COVID-19 Prediction Through Google Search Trends

Megan Backus and Daniel Morton

CS229 Final Project, Fall 2021

December 6, 2021

1 Introduction

Over the past two years, the public eye has been focused on one key dataset: COVID case numbers. Yet, trends in COVID-19 case numbers have been relatively difficult to predict due to the many dependent factors and the unpredictability of human behavior. While many machine learning approaches have been taken to model and predict these disease trends, we approach this by incorporating Google trends data for pandemic-related search queries, in the hopes that this is a valuable metric for relating human behavior to the prediction of COVID-19 case counts.

In this project, we explore the effectiveness of three model types in predicting COVID-19 trends using search query frequency data: multivariate linear regression, random forest regression, and long short-term memory (LSTM).

2 Related Work

Previous studies have sought to model COVID-19 case count trends and characterize how external factors can influence the case numbers, with a few previous reports looking at the inclusion of search query trends as features in COVID case prediction models. These works, such as [3] and [4], look at the correlation between case count trends and search query trends within the global context and compare these correlation results between countries. In addition, similar attempts to leverage search trends as features for multiple-regression models have been made for non-COVID disease forecasting [7]. Previous case count modeling work utilizing search query trends has been focused primarily on unsupervised learning methods [4], to gain insight about the indicative nature of online search behavior. Nguyen et al. [3] analyzes the impact of adding search query data to the feature set of supervised learning models for COVID prediction on the country scale, concluding that these search trends are generally highly correlated with cases. Similar LSTM models to our implementation have been used on the country-wide case count scale for several other countries such as in [1] and [5]. We expand on this work by also attempting to use Google search query data to model COVID case count trends, but in contrast to this previous work, we consider several different models to forecast temporal trends on the US-state-wide scale.

3 Dataset and Features

We used COVID case data taken from the COVID-19 Data Repository from the Center for Systems Science and Engineering at Johns Hopkins University. From this repository, we extracted time series data of confirmed cases in the US over a 21-month period (February 2020 to October 2021), with a primary focus on the state of California. As the case data is stored in a per-county basis, we extracted the data for all counties within California and took the overall sum.

The Google trends data was then extracted using the pytrends API for 11 search queries we predicted to be strongly correlated with case counts:

covid	covid symptoms	covid cases	virus
coronavirus	coronavirus symptoms	cough	vaccine
COVID-19	coronavirus cases	covid vaccine	

Table 1: Search queries used for the feature set

Google Trends formats its data as a relative search interest, scaled as a value between 0 and 100 depending on the number of searches at a given day and the maximum number of searches within the requested timeframe. This does have some limitations though: for up to 9 months, the data returned is on a daily basis, but any longer than that, the data will be returned on a weekly basis, dramatically reducing the resolution of the trends. Since we sought to fairly compare data across a 21-month timeframe, additional pre-processing of the data was necessary before we could use it in the models: namely, using an overlapping method of reconstructing Google trends data where data for shorter, overlapping periods are concatenated and normalized by the corresponding lower-resolution larger-timeframe data.

4 Methods

For each of the following models, we trained using the daily Google trends data for the aforementioned search terms, and then predicted the daily case count values

over a subsequent test window of time. To reflect the delay in correlation between COVID-related search queries and case count impact, we use a shifted time delay of 6 days, which was experimentally determined as mentioned in the hyperparameter tuning section below.

4.1 Multivariate Linear Regression

As a performance baseline, we trained a multivariate linear regression model using ordinary least squares. This model seeks to minimize the residual sum of squares between the training data and the predicted values, by adjusting a set of weights $\beta_i ... \beta_n$. This model takes the form:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + \epsilon$$

where Y is the predicted output, X_i is the i$^{\text{th}}$ feature and B_i is the associated weight.

4.2 Random Forest

We also trained a random forest regression model to leverage ensemble learning for case count prediction. Random forests (as applied to regression rather than classification problems) work by sampling the dataset and building a series of decision trees, then averaging the outputs from the trees to form a prediction. As compared with standard decision trees, the use of bootstrap sampling is a significant advantage of random forests, as this helps reduce overfitting and improves the overall performance of the model.

