Journal of Aviation/Aerospace Education & Research

Volume 33
Number 4 *Special Issue: Artificial Intelligence in Aviation*

Article 9

2024

An Enhanced Deep Autoencoder for Flight Delay Prediction

Desmond B. Bisandu PhD
Cranfield University, desmond.bisandu@cranfield.ac.uk

Dan Andrei Soviani-Sitoiu MSc
Cranfield University, d.soviani@fugro.com

Irene Moulitsas PhD
Cranfield University, i.moulitsas@cranfield.ac.uk

Follow this and additional works at: https://commons.erau.edu/jaaer

Part of the Aerospace Engineering Commons, Computer Sciences Commons, and the Data Science Commons

Scholarly Commons Citation
Bisandu, D. B., Soviani-Sitoiu, D. A., & Moulitsas, I. (2024). An Enhanced Deep Autoencoder for Flight Delay Prediction. *Journal of Aviation/Aerospace Education & Research, 33*(4). DOI: https://doi.org/10.58940/2329-258X.2056

This Article is brought to you for free and open access by the Journals at Scholarly Commons. It has been accepted for inclusion in Journal of Aviation/Aerospace Education & Research by an authorized administrator of Scholarly Commons. For more information, please contact commons@erau.edu.

JAAER

An Enhanced Deep Autoencoder for Flight Delay Prediction

Desmond B. Bisandu[1a], **Dan Andrei Soviani-Sitoiu**[1b], **Irene Moulitsas**[1c]
[1]Cranfield University, UK
[a]desmond.bisandu@cranfield.ac.uk, [b]d.soviani@fugro.com, [c]i.moulitsas@cranfield.ac.uk

Abstract

Accurate and timely flight delay prediction cannot be overemphasized because of the ever-increasing demand for air travel and its importance in deploying intelligent transportation systems. Nonetheless, there has not been a universal solution to the problem, as more intelligent flight decision systems are required for the aviation industry's future growth. Existing flight delay classification and prediction approaches are mainly shallow traffic models and do not satisfy many applications in the real world. Our motivation to rethink the deep architecture model for predicting flight delays emanates from the problem. In this research, we proposed a technique that modified stacked autoencoder architecture parameters for training the network and understanding the link between space, time and information gained from the flight on-time data. We developed three different types of autoencoders based on the architecture of the modified stacked autoencoder. The models learn the generic flight delay features, and it's trained greedily in a layer-wise fashion. To the best of our knowledge, this is the first time these performances of vanilla autoencoder, logistic regression autoencoder and Multilayer perceptron for classification were evaluated based on the developed modified stacked autoencoder architecture. Moreover, our experiment demonstrates that the models achieved varying levels of accuracy in the flight delay classifications task. The deep vanilla autoencoder shows superior accuracy, recall and precision performance compared to logistic regression autoencoder and Multilayer perceptron autoencoders at different parameter settings.

Keywords: *Autoencoder, Deep Learning, Flight Delay Prediction, Machine Learning, Flight On-Time Dataset*

Introduction

Recently, air transportation has become a more acceptable means of travel by humans. Precise flight delay categorization is profitable but challenging due to the uncertainties and breakdowns inside the procedures. Even though people can travel by car or boat, air travel is superior for most passengers due to its efficiency and reliability (Tan et al., 2018). Airlines, airports, and passengers lose when flights are delayed. All actors in the commercial aviation industry rely on their predictions when making decisions.

The complexity of the air transport industry and the vast collection of flight data make predicting flight delays challenging. Delays, defined as deviations from scheduled arrival/departure times, are key performance indicators in aviation. In 2013, flight delays affected 36% of flights in Europe, 31.1% in the US, and led to over 30-minute delays or cancellations for 16.3% of flights in Brazil (Carvalho et al., 2021). Factors contributing to delays include extreme weather, late-arriving aircraft, air carrier issues, the National Aviation System, and security, with respective impacts reported in 2017 by the Bureau of Transportation Statistics (BTS) USA (Q. Li & Jing, 2022; Mofokeng & Marnewick, 2017; Muros Anguita & Díaz Olariaga, 2023; Yazdi et al., 2020)

Flight delays negatively impact passengers, airlines, and airports economically. Passengers incur higher travel costs due to the need for early arrival, while airlines face fines and increased operational expenses. Environmental concerns arise from increased fuel consumption and emissions. Delays also affect airline marketing by undermining customer loyalty and influencing consumer choice, linked to factors like flight frequency and service complaints (Balakrishna, Ganesan, & Sherry, 2008; Balakrishna et al., 2009; Bisandu & Moulitsas, 2023, 2024; Bisandu et al., 2022; Dhanawade et al., 2019; Hondet et al., 2018; Mizufune & Katsumata, 2019; Rebollo & Balakrishnan, 2014). Predicting delays aids in better management decisions for airports and airlines and allows passengers to adjust plans (Balakrishna, Ganesan, Sherry, & Levy, 2008; Ganesan et al., 2010). The analysis of extensive aviation data, enhanced by data scientists' computational skills, supports understanding and improving the flight environment (Guo et al., 2021, 2022; B. Yu et al., 2019; Y. Yu et al., 2021).

This study contributes to the commercial air transport industry by analysing flight delay predictions through machine learning and data science, focusing on non-weather-related delays. It reviews current trends in delay prediction, evaluates various modelling strategies, and highlights the ongoing challenge of developing sustainable solutions amid industry growth. We introduce a method to predict non-weather delays for US flights using a dataset of on-time flights, employing a deep autoencoder with adjusted parameters. We assess the effectiveness of three models: Multilayer Perceptron Autoencoder, Vanilla Autoencoder, and Logistic Regression, utilizing a deep-stacked autoencoder architecture with re-scaled hidden layers. The models' performances were evaluated based on accuracy, precision, and recall.

The rest of the paper is organized as follows: Section 2 presents the related work; Section 3 discusses the methodology approach, while in Section 4, the results and discussion are presented, and Section 5 has the conclusion and future direction.

