NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Predicting Flu Outbreaks with RNNs

CE7454 Group 38
Darren Chua
Bill Pung Tuck Weng
Jiehuang Zhang

Outline

1. Problem Identification
2. Data Acquisition
3. Data Exploration
4. Pre-processing
5. Data Analysis with Deep Learning
6. Analysis of Results
7. QnA

Data Problem

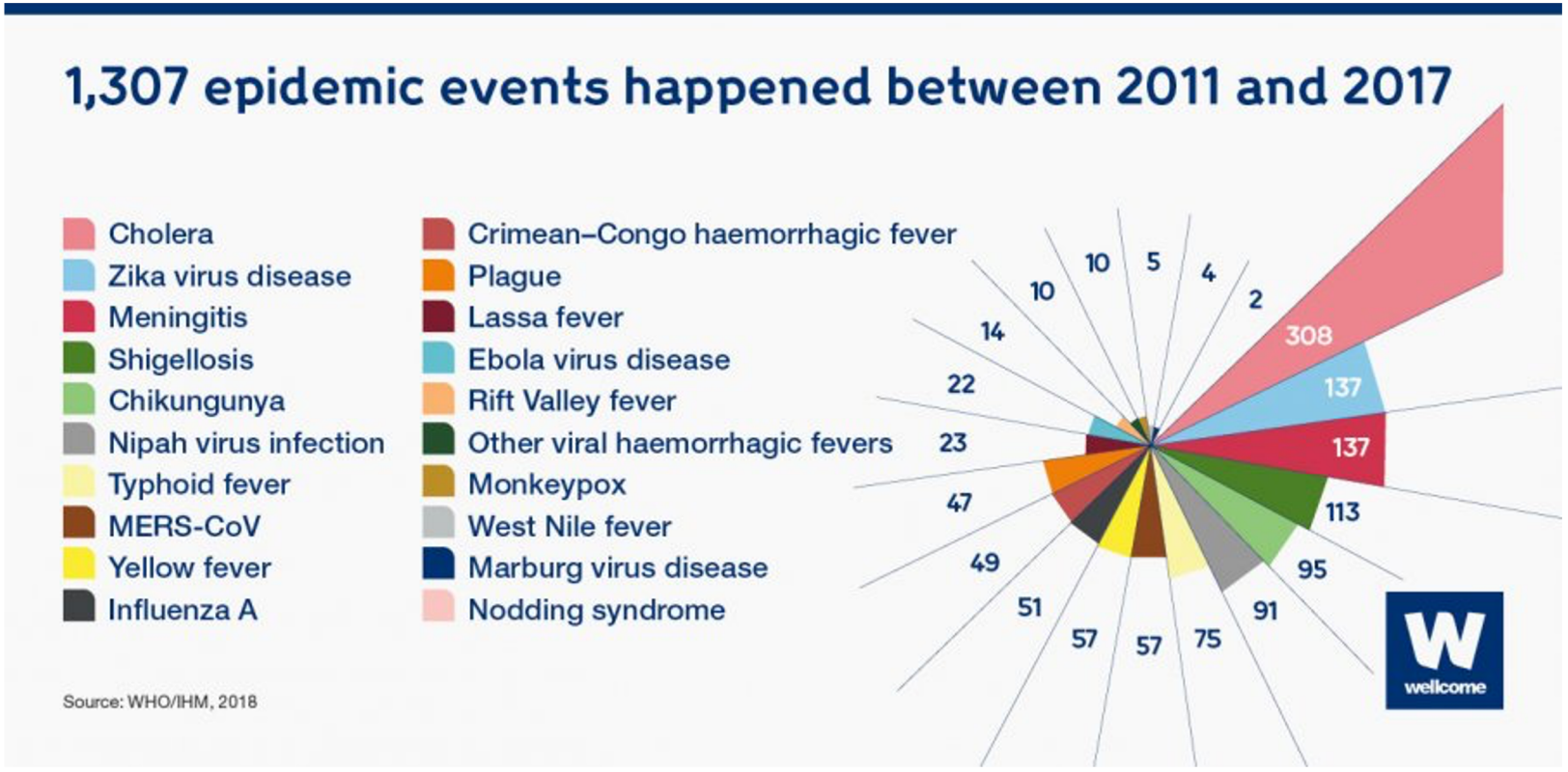
Massive population in developing countries

