CS230

Predicting Flight Delays with Deep Learning

Ethan Chi
Department of Computer Science
Stanford University
ethanchi@cs.stanford.edu

Jillian Tang
Department of Computer Science
Stanford University
jilingt@stanford.edu

Abstract

The prediction of flight delays is a problem of great interest, due to the steep monetary and logistical costs that delays incur. However, previous probabilistic and machine learning approaches have not seen great success, due to the highly interconnected and sequential nature of air transit networks. In this paper, we present a deep learning model which uses a modified version of the *LSTNet* architecture to predict flight delays. Trained on 251,756 flights departing from ORD in 2009, our model was able to significantly outperform baseline statistical models, as well as achieving performance comparable to the state of the art.

1 Introduction

In 2017, approximately 24% of all flights in the US were delayed by at least 15 minutes, with numbers even higher in other countries. These delays present major challenges for all areas of the American transport system, including airports, airlines, and passengers. Consequently, predicting flight delays is a problem of great interest.

Due to continuous logging of both flight data and of weather data at airports, there is an enormous amount of available data; however, predicting flight delays has historically been a difficult task due to the interconnected and sequential nature of flight delays.

2 Related work

Prior to 2005, several authors used non-AI techniques for this task [1]. Mueller et al. [2] attempted to fit flight delays using normal and Poisson distributions; Boswell et al. [3] recognized the fundamentally sequential nature of the problem and used transition matrices to model the propagation of delays to subsequent flights, while Wu et al. [4] used a Markov chain model to analyze the spread of flight delays. However, all of these focused on average trends in delay given an airport and a date; none attempted to predict individual flight delays.

The advent of machine learning techniques enabled researchers to investigate the prediction of individual flight delays. Rebollo et al. [5] used random forest classification on network delay states and temporal variables to predict flight delays for specific flight connections, achieving an average accuracy of 80.9%. Lu et al. [6] used a C4.5 decision tree to predict flight delays, achieving an accuracy of 79.27%. While these were a great improvement over only being able to predict average delays over a time period, neither of these models used sequential data, which limited their accuracy. In contrast, Balakrishna et al. [7] used a nonparametric reinforcement learning algorithm to predict taxi-out time (a significant contributor to flight delay), achieving an accuracy of 93.7% at Tampa International Airport.

Despite the growth in usage of machine learning techniques, deep learning techniques have not seen significant use in modelling flight delays. The only major work in the literature on this subject is by Kim et al. [8], who used a LSTM surrounded by hidden dense layers to model average flight delays on a day-to-day basis

CS230: Deep Learning, Winter 2019, Stanford University, CA.

(either 7 or 9 days). The output of this LSTM was then combined with data for the specific flight through a fully-connected dense layer to output a prediction. This architecture was able to achieve an accuracy of anywhere from 71.34% to 90.95% depending on the airport analyzed, with an overall accuracy of 86.99%.[1]

3 Dataset and Features

3.1 Dataset

Our dataset was drawn from the following two sources:

- Flight delay data aggregated from all commercial flights in the USA (Bureau of Transportation Statistics [9]).
- Weather data from the National Oceanic and Atmospheric Administration, which records data at each airport from hourly to five-minute increments. The data was rehosted and cleaned by Iowa State's Mesonet system. [10]

In total, our dataset contained 394,781 flights which departed from Chicago O'Hare International Airport in 2011. The dataset was split according to a 80%-10%-10% train/dev/test distribution.

3.2 Features

Each of the flights was coded with the following features:

`month`	Month	`p01mm`	1-hour precipitation (mm)
`day`	Day	`vsby`	Visibility (miles)
`time`	Time	`skyc1`	Cloud coverage
`station`	Departure airport	`wxcodes`	Present weather codes
`tmpc`	Temperature (°C)	`OP_UNIQUE_CARRIER`	Airline carrier
`dwpc`	Dew point (°C)	`OP_CARRIER_FL_NUM`	Flight number
`relh`	Relative humidity (%)	`ORIGIN_AIRPORT_ID`	Departure airport
`feel`	Heat index/Wind chill (F)	`DEST_AIRPORT_ID`	Destination airport
`drct`	Wind direction	`CRS_ARR_TIME`	Scheduled arrival time
`sped`	Wind speed (mph)	`CANCELLED`	Cancelled or not
`alti`	Altimeter (in)	`CANCELLATION_CODE`	Reason for cancelling
`mslp`	Sea level pressure (mb)	`ARR_DELAY`	**Outcome—arrival delay**

3.3 Preprocessing

We merged flight data with a based on the available weather closest to the time of departure of the flight. Categorical variables such as `month`, `station`, `skyc1`, `wxcodes`, `OP_UNIQUE_CARRIER`, `ORIGIN_AIRPORT_ID`, `DEST_AIRPORT_ID`, and `CANCELLATION_CODE` were then one-hot encoded. In addition, each weather feature was included both for the departure and arrival airport. In total, after preprocessing, each flight had 325 data features.