Related Work

For several decades, accurate flight delay prediction has become a challenging issue for practitioners and researchers. However, the advent of high computational methods and algorithms has improved the performance of deep learning methods demonstrating state-of-the-art results compared to statistical or classical machine learning techniques evaluated on their prediction accuracy, precision, and recall.

Gupta (2018) introduced a deep learning approach using a stacked autoencoder to predict airport delay times, contributing to deep learning applications in aviation and civil aviation management. The study accounted for the complex temporal and spatial relationships of airline delays and used FAA data from January 2016 to December 2017, including details like date, time, and delays. Hondet et al. (2018) developed an optimized forecasting model employing the Levenberg-Marquart algorithm for flight delay predictions, aiming to reduce data dimensions for deep learning networks. They tested the model's effectiveness without the denoising autoencoder and without the LM algorithm, using under-sampling and over-sampling for data balance. Their findings indicate that combining a stack denoising autoencoder with the LM algorithm and balancing the dataset enhances forecasting accuracy and precision.

Pejovic et al. (2009) developed a two-stage model employing supervised machine learning to predict on-time flight performance, initially using binary classification for delay occurrence and then regression for delay duration. Gradient Boosting Classifier excelled in classification, whereas Extra-Trees Regressor led in regression. Bhadra (2009) modelled airports as network nodes and flights as links, using an agent-based methodology to simulate delay propagation, which closely mirrored real-world patterns and highlighted busy airport clusters. Shabanpour et al., (2022) combined Bayesian Networks and Gaussian mixture models to predict downstream flight delays based on upstream conditions, achieving high accuracy and reliability in their real-time analysis.

Shabanpour et al. (2022) explored deep learning, specifically using RNN and LSTM architectures, to predict air traffic delays. They improved daily delay status prediction by integrating historical performance and weather data, leading to precise flight delay forecasts. Campanelli et al. (2014) and Ciruelos et al. (2015) addressed optimizing flight routes and scheduling to minimize congestion and delays within an air traffic network. Ogunsina and Okolo (2021) applied subgroup detection in data mining to identify characteristics of flight delays, revealing that factors like the day of the week and month could highlight flights prone to significant delays.

Chen et al. (2016) and Saadat and Moniruzzman (2019) evaluated a machine learning-based regression model for flight delay predictions, achieving high Coefficient of Determination scores for both arrivals and departures. Yazdi et al. (2020) proposed a method using neural networks and deep learning to predict airline arrival delays, considering factors from weather to distance. The study found neural networks with a single hidden layer of three neurons achieved 92% accuracy, while deep networks with two layers of four neurons each reached 77% accuracy. They aimed to assess when Deep Feedforward Neural Networks surpass Support Vector Machines and simpler Neural Networks, discovering that increasing epochs significantly enhances prediction accuracy.

Ciruelos et al. (2015) aimed to develop classifiers for predicting flight delays with a focus on cost sensitivity. Rong et al. (2009) introduced a classification strategy that selects the most accurate single-rule prediction based on minimal error, highlighting patterns distinguishing delayed from on-time flights and incorporating weather forecast data. Kim et al. (2016) sought to predict departure delays at specific airports or links using Random Forest, noting the algorithm's robustness over a 24-hour prediction horizon and its adaptability to network status variability. Cai et al. (2017) applied the Reinforcement Learning (RL) algorithm to predict taxi-out times at airports with improved accuracy using surface surveillance data and a Markov decision process for dynamic system state analysis. Mofokeng and Marnewick (2017) utilized a nonparametric RL approach within a stochastic dynamic programming framework for taxi-out time prediction, demonstrating its effectiveness even at challenging airports.

Proenca et al. (2019) introduced a machine learning technique for identifying large-scale aircraft delays using a two-step process involving unsupervised learning for standard setting and supervised learning for alert model creation. Manna et al. (2018) developed a multilevel input layer ANN to handle nominal variables and provide interpretable connections between inputs and outputs, showing superior performance to traditional gradient descent backpropagation in terms of prediction error and training time. Venkatesh et al. (2017) compared different forecasting methods for air traffic delays, finding that ANN excelled in Origin-Destination delay classification, while the Markov Jump Linear System was most effective for delay regression, noting the varying relevance of time of day and network characteristics in delay prediction.

Leveraging Deep Learning's success in predictive tasks, particularly for flight delays, this study introduces a method to enhance delay predictions by adjusting autoencoder parameters, specifically through nested multiplication of hidden layers and employing dropout before the output layer. This approach is tested across three autoencoder types. Unlike previous research focusing on data pre-processing and conventional deep autoencoders, this paper uniquely explores parameter re-scaling within a stacked autoencoder framework. The goal is to assess the effectiveness of re-scaled parameters on different autoencoders for predicting flight punctuality, aiming to contribute insights for accurately forecasting delays in aviation.

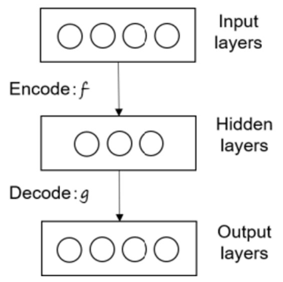
Materials and Methods

There are many interesting applications of autoencoders in different fields, such as visualization, dimensionality reduction and data compression. Recently, it has been discovered to have the capability of training neural networks because of its difference in the gradient magnitudes of the lower and higher layers and the lack of generalization of deep networks due to many parameters in the deep neural network. Also, the objective function curvature is difficult to find a good local minimum. Pretraining the network helps divide the entire deep network into a sequence of steps. Pretraining the deep network is intended to address the difficulties mentioned above, and the training process of the deep network is divided into steps, such as the pretraining step, where shallow autoencoders sequence use unsupervised data one layer greedily at a time. Training the last layer with a supervised dataset and finally fine-tuning the entire network with supervised data and backpropagation.

