layer of neural network parameterized by ϕ, which outputs the mean and variance of the Gaussian Distribution $q_\phi(Z_M|X_M)$. Adam [12] optimizer is used to maximize the evidence lower bound (ELBO) of the log-likelihood. After canceling the redundant terms, the ELBO is written as:

$$\mathcal{L} = -E_{Z_M, G, U_D, v \sim q_\phi(Z_M|y_M)p_\theta(G, U_D, v|D)}[\log P(Y_M|Z_M, U_M, v) + \log P(Z_M|G, U_R) - q_\phi(Z_M|y_M)] \tag{32}$$

As sampling for the Bernoulli distribution leads to discrete correlated data points, the Gumbel softmax trick [9] is used to make the model differentiable.

During the test phase, the optimal parameter θ_{OPT} is used to base the predictive distribution of a new unseen partial sequence y^* on the reference set as:

$$p(Y^*|R, y^*) = p_{\theta_{OPT}}(U_R, u^*|y_M, y^*)p(a^*|U_R, u^*)p_{\theta_{OPT}}(z^*|a^*, U_R, u^*)p_{\theta_{OPT}}(Y^*|u^*, z^*, v)dU_R dz^* dv \tag{33}$$

Here, a^* is the binary vector that denotes which reference sequences are the parents of the new sequence. u^* and z^* are the new sequence's latent embedding and local latent variable, respectively.

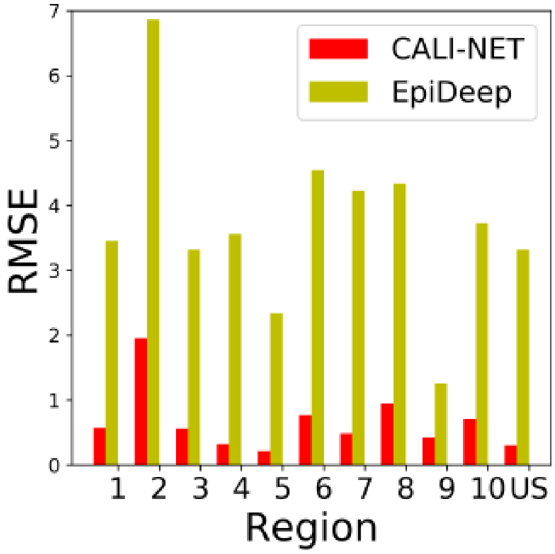

Figure 6: Comparison of results generated by EpiDeep and CALI-NET in period T_1.

6.3 Results

As previously done in the experiments of EpiDeep, the evaluation metrics are RMSE, MAPE and Logarithmic Score. Additionally, a new metric called Calibration Score (CS) is introduced. For a model M, if we define a function $k_M : [0,1] \rightarrow [0,1]$ where for each value of confidence $c \in [0,1]$, if $k_M(c)$ denotes the fraction of ground truth that lies inside the c confidence interval of predicted output distributions of M. For a perfectly calibrated model M^*, we expect $k_{M^*}(c) = c$. CS measures the deviation of k_M from k_{M^*}. Also, the Calibration Plot (CP) is defined to be the profile of $k_M(c)$ vs c for all $c \in [0,1]$. Some of the baseline models are:

- **EpiDeep [1]**: This is the baseline model that has been elaborated on before.
- **Delta Density (DD) [2]**: This is the baseline model that has been elaborated on before.
- **EB**: This is the Empirical Bayes framework which has been elaborated before.
- **Gated Recurrent Unit (GRU)**: This is a popular deep learning sequence encoder that has been used while constructing EpiFNP.
- **Gaussian Process (GP) [23]**: This is a recently proposed statistical method for predicting flu using Gaussian Processes.

The experiments show that EpiFNP outperforms all the baselines in prediction accuracy. Moreover, a further investigation is conducted on the calibration quality, which shows that EpiFNP was the clear winner in both national and regional forecasts by CPs. Furthermore, an ablation study showed that the data correlation graph is the most relevant component in determining uncertainty bounds. Also, EpiFNP seems to adapt and react reliably to novel scenarios, which shows that it can perform well even during the presence of novel patterns.