To avoid saturation, all non-categorical features were normalized to have zero mean and unit variance.

3.4 Characteristics

Figure 1 displays correlation plots of flight delay and non-categorical weather features. There are at most subtle correlations on hourly precipitation and features relating to wind speed, with no immediately obvious correlations for the other features; therefore, a more complex model, such as a neural network, is necessary to fit the data.

[1] However, see Section 3.4 for issues with the use of solely accuracy as a metric.

Approximately 80% of the flights in the data were not delayed, meaning that this data suffers from a class imbalance. This has large implications for our metrics – namely, high accuracy does not necessarily correspond to a good classifier, as a classifier that predicted *non-delayed* for every flight would achieve an accuracy of about 80%. Therefore, we evaluate all of our models on both accuracy and the F1 score (the harmonic mean of precision and recall) to provide a better view of how a model fits the data.

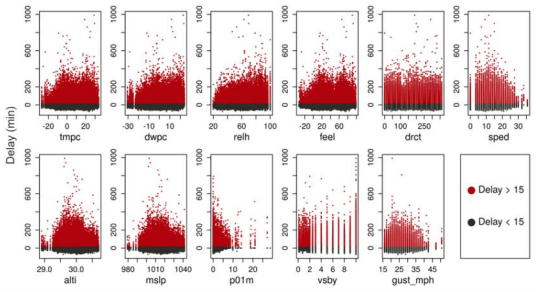


Figure 1: Correlations of weather and delay

4 Methods

4.1 Machine Learning Approaches

As a baseline, we evaluated the following three machine and statistical learning techniques:

- **Naïve Bayes** – a probabilistic classifier which assumes independence between features
- **Logistic Regression** – a simple binary classifier
- **Gradient Boosting Machine** – fits ensemble of decision trees. We used 1000 trees with a learning rate of 0.1 selected after tuning.

4.2 Deep Learning Approaches

We implemented two deep learning models, both using the Tensorflow [11] Python library.

LSTM + CNN implements a model inspired by LSTNet [12], an architecture for modeling short- and long-term time series data. As the first layer of our model, we run destination airport data[2] and arrival delay for the 15 most recent flights through a single convolutional layer without pooling with k filters, each of size $(filter\_width, num\_variables)$. Here, the x-dimension represents the position in the sequence of flights, while the y-dimension represents the variables of the input data. Due to the high dimensionality of our input, this step is useful to extract local dependencies between variables and reduce the dimensionality of the input, greatly reducing the amount of computation necessary going forward. In the CNN, the kth filter applies the following transformation:

$$h_k = W_k * X + b_k \tag{1}$$

where $*$ represents the convolution operation.

The single convolutional layer has 6 filters of width 6 and height $num\_variables$. The output of the convolutional layer is fed into a recurrent layer as described in LSTNet. However, in place of the Gated Recurrent Unit used in LSTNet, we use a LSTM (long short-term memory network) [13], which remembers information over arbitrary time intervals.[3] The hidden state of each LSTM at time t is governed by the following equations:

$$\begin{aligned} f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\ i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\ o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\ c_t &= f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\ h_t &= o_t \circ \sigma_h(c_t) \end{aligned} \tag{2}$$

[2]Note that we don't feed in data for the departure airport, as it's identical for all of the flights in the sequence and the flight to be predicted.

[3]We found that LSTMs performed better on this task.

The output of the final RNN node is then combined with the weather data for the current flight and fed into five dense layers, all but the last of which use the $ReLU$ activation function. The final layer uses the sigmoid activation function to produce an output probability between 0 and 1.[4]

To ameliorate the issue of class imbalance, we use a slightly modified log loss to ensure that the model does not always predict no delay. Here, the higher coefficient on the first term ensures that examples with delayed flights are weighted much more heavily than non-delayed flights.

$$\mathcal{L} = -2.6y \log \hat{y} - (1 - y) \log(1 - \hat{y}) \quad (3)$$

LSTM + Dense is identical to LSTM + CNN, except that the preprocessing CNN is replaced with several densely-connected layers as described in Kim et al. [8]. (We used 3 layers in our final network.)