The autoencoder can learn data features and represent the statistical characteristics of training samples. The input of the model is its target output (Ye et al., 2020). The autoencoder contains an encoder and a decoder. The encoder shrinks the input while the decoder tries to reconstruct the input from the encoder's compressed form. The encoder model is saved after training, whereas the decoder is deleted. The encoder can then be used as a data preparation approach to extract features from raw data so that a new machine learning model (Bisandu & Moulitsas, 2022; Bisandu et al., 2021; Ma et al., 2015; Xiao et al., 2014). An autoencoder with a hidden layer and an output layer is shown in Figure 1 (Ye et al., 2020).

Figure 1

Structure of Autoencoders



The reconstruction process of an input x of an autoencoder can be explained using Equation 1, Equation 2, Equation 3, Equation 4, and Equation 5, respectively.

Assume we have a set of data $x_{(i)}$ belonging to the set of real numbers at points $\{x_{(1)}, x_{(2)}, \ldots, x_{(m)}\}$ with many dimensions. Mapping the set of the input to another data point set $z_{(i)}$ with points $\{z_{(1)}, z_{(2)}, \ldots, z_{(m)}\}$ having a lower dimension than $x_{(i)}$ and can be reconstructed back to $x_{(i)}$ effectively. Mapping the data back and forth in a systematic manner can be achieved by encoding the input layer $x_{(i)}$ into a hidden layer $z(x_{(i)})$ and $\bar{x}(z_{(i)})$ is the reconstructed result shown in Equations 1 and 2, respectively.

$$z(x_{(i)}) = W_1 x_{(i)} + C1 \tag{1}$$

$$\bar{x}(z_{(i)}) = W_2 z_{(i)} + C2 \tag{2}$$

where W_1 and $C1$ are the weighted matrix and coding bias vector, W_2 and $C2$ are the decoding matrix and coding bias matrix, respectively. To minimize stochastic gradient descent, the reconstruction error of approximating $x_{(i)}$ by $\bar{x}_{(i)}$ we set an objective function which sums the squared differences between $x_{(i)}$ and $\bar{x}_{(i)}$:

$$\mathcal{L}(W_1, C1, W_2, C2) = \sum_{i=1}^{n} (\bar{x}_{(i)} - x_{(i)})^2 \tag{3}$$

$$= \sum_{i=1}^{n} (W_2 z_{(i)} + C2 - x_{(i)})^2 \tag{4}$$

$$= \sum_{i=1}^{n} W_2(W_1 x_{(i)} + C1) + C2 - x_{(i)})^2 \tag{5}$$

We are looking closely at how an autoencoder works. It takes a high-dimensional input, such as a picture or a vector, and runs it through a neural network that compresses the data into a shorter form with two principal components. The first is the encoder, a collection of fully connected or convolutional layers that compress the input into a smaller representation with fewer dimensions than the data, a process known as a bottleneck. It now reconstructs the input using full connected or convolutional layers from this bottleneck.

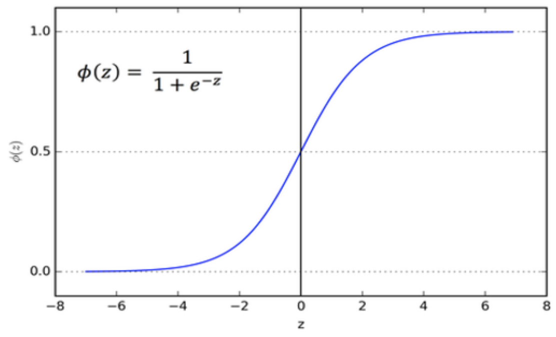
We employed the sigmoid activation function in the deep neural network. It is a non-linear AF that outputs a continuous value between 0 and 1, defined as shown in Equation 6:

$$a = g(z)\frac{1}{1+e^{-z}} \tag{6}$$

The sigmoid function is like the step function in that it is continuous and avoids the leap in output values; the sigmoid function is a mathematical function that maps any input value to a value between 0 and 1 (Balakrishna, Ganesan, & Sherry, 2008; Balakrishna et al., 2009; Ganesan et al., 2010; Gopalakrishnan & Balakrishnan, 2017; Le Ny & Balakrishnan, 2011; Pfeil & Balakrishnan, 2012; Pfeil et al., 2008; Rebollo & Balakrishnan, 2014). Figure 2 shows a plot of the sigmoid AF (Aigner et al., 2007; Alkhayrat et al., 2020; Belcastro et al., 2016; Castaing et al., 2016; Ciruelos et al., 2015; Orimoloye et al., 2020; Osorio, 2019; Rodríguez-Sanz et al., 2022).

Figure 2

Sigmoid Activation Function



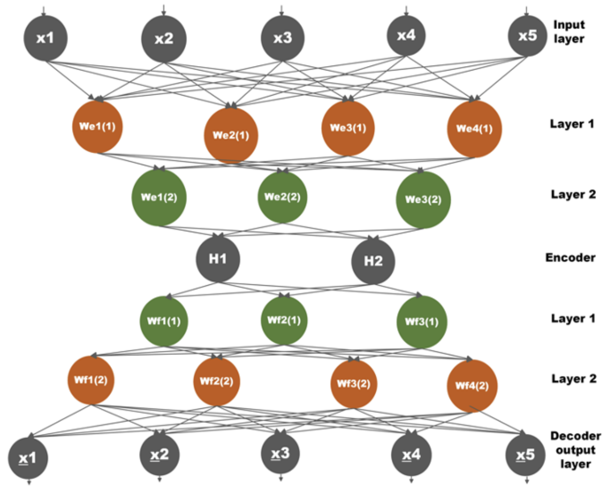
Proposed Architecture

Using an autoencoder for flight delay classification leverages its strength in feature learning and dimensionality reduction to handle the complex and high-dimensional nature of flight data. Our proposed architecture is based on the stacked autoencoder framework, a form of deep neural network characterized by layers of sparse autoencoders connected sequentially, where the output of each hidden layer serves as the input to the subsequent one. We enhanced this structure by embedding a single dropout within all hidden layers and implementing a normalization technique through multiplication of each layer's output with its predecessor's before progressing to the next layer. Additionally, we incorporated dropout in the final hidden layer specifically to mitigate potential overfitting or underfitting, ensuring the model's robustness and generalizability. This is to ensure that models perform well not just on the data they were trained on, but also on new, unseen data during the classification or prediction. The compressed features extracted by the encoder are then used as input for a classification model. This model is trained in a supervised manner to predict flight delays, using labels that categorize flights based on their delay status (e.g., on-time, minor delay, major delay). The reduced dimensionality of the input data helps improve the training efficiency and can lead to better model performance since the model can focus on the most relevant features.