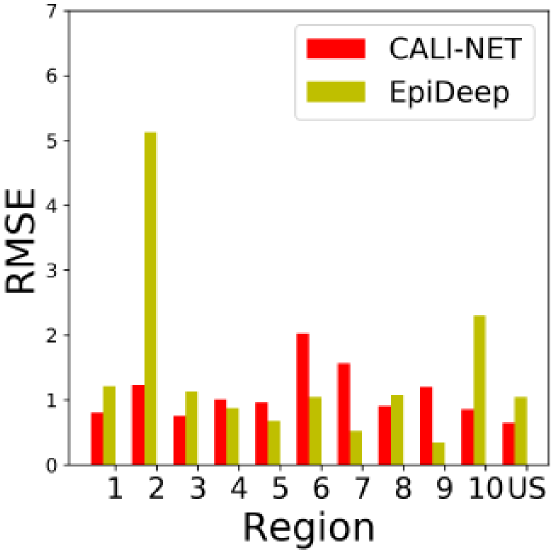

Figure 7: Comparison of results generated by EpiDeep and CALI-NET in period T_2.

Table 1: Comparison of Results of EpiDeep and EpiFNP: k-weeks ahead forecasting for seasons 2014/15-2019/20

Model	Time	RMSE	MAPE	LS	CS
EpiDeep	k=2	0.73	0.14	4.26	0.24
	k=3	1.13	0.23	6.37	0.15
	k=4	1.81	0.33	8.75	0.42
EpiFNP	k=2	0.48	0.089	0.56	0.068
	k=3	0.79	0.128	0.84	0.081
	k=4	0.78	0.123	0.89	0.035

7 Discussion

EpiDeep's novelty lies in that it was one of the first deep learning architectures proposed that provided reliable wILI forecasts, which was sizeable progress from the previously used models for this task. However, the COVID-19 pandemic was unexpected, which required a modification of the vanilla EpiDeep model to adapt to the data contamination during the pandemic. This was highly visible in the results.

If T_1 is the period of the non-seasonal rise of wILI due to contamination by COVID-19-related issues (EWs 9-11), and T_2 is the period when the COVID-ILI trend is declining more in tune with the wILI pattern (EWs 12-15), and T is the entire course (EWs 9-15), then by Comparison of results shows that CALI-NET outperforms EpiDeep significantly.

On the other hand, EpiFNP, which captures the functional uncertainty in predicting ILI values, also outperforms EpiDeep significantly. In terms of measuring how well-calibrated EpiFNP's uncertainty bounds are, it is done using the previously defined metric CS, for which CPs are plotted. In doing that, we observe that EpiFNP's CP is much closer to the diagonal line (ideal calibration) than EpiDeep.

Table 2 shows a comparative analysis of the characteristics of each of the three papers discussed.

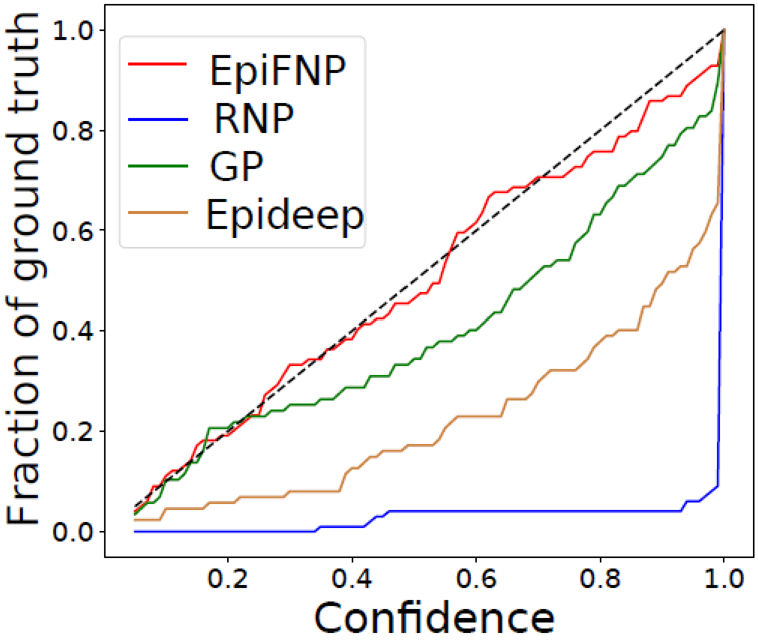

Figure 8: CPs of EpiFNP and next three accurate baselines (one of which is EpiDeep), k=4.