↓

Difficulty in allocating scarce medical resources

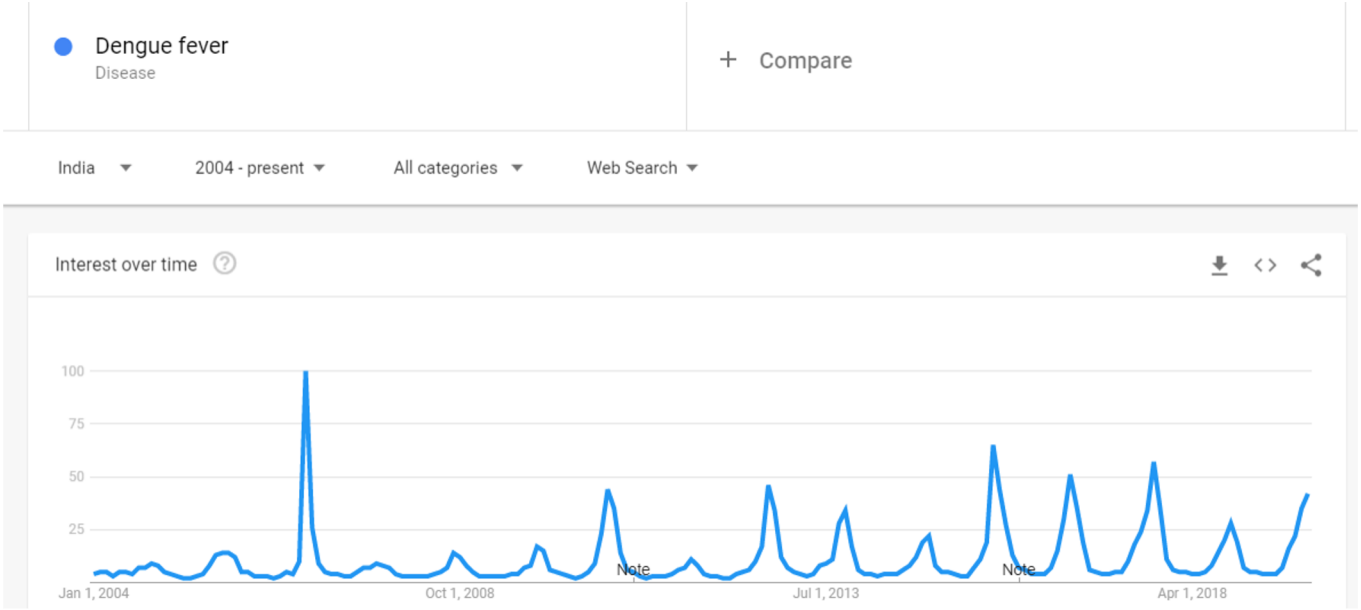
↓

Periodic outbreak of diseases over time



Proposed Approach

1. We built a scraper with an API (Pytrends) for google search trend data on disease keyword search



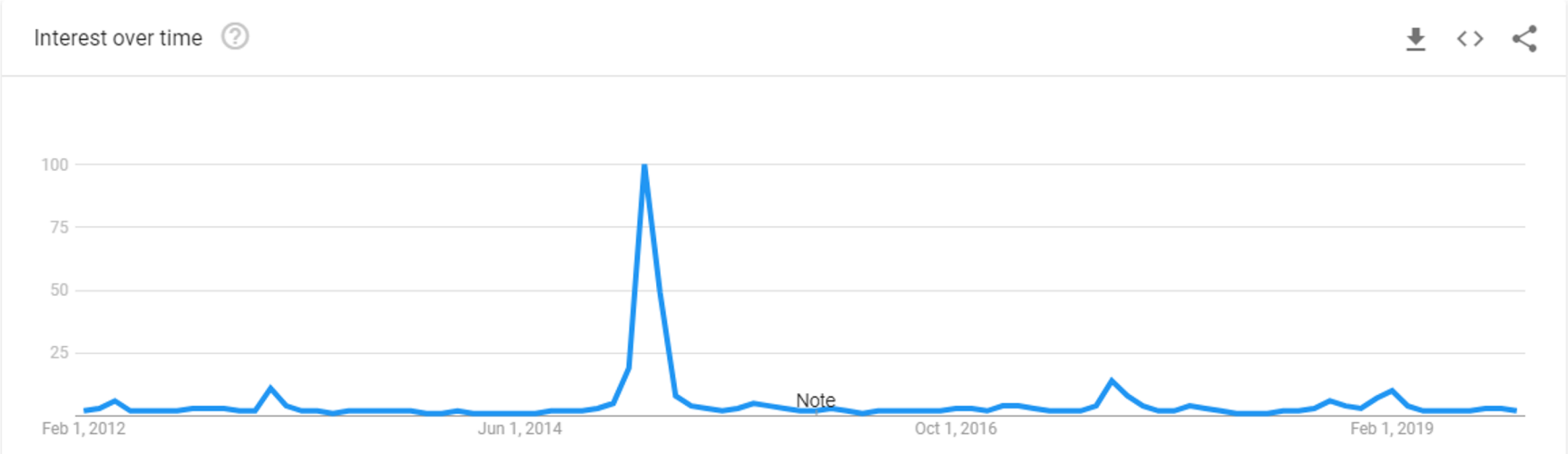
2. Using the data, we train RNN-based sequential models to predict in advance the search trend for each state in India