The hidden layers are trained using an unsupervised technique and subsequently fine-tuned using a supervised method, as shown in Figure 3.

Figure 3

The Proposed Architecture



A regular predictor should be added to the top layer of the network to use the deep autoencoder network for flight delay prediction. The basic structure of a stacked autoencoder is as follows:

- **i.** Train the autoencoder and acquire the taught data using input data.

- **ii.** The preceding layer's learned data is utilized as input for the next layer and works until the training is finished.
- **iii.** Once all the hidden layers have been trained, the cost function is minimized using the back-propagation technique, and weights are adjusted with the training data for fine-tuning.

Experimentation

We developed our proposed stacked autoencoder network architecture on three approaches: Multilayer perceptron, vanilla, and logistic regression.

Multilayer Perceptron Autoencoder

In this approach, rebuilding the model feature can be neglected after fitting the model, and the model can be used up to the bottleneck layer. The model produces a fixed-length vector with a compressed version of the input data at the bottleneck layer.

Figure 4 shows the entire process of the Multilayer perceptron autoencoder.

It starts taking the input from the provided dataset and filtering the data due to memory concerns. The data is divided into 70% train and 30% test sets before designing and fitting the model, and the input is scaled by normalizing the values range between 0-1. The encoder is defined with various numbers of hidden layers. A similar structure will determine the decoder but in a reverse manner. Sigmoid activation and batch normalization ensure that the model learns well. Since reconstruction is a multi-output regression issue, the model will be trained using the Adam version of stochastic gradient descent, which minimizes the mean squared error. After training, it plots the loss for the train and test sets to confirm the model learned the classification problem, the precision-recall curve and the precision and recall for different threshold values.

Logistic Regression Autoencoder

The approach takes the input from the dataset, as in Figure 5.

The data is scaled and filtered to make it suitable for the autoencoder. The model consists of an input layer, with the size of the input variables taken from the dataset. The encoder comprises several hidden layers to observe and analyse the model's behaviour when the number of hidden layers is changed. The output layer takes the input from the last hidden layer and is retained to decode and visualize the encoded data. Data encoding can aid in creating a linear classification border for the data. The Logistic Regression model is used on the encoded data. It continuously creates training and testing data from the original and encoded data, makes the Logistic Regression model, and assesses its effectiveness. After it builds the Logistic Regression model, it plots the loss for the 70% train and 30% test sets to confirm the model learned the classification problem, the precision-recall curve and the precision and recall for different threshold values.

Vanilla Autoencoder

In the vanilla autoencoder (see Figure 6), the input and the output are the same, the first being reconstructed using the Adam optimizer and the mean squared error loss function.

As shown in Figure 6, it starts by reading the input from the dataset, filtering and replacing the input names with specific variables names used for the vanilla autoencoder, such as 1, 2,... In the pre-processing stage, the normalization takes an important role in scaling down the range of data before it is used in the next stage. The data is again split into train and test sets, and the class label is dropped since it is an unsupervised method. As in the other two approaches, the autoencoder consists of an encoder and a decoder; the encoder has multiple hidden layers with a specific hidden size each. The decoder takes the information from the last hidden layer and reproduces the classification. After the model is trained, it plots the loss for the train and test sets to confirm the model learned the classification problem, the precision-recall curve and the precision and recall for different threshold values. It ends with the confusion matrix, which shows how the results were classified.

Dataset Collection and Description

First and foremost, model inputs must be selected at the start of the phase to learn and produce the final structure based on them. The dataset used to evaluate the model was compiled from historical data that includes flight schedule information for three months, from October to December 2000. Table 1 lists the variables that are utilized as inputs. There are 23 factors in the original dataset. However, certain factors were eliminated since they were irrelevant to the desired outcome. The data was provided by a real-world source from the United States (US) Bureau of Transportation Statistics.

Table 1

Dataset Features

S/No.	Input	Reference	Type
1	Origin/Destination airport	3 letters format, e.g. LAX -Los Angeles International Airport	String
2	Planned Departure and Arrival Time	3/4 digits format, e.g. 1520 – 03:20pm	Integer
3	Flight Number	4 digits format, 1451	Integer
4	Real Departure and Arrival Time	3/4 digits format, e.g. 1520 – 03:20pm	Integer
5	Unique Carrier	2/3 letters format, e.g. PS	String
6	Elapse Time	2 digits format, e.g. 94	Integer
7	Departure/Arrival Delay	2 digits format, e.g. 11	Integer

Figure 4

Multilayer Perceptron Autoencoder

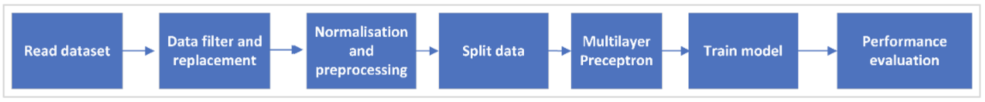
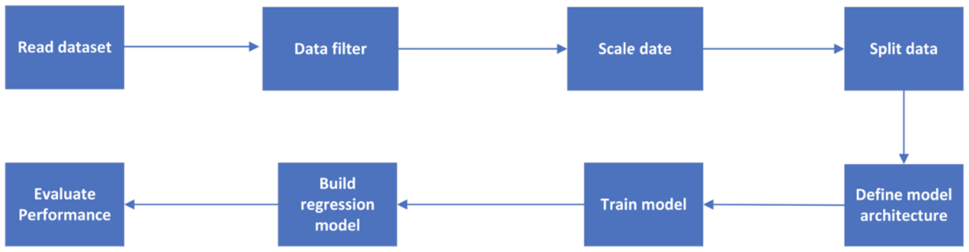


Figure 5

Logistic Regression Autoencoder



Dataset Pre-Processing

We cleaned the data by removing the ”missing values” and replacing them with zeroes, while the inputs with less than 15 minutes of delay time and other irrelevant labels were dropped. The ”Delayed” label was created to track late flights for arrival and departure flights.