Table 2: Comparison of Results of EpiDeep and EpiFNP: k-weeks ahead forecasting for seasons 2014/15-2019/20

Model	COVID-19	Prediction Type	Significance
EpiDeep [1]	Absent	Point	First Deep Learning Framework
Cali-NET [18]	Present	Point	Effective Prediction in onset of similar disease
EpiFNP [10]	Present	Distribution with Uncertainty	Deep learning with Uncertainty

Also, EpiDeep and EpiFNP extracted similar meaningful insights to examine the seasons that are more likely sampled each week. They observed that the seasons with higher probabilities showed similar patterns to the current test sequence (season to be forecasted). Figure 9 illustrates that.

8 Conclusion

This report introduced three flu forecasting deep learning models and described their components. Also, the chronology of model development was illustrated, which led to the need for the development of each model. Also, this report highlighted the critical objectives and challenges associated with this task as a whole.

EpiDeep was the first deep learning model that was designed to overcome specific challenges in influenza forecasting. Due to its modular neural structure, it provided a platform for neural networks to perform effectively in flu forecasting. Another modification of EpiDeep would be to make a version that directly takes data from epidemiological models. Further, ILI data usually has a geographical structure (e.g., flu incidence in nearby states would be expected to be similar [14]). These types of constraints can also be explicitly codified in the loss functions of the predictor module of EpiDeep. Also, the techniques of using embedded clustering for forecasting can help with

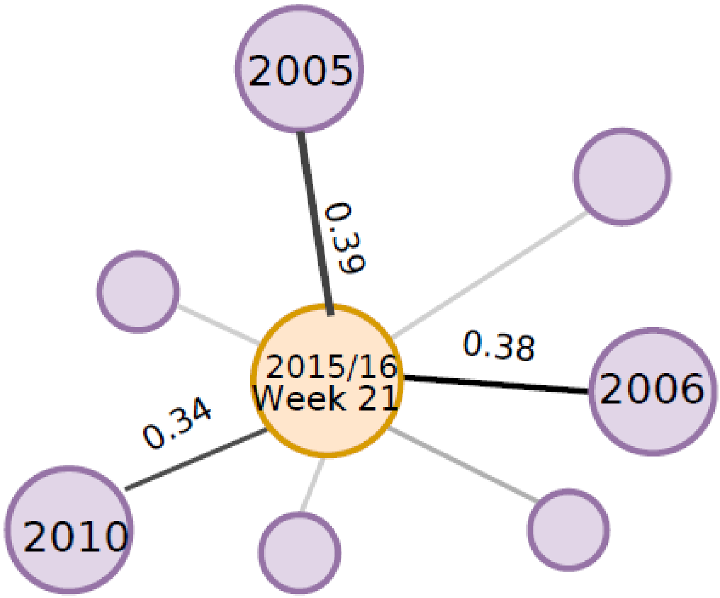

Figure 9: Average edge probabilities for Week 21 of 2015/16 season. We can clearly see that the current season is similar to 2005/06, 2006/07, and 2010/11 seasons.

other sparse time-series data.

Notably, even though EpiFNP seems to be the current state-of-the-art model (SOTA), it can be affected by any systematic biases in data collection. This can be mitigated by creating a reliable synthetic wILI data generator which will also deal with the problem of data sparsity.

A significant conclusion is that each of the three models discussed has a more generalized setup and can be adapted for other general sequence modeling problems. For example, a setup similar to EpiFNP can handle other diseases, and the core technology can be adapted for other general sequence modeling problems. Further, EPIFNP can be extended to use heterogeneous data from multiple sources, similar to Cali-NET. We can also explore incorporating domain knowledge of prior dependencies between different sources/features (e.g. geographically close regions are more likely to have similar disease trends).

On the other hand, CALI-NET effectively captures non-trivial atypical trends in COVID-ILI evolution, whereas other models and baselines do not. These results can guide forecasting models in an generalized emerging disease scenario. In the future, a version of Cali-NET can be applied to other source models (in addition to EPIDEEP-CN), as well as designing more sophisticated architectures for the target CAEM model.

References

[1] Bijaya Adhikari, Xinfeng Xu, Naren Ramakrishnan, and B. Aditya Prakash. Epideep: Exploiting embeddings for epidemic forecasting. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining*, KDD '19, page 577–586, New York, NY, USA, 2019. Association for Computing Machinery.

[2] Logan C. Brooks, David C. Farrow, Sangwon Hyun, Ryan J. Tibshirani, and Roni Rosenfeld. Flexible modeling of epidemics with an empirical bayes framework. *PLOS Computational Biology*, 11(8):1–18, 08 2015.