For our model parameters, we enabled bootstrap sampling, used the squared error as a measure of split quality, and used a predefined random seed to improve consistency between separate training times. For the sampled trees, we allowed these to use up to the full amount of features in the model (k features per tree $\leq$ 11 total search queries). A total of 1000 trees were included in the forest. Varying 'max_samples' and 'max_features' within the model was found to have small, detrimental effects when compared to using the 'auto' value for each (setting the maximum to be the total number of data points and features, respectively).

A visualization of one of the decision trees used can be seen in Figure 1. Interestingly, when visualizing the impurity-based feature importances within these trees, this model indicated that "covid symptoms" was significantly more important than the other search queries, coming in at 66%. The query "coronavirus" was in second place at 23%, and all others were 3% or less. The importance of "covid symptoms" seems to make sense - a person who may have some initial symptoms appearing may google this a few days before deciding to get a test, with the official positive/negative result coming in a day or two after that. The equation for defining these feature importances is as follows: for importance i_j, feature j, reference score s, repetitions k in $1, ..., K$,

$$i_j = s - \frac{1}{K} \sum_{k=1}^{K} s_{k,j}$$

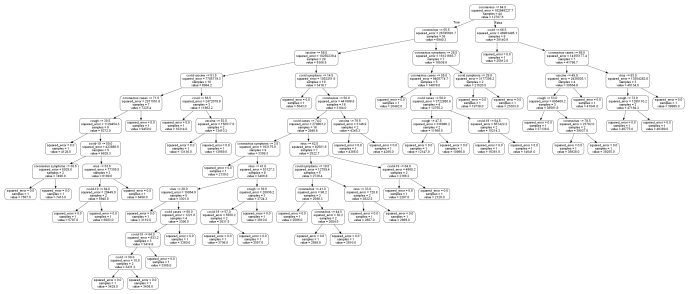

Figure 1: Random Forest tree structure example from initial testing

4.3 LSTM

Finally, we experiment using a long short-term memory model on our time series COVID case data. This model, which is a type of recurrent neural network, utilizes repeating instances of the architecture depicted below.

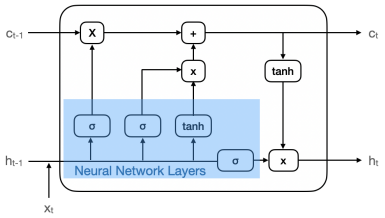

Figure 2: LSTM Architecture

The LSTM model utilizes recurring instances of these blocks, which receives an input sequence and utilizes gates within the block to generate the output. Three gates are present:

1. Forget gate: conditionally chooses block information to discard
2. Input gate: conditionally chooses input values to include in update
3. Output gate: conditionally chooses values to include in block output

These gates are described by the following equations:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i)$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$$

$$h_t = o_t \circ \sigma_h(c_t)$$

where W and U are weight matrices for the input and recurrent connections.

We implement the LSTM model in TensorFlow and train with a hidden layer consisting of 50 neurons, using a batch size of 60, the mean absolute error loss function, and Adam stochastic gradient descent for 50 epochs.