Software Settings for Experiment

The implementation was done in Jupyter Notebook using Python programming language. It is a free web programme that lets users create and share scripts with live software, equations, visualizations, and texts. Data cleaning and transformation, statistical modelling, numerical simulation, data visualization, machine learning, and other applications are just a few of the potential use of the tool (Ayoubi et al., 2018; Bisandu et al., 2021; Borse et al., 2020; Holguín-Veras et al., 2012; Loughran & Elliott, 2022; Orimoloye et al., 2020; Petersen et al., 2008; Shabanpour et al., 2022; Sharifzadeh et al., 2016; Wei et al., 2014; Wu et al., 2022; Yap et al., 2019). The relevant libraries used in the script can be found in Table 2.

Table 2

Libraries

S/No.	Library	Version
1	Keras	2.4.3
2	Numpy	1.19.2
3	Pandas	1.1.3
4	TensorFlow	2.5.0
5	Itertools	0.25.1
6	Matplotlib	3.3.2
7	Sklearn	0.23.2

Evaluation Metrics

Accuracy, precision, and recall are the most important metrics for evaluating the performance of categorization machine learning models. A variety of indicators were used to examine the effectiveness of the outcomes. Several notions must be introduced, such as True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) in computing the model performance. The simplest intuitive performance metric is accuracy, the ratio of accurately predicted observations to all observations as shown in Equation 7.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{7}$$

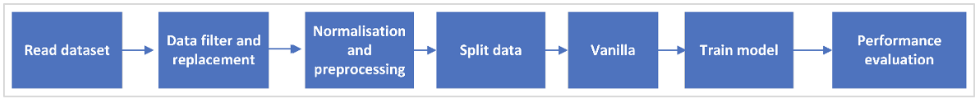
The ratio of accurately predicted positive observations to total expected positive observations is precision, as shown in Equation 8.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{8}$$

The ratio of accurately predicted positive data to all findings in the actual class is known as recall, as shown in Equation 9.

Figure 6

Vanilla Autoencoder



$$\text{Recall} = \frac{TP}{TP + FN} \tag{9}$$

Precision-Recall is a helpful metric of prediction success when the classes are severely unbalanced. Precision measures result from usefulness in data collection, while recall indicates how many properly relevant results are retrieved.

For various thresholds, the precision-recall curve depicts the trade-off between them. A large area below the curve indicates good recall and precision, with high precision indicating a low false-positive rate and high recall indicating a low false-negative rate. Both indicate that the classification produces precise results (Y. Chen et al., 2017; Guo et al., 2021; Y. Li et al., 2018; Ma et al., 2015; Piao et al., 2023; Xiao et al., 2014).

Results and Discussion

The program is written in Python Programming language, running on local machines or Crescent HPC Cluster. On Crescent, it operates on GPU nodes, which have two Intel E5-2698 v3 CPUs with 32 CPU cores, having a total of 256GB of shared memory.

The experiments were repeated with varied input settings to see how the accuracy and loss changed and get the best prediction possible. We evaluated the performance of Multilayer perceptron autoencoder, logistic regression autoencoder and vanilla autoencoder in classifying and predicting flight delays to determine which strategy performs best in these situations. A Precision-Recall graph was used to evaluate and determine the quality of the results, as the classes are imbalanced. When analysing probabilistic predictions, plots from the charts are generated and used to analyse the performance trade-off for various threshold values.

Vanilla Autoencoder

Table 3 compares the performance of the flight delay classification and prediction based on different input settings in terms of area under the precision-recall curve, accuracy, precision, and recall. The characteristics were analysed at various levels of model development with the different amount of input dimensions. The first experiment was evaluated with 3 hidden layers, 33 input dimensions, and 100 epochs, resulting in 42.07% accuracy and 19.99% precision. The hidden layers varied from 3 to 10, with an input dimension from 33 to 500 and epochs from 100 to 300. Increasing the number of epochs and adding one layer led to almost the same results, giving an accuracy of 55.00%, precision of 27.03% and a recall of 63.57%. It leads to a recall of 55.92%, showing that half of the relevant data were considered. However, there is no mention of how many unnecessary data were recovered, as seen in the accuracy and precision results.

Table 3

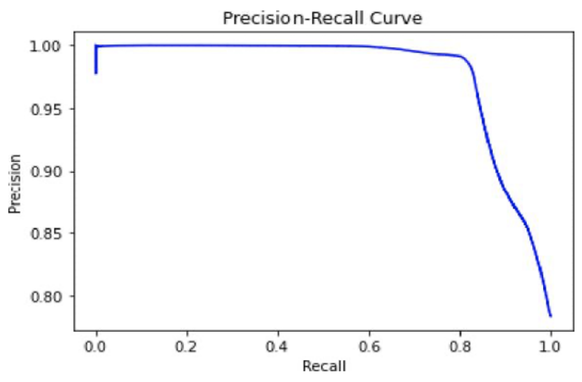
Performance Comparison of Vanilla Autoencoder

Hidden Layers	Input Dimensions	Epochs	Area	Accuracy	Precision	Recall
10	40	300	0.9773	0.8036	0.5246	0.9878
10	100	300	0.9462	0.6712	0.3883	0.9037
5	40	200	0.9846	0.5651	0.3320	0.9990
5	500	200	0.9527	0.7652	0.4691	0.6485
4	33	200	0.8527	0.5500	0.2703	0.6357
3	33	100	0.7475	0.4207	0.1999	0.5592

Figure 7 represents the precision-recall curve, a plot of precision and recall for different thresholds.