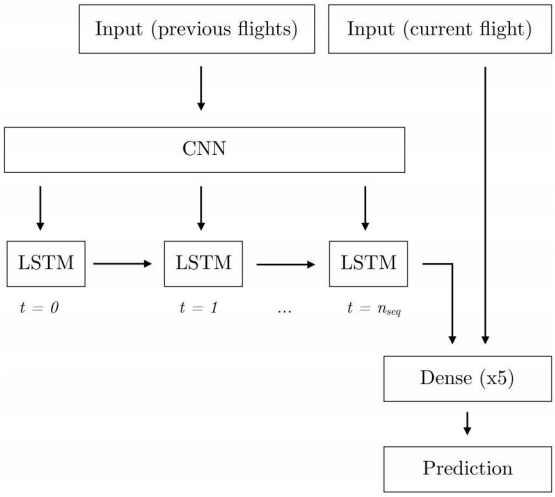


Figure 2: LSTM + CNN Architecture

5 Results and Discussion

Due to the class imbalance of delayed and non-delayed flights, evaluating models on accuracy can be deceiving. For example, a model that predicts all flights to be non-delayed will have approximately 80% accuracy, due to correctly predicting all non-delayed flights. This kind of approach is especially problematic, since false negatives (predicting a delayed flight to be non-delayed) are especially deleterious for users. Instead, a better metric is the F1 score, the harmonic mean of the precision and recall. Therefore, throughout this paper, we evaluated our models on a combination of accuracy and F1 score (the mean of precision and recall).

For our deep learning models, we investigated various combinations of hyperparameters:

- For our **learning rate**, we examined the values {0.01, 0.003, 0.002, 0.001, 0.0005, 0.0001}. We found that the best performance came with a value of 0.0005, which led to a longer training time but made training more reliable
- We used **Batchnorm**, as we found it greatly sped up training. We examined the values {100, 1000, 2000, 5000, 10000}. We used a batch size of 5,000, as this significantly sped up computation without affecting accuracy.
- For our **sequence length**, we examined the values {6, 8, 9, 10, 15, 20, 25}. We used a sequence length of 15 (i.e. we considered the last 15 flights to have departed), as beyond this point increasing the sequence length did not significantly increase performance.
- We ran the model for 500 **epochs**, as we started to overfit after this point.
- Due to major overfitting issues during training, we apply inverted **dropout** with `keep_prob` 0.5. In addition, we apply **L1 regularization** with $\lambda = 0.003$ to the weights of both the CNN (if applicable) and dense layers.

5.1 Results

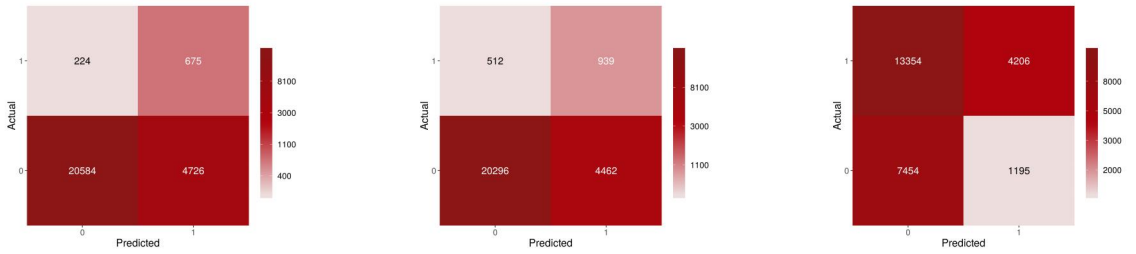
We present our results in Table 1. The cutoff for converting predicted probabilities to classes is 0.5.

As expected, statistical learning methods performed poorly. Although a GBM and logistic regression performed the best in terms of precision, this was because they predicted mostly *non-delayed* for all flights, as seen from the low recall and F1 values. On the other end, the Naive Bayes model predicted mostly *delayed* for all flights, achieving the highest recall but low precision.

[4]The original LSTNet framework as described in Lai et al. utilizes a hand-tuned *recurrent skip connection*, which allows the network to leverage periodic patterns in datasets. However, as analysis of our dataset shows it not to be strongly periodic, we did not implement this connection.