5 Experiments/Results/Discussion

5.1 Hyperparameter Tuning

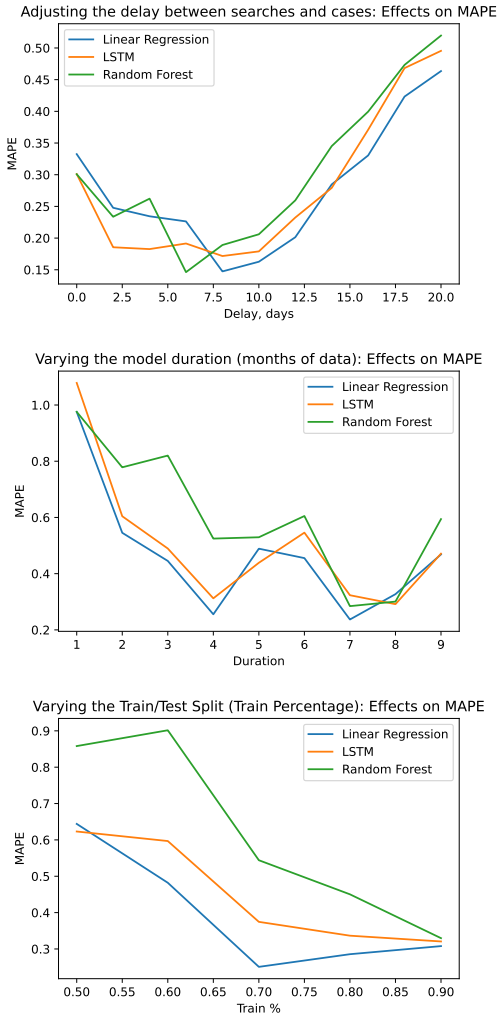

Figure 3: Varying parameters to judge effects on model performance

Key hyperparameters across the models trained include 1) time delay between search trends and case counts, 2) length of time frame, and 3) train/test split percentage. Variations of each of these are shown in Figure 5, plotted with their respective effect on the model accuracy.

From these studies, we determined that the minimum error can be found using a time delay of roughly 6-8 days, along with a time window of approximately 7-8 months and a relatively high train/test split (90% train). The specific "best" values vary slightly depending on the model being considered, but overall, these trends do match our prior hypotheses about the data. The 6-8 day time delay intuitively makes sense with when an infected individual might start noticing preliminary symptoms, then eventually decide to get tested. A time frame of 7-8 months gives a large enough set of data to reduce overfitting, but not so large that the time frame covers significant external factors that influence how COVID spreads (like vaccines). The high train/test split may indicate some overfitting, but this parameter more relates to the predictive duration of our model - which is best at predicting about 2 weeks into the future. If we were to apply this model today, we would only consider the most recent months for training, and would predict trends roughly 2 weeks in advance.

5.2 Feature Importance

We also performed experiments to gain insight into which features have the highest impact on model performance. The results of a feature ablation study on the LSTM model is depicted in Figure 4, revealing a high importance for queries such as "coronavirus symptoms" and "covid", but quite low for "cough."

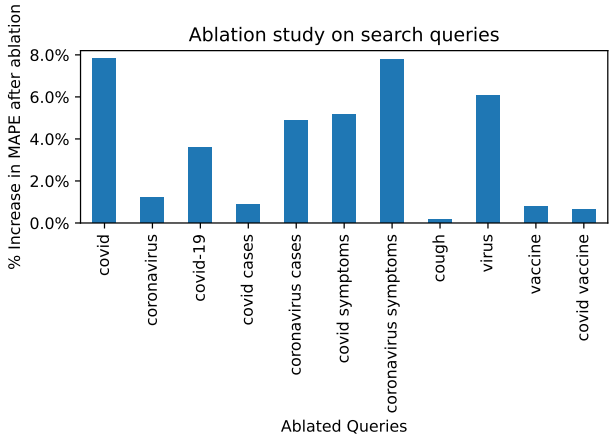

Figure 4: Ablation study on search queries in LSTM

5.3 Model Performance and Comparison

A performance comparison across the three models can be seen in Figure 5 for a sample four-month period. As can be seen from the prediction plots, the LSTM model outperforms the baseline model and the random forest model in its ability to predict COVID case numbers over the test window.