Figure 7

Precision-Recall Curve for 10 Layers and 300 Epochs



The output for the model is constructed with one input layer, 10 hidden layers, 40 input dimensions for each layer, an output layer, which is also the decoder of the autoencoder, and 300 epochs. The activation function used for each hidden layer was Sigmoid. The curve produces the expected form. Precision is high at thresholds with poor recall and begins to diminish at thresholds with high recall. The high area under the curve shows high recall and precision, highlighting the low false-negative and high false-positive rates, as seen in Figure 8. The average score for precision demonstrates that the classifier produces almost accurate results, while the high recall illustrates that the most positive results had been returned.

Figure 8

Confusion Matrix for 10 Layers and 300 Epochs

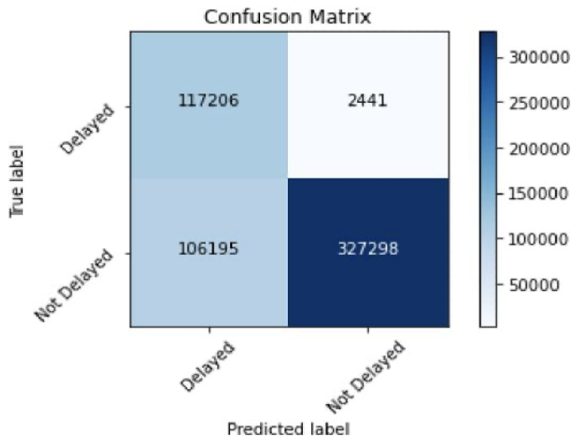


Figure 8 shows how the model accurately classifies 97.95% of delayed flights and 75.50% of non-delayed aircraft. There are almost 600,000 flight records in this classification. The model accurately identifies 117,206 delayed flights and 327,298 non-delayed flights, which is impressive for the delay prediction software. Figure 9 represents the precision and recall resulting from the previous model. The black line represents the precision, while recall can be seen in blue.

According to the precision equation, reducing the classifier's threshold raises the denominator by expanding the number of outcomes returned. The threshold was set to 0.5 as too high; the outputs may all be true-positive, increasing precision. If the prior threshold were too low, reducing it further would have generated false positives, reducing precision.

Logistic Regression Autoencoder

Table 4 compares the performance of the autoencoder that assembles a logistic regression model built on different input settings in terms of area under precision-recall curve, accuracy, precision, and recall.

Figure 9

Precision and Recall for Different Thresholds

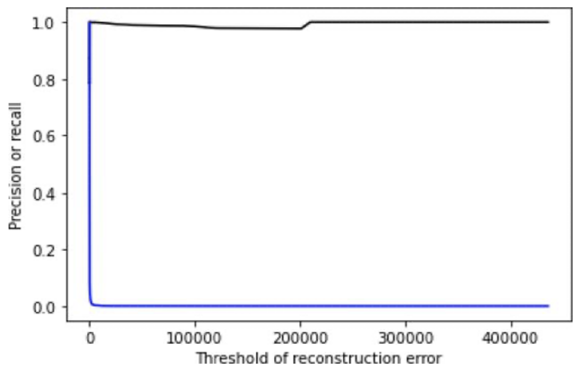


Table 4

Performance Comparison for Logistic Regression

Hidden Layer	Input Dimension	Epochs	Area	Accuracy	Precision	Recall
17	100/50/25	300	0.4184	0.7477	0.7030	0.9784
11	100/50/25/12/6	30	0.6384	0.7669	0.7805	0.5536
11	100/50/25/12/6	300	0.4229	0.7851	0.7318	0.9936

The characteristics were analysed at various levels of model development with the different amount of input dimensions. The hidden layers used to implement the autoencoders are 10 to 17, an input dimension of 6 to 100 and epochs of 30 to 300. This implementation uses the same number of hidden layers in the decoder as in the encoder, the last being the output layer, making it an asymmetric structure. The first experiment was evaluated with 11 hidden layers and 30 epochs, resulting in 76.69% accuracy and 78.05% precision. The recall of 55.36% shows that half of the relevant data were considered. However, there is no mention of how many unnecessary data were recovered. The precision and recall graph area were the highest for this implementation.

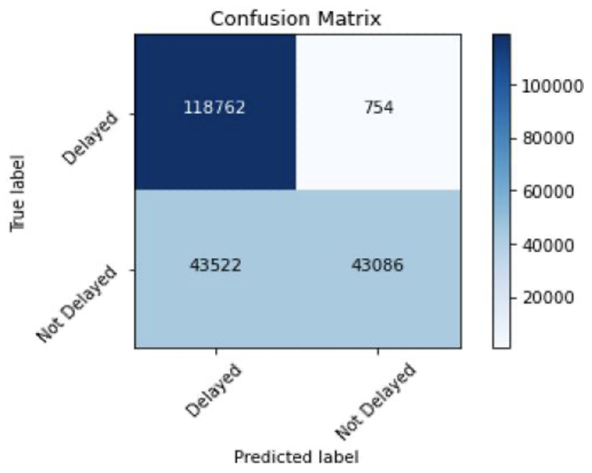
Increasing the number of epochs by 10 times and adding 6 more layers led to a recall of 97.84%, giving an accuracy of 74.77% and precision of 70.30%. The best result was achieved with 6 hidden layers for the encoder and 5 for the decoder. Inputting a total of 300 epochs, a recall of 99.36% was gained, with an accuracy of 78.51% and a precision of 73.18%.

Figure 10 shows how the model accurately classifies 99.36% of delayed flights and 49.74% of non-delayed aircraft. This classification has almost 200,000 flight records, and the model accurately identifies 118,762 delayed flights and 43,086 non-delayed flights.