3. Testing of the model is done by comparing the correlation coefficients with ground truths

Motivation

If our model is accurate, they will enable healthcare providers to better allocate medical resources to combat disease epidemics

Data Acquisition & Exploration



- Keyword for our project: Flu
- Weekly search volume by residents of that state
- Date range : 01/01/2012 – 12/31/2019
- Included selected states of India, 1 csv file for each state
- Order of data is of tenth/hundredth == extremely small
- Issue 1: Values are not absolute but relative.
- Issue 2: Temporal resolutions is dependent on search date length

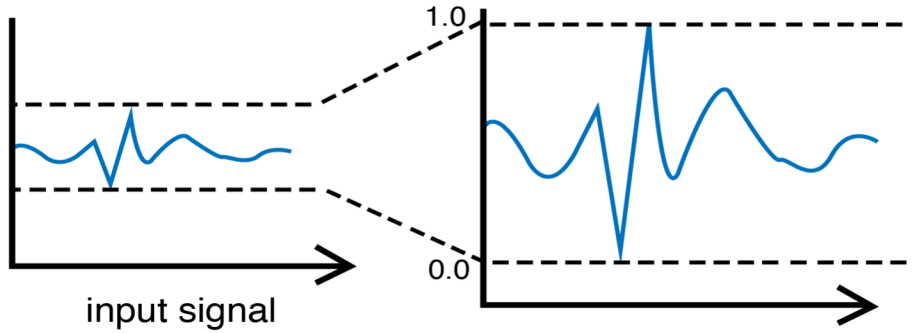
Raw data looks like this

2018-01-14	4
2018-01-21	7
2018-01-28	9
2018-02-04	10
2018-02-11	7
2018-02-18	10
2018-02-25	10
2018-03-04	7
2018-03-11	3
2018-03-18	5

Data Preprocessing

Issue 1: Apply Min-Max scaling to each data instance (X, Y) in train set

```
sc = MinMaxScaler(feature_range = (0, 1))
```



- Based on intuition and some research on disease outbreak, X = 5 weeks of search trend data, Y (true label) = 6th week
- Essentially, we aim to predict the 6th week by a look back of the past 5 weeks
- Why is it ok?
  - -Relative trend with the past weeks is more important
  - -Google's data is not absolute anyway
- For test set, we scale inputs only to prevent "look-ahead bias"

Issue 2: Scrape data per year so that it is of weekly resolutions

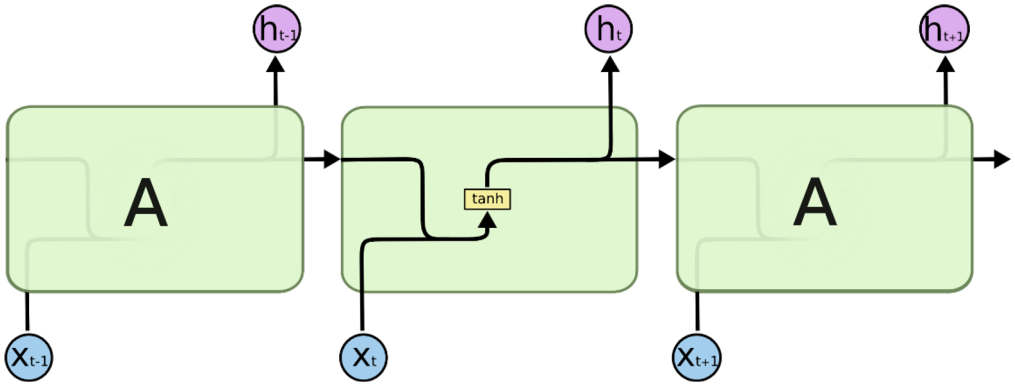
Training

1. Each data instance is obtained by rolling a 6 weeks window forward by 1 week throughout the time-series for each state
2. Hence, the long time-series data is broken up into data instances
3. During training, we randomly sample a batch size of bsz = 64
4. This way it is less likely to over-fit since our data size is extremely small
5. MSE loss is used for our regression problem
6. Empirically, the loss plateaus at around 300 epochs on the average for all the states hence we chose the maximum epoch to be 500 to be safe