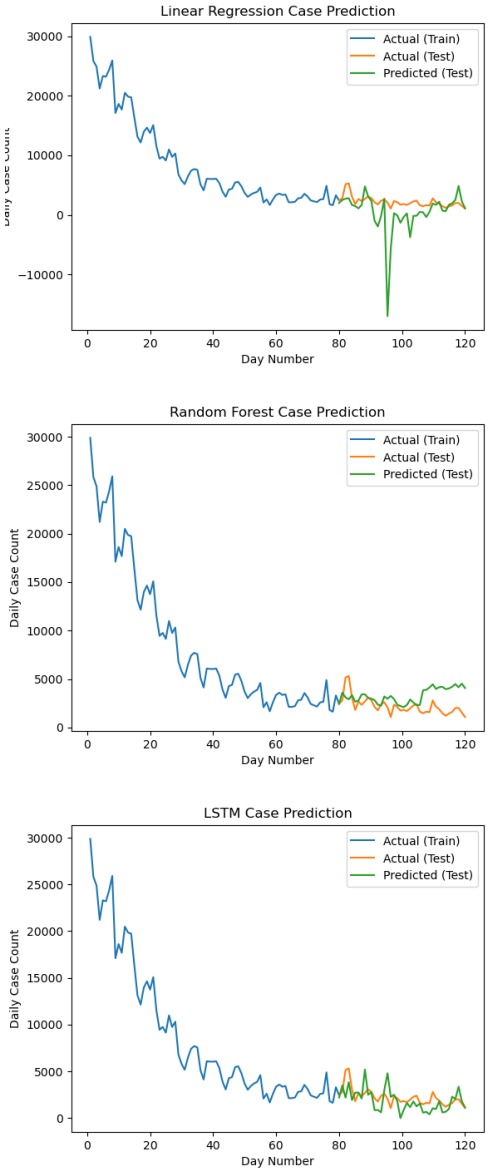

Figure 5: Model performance comparison for a 4-month period (January-April 2021)

As an evaluation metric for comparing our models, we computed the mean absolute percent error of each model's predicted output. The magnitude of cases varies drastically over the duration of time relevant to the COVID-19 pandemic (i.e., an estimate of case numbers that differs by 1000 would mean something very different in March 2020 and December 2020) and using a percent error metric helps to normalize this across different timeframes. For a number of fitted points n, actual value A_t, and forecast value F_t,

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\frac{A_t - F_t}{A_t}$$

To compare model performance over different time periods, we train and test our models and compute the mean absolute percent error for a sliding window of 4 months over the larger time frame of 21 months. This allows for a quantitative assessment of the relative model performance. As can be seen from Figure 6, the LSTM model generally produces results with a lower MAPE value over the overall time frame. There is also a significant increase in the error in the timeframes which include the beginning of 2021, but this is expected due to the huge decrease in COVID case numbers brought on by the vaccine rollout that occurred in January 2021, an effect which cannot be accurately predicted from these Google search trends alone.

These tests over a wide range of timeframes and durations also served to reduce the chance of overfitting to any specific period. On top of this, we also tested various means of sectioning our data into train/test sets, and limited the upper bound of the training data percentage. However, some overfitting effects and high error rates can be attributed to our evaluation metric. While MAPE was very valuable to compare our model performance across timeframes with varying case numbers, this can sensitive to outliers, which can often occur in data such as this. These outliers and spikes in the data can then skew the MAPE high, when making comparisons between noisy data and predictions. Some methods to improve this include using MdAPE (median average percent error), which is a slightly uncommon evaluation metric, or potentially applying a low-pass filter to the data before training (this would be considered in future work).

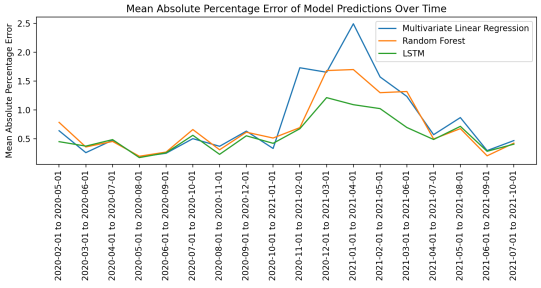

Figure 6: Effects on MAPE by considering different timeframes within COVID

5.4 Additional Experiments

State	Linear Regression	LSTM	Random Forest
Ohio	0.223	0.192	0.307
Alabama	0.507	0.277	0.411
Texas	0.353	0.347	0.507
Illinois	0.448	0.533	0.503
Florida	0.343	0.408	0.536