Figure 11 depicts the logistic regression model's training and test loss. It was found that the model works

Figure 10

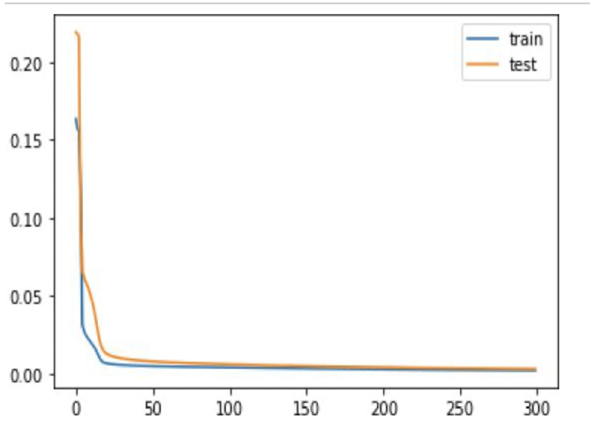
Confusion Matrix for 11 Layers and 300



well during the training and test phases, with consistently great results over the whole training set, which is a major benefit when designing a flight delay classification model.

Figure 11

Model Loss for 17 Layers and 300 Epochs



Multilayer Perceptron Autoencoder

Table 5 compares the performance of the autoencoder Multilayer perceptron autoencoder. Here, after fitting the model, the rebuilding component of the model can be neglected, and the model can be used up until the bottleneck. The model produces a fixed-length vector with a compressed version of the input data at the bottleneck. The characteristics were analysed at various levels of model development with the different amount of input dimensions. The hidden layers used to implement the autoencoders are 5 to 10, with an input dimension of 11, with epochs from 30 to 100. The best result for this case was achieved with 3 hidden layers for the encoder, one hidden layer for the bottleneck and a decoder. Inputting a total of 30 epochs, a recall of 96.19% was gained, with an accuracy of 60.27% and a precision of 59.79%. In this implementation, it can be observed that if the number of epochs increases, the recall begins to drop significantly.

Table 5

Performance Comparison for Multilayer Perceptron

Hidden Layers	Input Dimensions	Epochs	Area	Accuracy	Precision	Recall
6	11	30	0.4841	0.5239	0.5778	0.6662
7	11	30	0.5121	0.6027	0.5979	0.9619
7	11	100	0.4905	0.5641	0.6320	0.5951
5	11	100	0.4857	0.5800	0.5558	0.1821

Figure 12 represents the precision-recall curve for the model above. The activation function used for each hidden layer was Sigmoid, while the activation function called 'linear' was used for the output layer. The curve produces the expected form. The area under the curve shows high recall and average precision, highlighting the low false-negative and high false-positive rates. Precision is high at thresholds with poor recall and begins to diminish at thresholds with high recall.

Figure 12

Precision-Recall Curve For 7 Layers And 30 Epochs

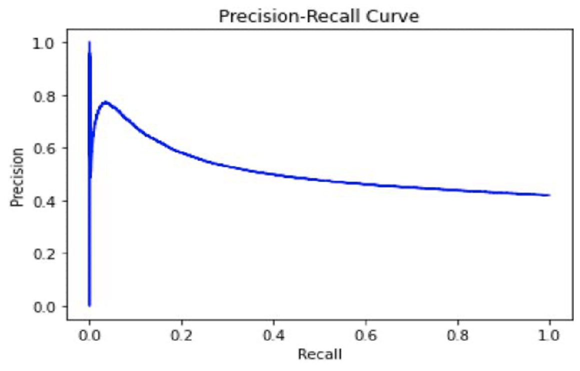


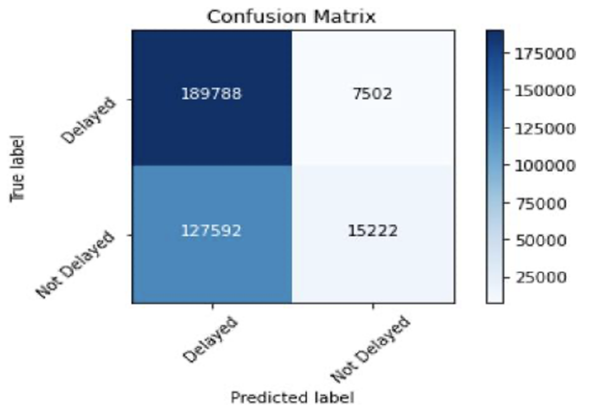
Figure 13 shows how the model accurately classifies 96.19% of delayed flights and 10.65% of non-delayed aircraft. This classification has almost 350,000 flight records, and the model accurately identifies 189,788 delayed flights and 15,222 non-delayed flights.

Comparative Discussion

In this research, the accuracy, precision and recall for the measures of the Vanilla autoencoder are 80.36%, 52.46% and 98.78%. The output for the logistic regression autoencoder is 78.51% accuracy, 73.18% precision and

Figure 13

Confusion Matrix for 7 Layers and 30 Epochs



99.36% recall, while the Multilayer perceptron autoencoder is 60.27%, 59.79%, 96.19%, as shown in Table 6. Therefore, the Vanilla autoencoder outperformed the other autoencoders. The main interest is in the autoencoder's output, so the focus was on the other three methods. The Multilayer perceptron for the classification method did not perform as expected.

The advantage of the vanilla autoencoder implementation is that almost 600,000 flight records were analysed in the classification, from which 97.95% of delayed flights and 75.50% of non-delayed flights were classified accurately. The implementation reveals a massive improvement when using a deep autoencoder, such as vanilla, having at least 200,000 more flight records analysed in the classification and successfully classifying over 25% of non-delayed aircraft compared to the logistic regression model. A method with numerous parameters to fit during training can produce different outcomes in classification tasks than other methods. The area gives another aspect where the vanilla autoencoder outperforms the other methods under the precision-recall curve. The wide area of 0.9773 below the curve denotes good recall and precision, with high precision implying a low false-positive rate and high recall indicating a low false-negative rate, compared to 0.4229 for logistic regression and 0.5121 for Multilayer perceptron.