[3] Prithwish Chakraborty, Pejman Khadivi, Bryan Lewis, Aravindan Mahendiran, Jiangzhuo Chen, Patrick Butler, Elaine O. Nsoesie, Sumiko R. Mekaru, John S. Brownstein, Madhav V. Marathe, and Naren Ramakrishnan. *Forecasting a Moving Target: Ensemble Models for ILI Case Count Predictions*, pages 262–270.

[4] Kyunghyun Cho, Bart Van Merriënboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. corr abs/1406.1078 (2014), 2014.

[5] Paweł Chudzian. Radial basis function kernel optimization for pattern classification. In Robert Burduk, Marek Kurzyński, Michał Woźniak, and Andrzej Żołnierek, editors, *Computer Recognition Systems 4*, pages 99–108, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[6] Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J. Rezende, S. M. Ali Eslami, and Yee Whye Teh. Neural processes. *CoRR*, abs/1807.01622, 2018.

[7] Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. Improved deep embedded clustering with local structure preservation. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 1753–1759. AAAI Press, 2017.

[8] Zhirui He and Hongbing Tao. Epidemiology and arima model of positive-rate of influenza viruses among children in wuhan, china: A nine-year retrospective study. *International Journal of Infectious Diseases*, 74:61–70, 2018.

[9] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax, 2016.

[10] Harshavardhan Kamarthi, Lingkai Kong, Alexander Rodríguez, Chao Zhang, and B. Aditya Prakash. When in doubt: Neural non-parametric uncertainty quantification for epidemic forecasting. *CoRR*, abs/2106.03904, 2021.

[11] Harshavardhan Kamarthi, Alexander Rodríguez, and B. Aditya Prakash. Back2future: Leveraging backfill dynamics for improving real-time predictions in future. In *International Conference on Learning Representations*, 2022.

[12] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.

[13] Christos Louizos, Xiahan Shi, Klamer Schutte, and Max Welling. The functional neural process. *CoRR*, abs/1906.08324, 2019.

[14] Fred S. Lu, Mohammad W. Hattab, Leonardo Clemente, and Mauricio Santillana. Improved state-level influenza activity nowcasting in the united states leveraging internet-based data sources and network approaches via argonet. *bioRxiv*, 2018.

[15] Seungwhan Moon and Jaime G Carbonell. Completely heterogeneous transfer learning with attention-what and what not to transfer. In *IJCAI*, volume 1, pages 1–2, 2017.

[16] Elaine O. Nsoesie, Richard Beckman, Madhav Marathe, and Bryan Lewis. Prediction of an epidemic curve: A supervised classification approach. *Statistical Communications in Infectious Diseases*, 3(1), 2011.

[17] Evan L Ray, Krzysztof Sakrejda, Stephen A Lauer, Michael A Johansson, and Nicholas G Reich. Infectious disease prediction with kernel conditional density estimation. *Statistics in medicine*, 36(30):4908–4929, 2017.

[18] Alexander Rodríguez, Nikhil Muralidhar, Bijaya Adhikari, Anika Tabassum, Naren Ramakrishnan, and B. Aditya Prakash. Steering a historical disease forecasting model under a pandemic: Case of flu and COVID-19. *CoRR*, abs/2009.11407, 2020.

[19] Muhamad Risqi U Saputra, Pedro PB De Gusmao, Yasin Almalioglu, Andrew Markham, and Niki Trigoni. Distilling knowledge from a deep pose regressor network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 263–272, 2019.

[20] Jeffrey Shaman, Edward Goldstein, and Marc Lipsitch. Absolute Humidity and Pandemic Versus Epidemic Influenza. *American Journal of Epidemiology*, 173(2):127–135, 11 2010.

[21] Xiaoyan Song, Meghan Delaney, Rahul K. Shah, Joseph M. Campos, David L. Wessel, and Roberta L. DeBiasi. Comparison of Clinical Features of COVID-19 vs Seasonal Influenza A and B in US Children. *JAMA Network Open*, 3(9):e2020495–e2020495, 09 2020.

[22] Qian Zhang, Nicola Perra, Daniela Perrotta, Michele Tizzoni, Daniela Paolotti, and Alessandro Vespignani. Forecasting seasonal influenza fusing digital indicators and a mechanistic disease model. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, page 311–319, Republic and Canton of Geneva, CHE, 2017. International World Wide Web Conferences Steering Committee.

[23] Christoph Zimmer and Reza Yaesoubi. Influenza forecasting framework based on Gaussian processes. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11671–11679. PMLR, 13–18 Jul 2020.