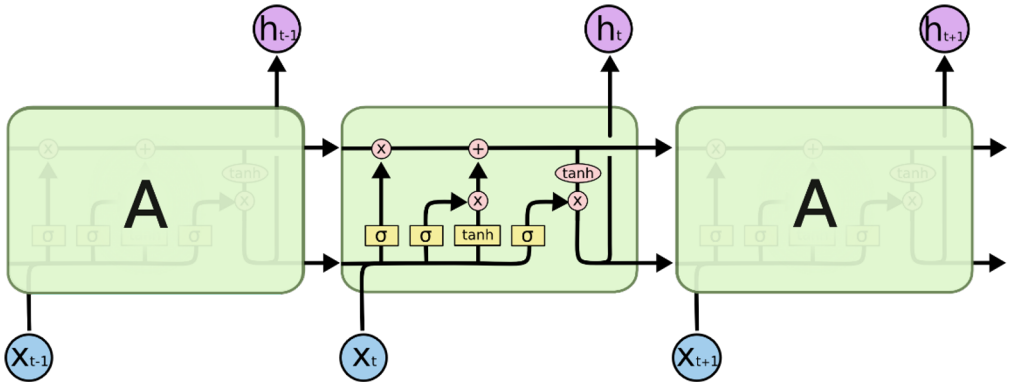
```
indx = np.random.randint(X_train.shape[0], size=bsz)
```

Analysis with Deep Learning

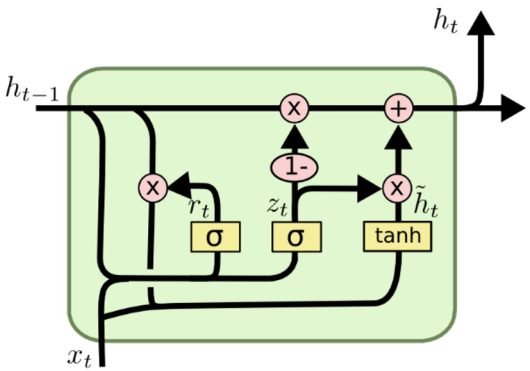
We train our model using 3 different models: RNN, LSTM, GRU



RNN



LTSM



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

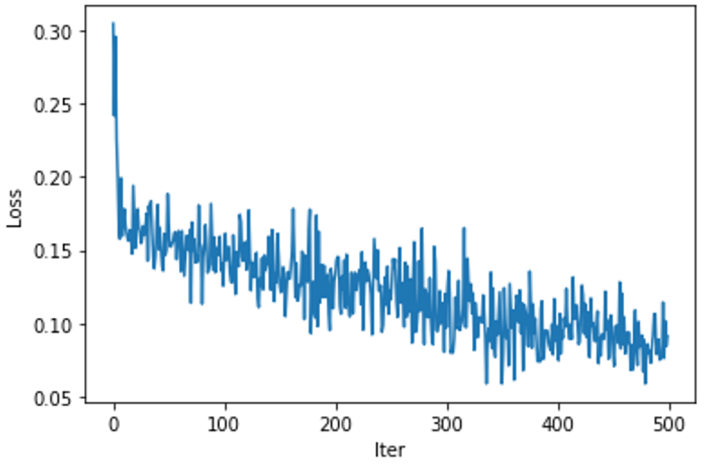
GRU

Source: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Analysis of Results

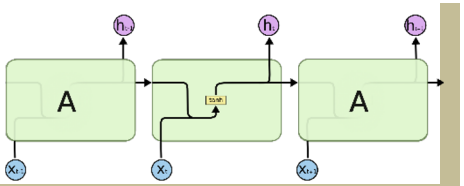
- After 500 epochs of training, we evaluate our model using the test set
- Use spearman & pearson correlation to measure how well the model predicts results of the 6th week (Random initialization: 0.8)
- RNN = GRU = LSTM due to size of dataset

$$pearson = \frac{\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\overline{x})^2(y_i-\overline{y})^2}} \qquad spearman = 1 - \frac{6\sum d_i^2}{n(n^2-1)}$$



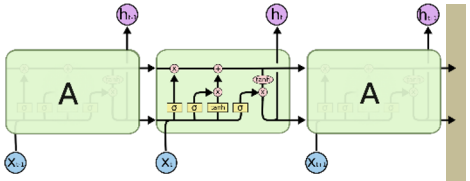
01 RNN

Average Spearman Coefficient: 0.842
Average Pearson Coefficient: 0.850



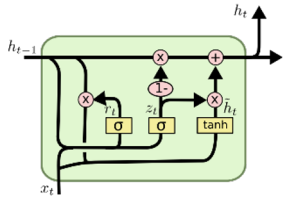
02 LSTM

Average Spearman Coefficient: 0.843
Average Pearson Coefficient: 0.850



03 GRU

Average Spearman Coefficient: 0.842
Average Pearson Coefficient: 0.851



NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Thank You!

QnA