Table 1: Model Performance

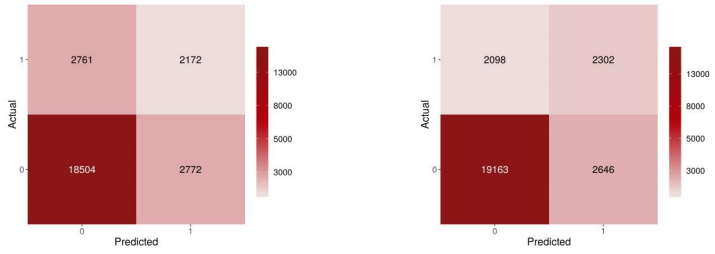
Model	GBM	Log. Reg.	Naïve Bayes	LSTM + Dense	LSTM + CNN
Accuracy	81.11%	81.02%	44.49%	78.89%	**81.90%**
Precision	**75.08%**	64.71%	23.95%	44.03%	52.32%
Recall	12.50%	17.39%	**77.87%**	43.93%	46.52%
F1 score	21.43%	27.41%	36.64%	43.98%	**49.25%**



Gradient Boosting Machine *Logistic Regression* *Naïve Bayes*

Figure 3: Confusion Matrices for Baseline Methods

Due to their ability to model sequential data, the deep learning algorithms (LSTM + Dense and LSTM + CNN) were far more effective than the baseline machine learning models. The LSTM + CNN model performed the best, with a F1 score of 49.21% and accuracy of 81.90%. Indeed, we were able to replicate the performance of the state-of-the-art paper in the field [8], using a model with much less computational complexity and expense. The LSTM + Dense model also outperformed the machine learning models, but was much more computationally expensive than LSTM + CNN.



LSTM + Dense *LSTM + CNN*

Figure 3: Confusion Matrices for Deep Learning Methods

6 Conclusion and Future Work

We implemented five models (three baseline machine learning models and two deep learning models) to predict flight delays given a particular flight's information and associated weather data. As we anticipated, our LSTM + CNN model greatly outperformed baseline models in terms of both F1 score and accuracy. Additionally, we were able to replicate the performance of the state-of-the-art paper in this field at less computational expense due to using computationally cheaper convolutional layers in place of dense layers.

As there is a large body of available data on flight delays, there is potential to incorporate data from more airports and years into our model. This would increase our training set diversity which could help it to generalize better to unseen data. Another possible avenue of exploration may be to investigate increasing the complexity of the architecture, such as by incorporating multiple layers of LSTMs to account for time patterns at different scales. It may also be possible to integrate features such as national threat level or economic state to explain patterns in the data that are not based on weather or previous flight statuses.

7 Contributions

Both authors contributed to preprocessing the data, writing the model, performing error analysis, and writing up the results.

Ethan Chi wrote the scrapers for flight delay and weather data and performed the hyperparameter tuning that gave the best final results.

Jillian Tang created the baseline ML models and produced data visualizations.

8 Code

The repository for this project is available here: https://github.com/jilingt/cs230-final-project.

9 Acknowledgements

The authors are grateful to Sarah Najmark, Shervine Amidi, and Will Song for helpful advice and discussions. Additionally, the authors thank Stanford's community computing environment, FarmShare, for the use of their cluster.

References

[1] A. Sternberg, J. Soares, D. Carvalho, and E. Ogasawara, "A review on flight delay prediction," *arXiv preprint arXiv:1703.06118*, 2017.

[2] E. Mueller and G. Chatterji, "Analysis of aircraft arrival and departure delay characteristics," in *AIAA's Aircraft Technology, Integration, and Operations (ATIO) 2002 Technical Forum*, 2002, p. 5866.

[3] S. B. Boswell and J. E. Evans, *Analysis of downstream impacts of air traffic delay*. Lincoln Laboratory, Massachusetts Institute of Technology, 1997.

[4] W.-w. Wu, T.-t. Meng, and H.-y. Zhang, "Flight plan optimization based on airport delay prediction," *Journal of Transportation Systems Engineering and Information Technology*, no. 6, p. 29, 2016.

[5] J. J. Rebollo and H. Balakrishnan, "Characterization and prediction of air traffic delays," *Transportation research part C: Emerging technologies*, vol. 44, pp. 231–241, 2014.

[6] L. Zonglei, W. Jiandong, and Z. Guansheng, "A new method to alarm large scale of flights delay based on machine learning," in *2008 International Symposium on Knowledge Acquisition and Modeling*. IEEE, 2008, pp. 589–592.

[7] P. Balakrishna, R. Ganesan, and L. Sherry, "Accuracy of reinforcement learning algorithms for predicting aircraft taxi-out times: A case-study of tampa bay departures," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 6, pp. 950–962, 2010.

[8] Y. J. Kim, S. Choi, S. Briceno, and D. Mavris, "A deep learning approach to flight delay prediction," in *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*. IEEE, 2016, pp. 1–6.

[9] B. of Transportation Statistics, "On-time performance: Reporting carrier on-time performance (1987-present)," Bureau of Transportation Statistics, 2018.

[10] N. Oceanic and A. Administration, "Mesonet," 2018.

[11] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[12] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long-and short-term temporal patterns with deep neural networks," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 2018, pp. 95–104.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.