Table 2: Comparison of model performance (MAPE) on other states

While most of our work for this project was focused on data from California, we decided to see how well this model could be applied to other states. We hypothesized that states without extremely large metro areas would have improved performance in our model (due to the difference in how COVID spreads in more dispersed environments), and this seems approximately true based on these results, particularly for the application of LSTM to Ohio and Alabama. Illinois, which has case counts that are highly impacted by Chicago, sees poor performance across all models. Florida and Texas also perform poorly though, with linear regression seemingly beating LSTM and RF, indicating some weakness in the model.

This state comparison uses the timeframe between February and June 2020, with a time delay of 6 days, and a 75% train / 25% test split.

6 Conclusion/Future Work

In this work, we sought to predict future COVID-19 case numbers using only the trends in Google search data, for a set of 11 COVID-related queries such as "covid symptoms" and "coronavirus cases". In our comparison of three different models (linear regression, random forests, and LSTM), we saw that LSTM performed the best, achieving a mean average percent error (MAPE) of as low as 19%. This outcome matched our hypothesis that LSTM would perform the best of our three models, as this is better suited to time-series data, but interestingly, linear regression was not as far off as expected.

Overall, we found that in general, our models using only Google search terms will at best predict COVID case numbers within 25% (on average) of the true value, with this error dramatically increasing depending on the noise in the COVID data or external factors not represented in the model.

However, considering that these Google search trends are the sole training data this model had, this does reveal some interesting links between online searches and real-world viral spread, and this reinforces findings from previous studies such as the importance of the search-case delay.

While the world hopes that future work on COVID spread will soon be unnecessary, this pandemic could remain for longer than expected with new variants, and the data we collect today can help inform future epidemics/pandemics. Such future work may include training and testing the performance of deep neural networks or conducting expansive feature selection experiments across a much larger collection of potentially related search queries to help reveal precisely which search terms have the most predictive relevance to COVID case data.

7 Contributions

Both team members contributed evenly to the project and the final report. Both Dan and Megan worked to implement the data scraping and preprocessing code and trained the baseline model. Megan focused on LSTM and code integration while Dan focused on RF and hyperparameter tuning.

Github link:
https://github.com/mb2532/CS229_FinalProject.git

References

[1] Ayyoubzadeh S, Ayyoubzadeh S, Zahedi H, Ahmadi M, R Niakan Kalhori S Predicting COVID-19 Incidence Through Analysis of Google Trends Data in Iran: Data Mining and Deep Learning Pilot Study JMIR Public Health Surveill 2020;6(2):e18828 https://publichealth.jmir.org/2020/2/e18828

[2] "COVID-19 Data Repository by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University." https://github.com/CSSEGISandData/COVID-19

[3] H. L. Nguyen, Z. Pan, H. Abu-gellban, F. Jin., Y. Zhang, "Google Trends Analysis of COVID-19 Pandemic," Frontiers, 7 Nov 2020. Accessed from arXiv: https://arxiv.org/pdf/2011.03847.pdf

[4] Lampos, V., Majumder, M.S., Yom-Tov, E. et al., "Tracking COVID-19 using online search." npj Digit. Med. 4, 17 (2021). https://doi.org/10.1038/s41746-021-00384-w

[5] Prasanth, S., Singh, U., Kumar, A., Tikkiwal, V. A., Chong, P. H. J., "Forecasting Spread of COVID-19 using Google Trends: A Hybrid GWO-Deep Learning Approach." Chaos, Solitons Fractals, Volume 142, January 2021, 110336, https://doi.org/10.1016/j.chaos.2020.110336

[6] "Pytrends: Pseudo API for Google Trends", https://github.com/GeneralMills/pytrends

[7] Wang, MY., Tang, Nj, "The Correlation Between Google Trends and Salmonellosis." BMC Public Health 21, 1575 (2021). https://doi.org/10.1186/s12889-021-11615-w

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

[9] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Mike Schuster, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.