Logistic regression autoencoder comes into an advantage with the loss function analysis. The experiment revealed a massive improvement from the precision point of view, having at least 20% compared to the other methods. It was found that the model works well during the training and test phases, with consistently great results over the whole training set, which is a major benefit when designing a flight delay classification model.

Table 6

Performance Comparisons of the Different Autoencoders

Model	Accuracy	Precision	Recall
Vanilla	0.8036	0.5246	0.9878
Logistic regression	0.7851	0.7318	0.9936
Multilayer perceptron	0.6027	0.5979	0.9619

Conclusion and Future Direction

Predicting flight delays is a fascinating study issue that has gotten considerable attention recently. Most studies have attempted to create and expand their models to improve the accuracy and precision of flight delay prediction. Because on-time flights are critical, flight delay prediction algorithms must be exact and accurate. In this study, we proposed implementing three approaches to an autoencoder: vanilla autoencoder, logistic regression autoencoder, and multilayer perceptron autoencoder. We compared the predicted accuracy, precision and recall of the deep feedforward neural network with the autoencoder approaches at different variable tunings and epochs. Our primary aim was to classify and predict flight delays using an autoencoder and re-scale its parameters. Our experimental results show that the deep vanilla autoencoder outperformed the other two implementations at different parameter adjustments. Comparing the three autoencoder models shows that the vanilla autoencoder's accuracy is greater by 1.85% than the logistic regression autoencoder and by 20.09% than the Multilayer perceptron for classification. At the same time, the area under the precision-recall curve is higher by 0.5744 and 0.4852; using the vanilla autoencoder to optimize the results positively affects classifying and predicting the delay forecasting. Flight delays are a popular issue because of their economic and environmental consequences. They may raise customer costs as well as airline operating costs. Aside from direct passenger effects, delay prediction is critical for every stakeholder in the air transport industry during the decision-making phase. This work will assist the aviation industry, especially the air transportation sector, enhance passenger experiences by improving flight delay decision support. The next step would be to test the correctness of this method on different datasets or sample data.

The practical implications of accurate flight delay prediction extend well beyond academic interest, touching the very core of air transportation logistics, customer satisfaction, and operational efficiency. In the real world, the ability to predict flight delays with high precision enables airlines and airports to optimize their schedules, manage resources more effectively, and improve overall service quality. It also allows passengers to make informed decisions regarding their travel plans, potentially minimizing the inconvenience caused by delays. By offering a method that surpasses others in accuracy and precision, our approach to using a vanilla autoencoder for delay prediction provides a significant advantage. This advantage is crucial for developing responsive and flexible operational strategies that can adapt to the unpredictable nature of air travel, thereby reducing economic losses and enhancing passenger experiences.

Moreover, our methodology stands out because it not only focuses on the predictive accuracy but also emphasizes the importance of model adaptability to various data environments. The improved performance of our vanilla autoencoder model, as demonstrated through rigorous testing across different parameter adjustments, showcases its robustness and reliability in predicting flight delays under diverse conditions. This reliability is vital for real-world applications, where the stakes of accurate predictions are high, and the cost of errors can be substantial. Our research paves the way for further advancements in flight delay prediction technologies, encouraging the adoption of more sophisticated Machine Learning techniques in the aviation industry. Future investigations could explore the integration of additional data sources, such as social media sentiment analysis or real-time weather updates, to further refine the accuracy of delay predictions. Ultimately, our work contributes to a broader understanding of how Machine Learning can be harnessed to address complex challenges in aviation, setting a new benchmark for excellence in the field.

Acknowledgement & Funding

The Petroleum Trust Development Fund (PTDF) Nigeria partially funded this work through grant number PTDF/ED/OSS/PHD/DBB/1558/19 for one of the first author's Ph.D. studies. We also thank the UKRI for the COVID-19 recovery grant under the budget code SA077N. This research was heavily affected by the COVID-19 pandemic during the first authors' Ph.D. studies. This led to an extension to registration for 3 months, which was funded by the UKRI doctoral extension recovery grant.

Authorship Contribution Statement

Desmond Bala Bisandu, and Dan A. Soviani-Sitoiu: Conceptualisation, Methodology, Investigation, Data Curation, Writing – original draft.

Irene Moulitsas: Supervision, Project administration, Funding acquisition, Writing – Review & Editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have influenced the work reported in this paper.

Data Availability

All data used in this research was obtained from the United States Bureau of Transportation Statistics website.

References

Aigner, W., Miksch, S., Müller, W., Schumann, H., & Tominski, C. (2007). Visualizing time-oriented data-a systematic view. *Computers and Graphics (Pergamon)*, *31*(3), 401–409. https://doi.org/10.1016/j.cag.2007.01.030

Alkhayrat, M., Aljnidi, M., & Aljoumaa, K. (2020). A comparative dimensionality reduction study in telecom customer segmentation using deep learning and pca. *Journal of Big Data*, *7*(9), 1–23. https://doi.org/10.1186/s40537-020-0286-0

Ayoubi, S., Limam, N., Salahuddin, M. A., Shahriar, N., Boutaba, R., Estrada-Solano, F., & Caicedo, O. M. (2018). Machine learning for cognitive network management. *IEEE Communications Magazine*, *56*(1), 158–165. https://doi.org/10.1109/MCOM.2018.1700560

Balakrishna, P., Ganesan, R., & Sherry, L. (2008). Airport taxi-out prediction using approximate dynamic programming: Intelligence-based paradigm. *Transportation Research Record*, *2052*(1), 54–61. https://doi.org/10.3141/2052-07

Balakrishna, P., Ganesan, R., & Sherry, L. (2009). Application of reinforcement learning algorithms for predicting taxi-out times. *Proceedings of the 8th USA/Europe Air Traffic Management Research and Development Seminar (ATM 2009)*, 255–261. https://catsr.vse.gmu.edu/pubs/ATM2009_TaxiOut.pdf

Balakrishna, P., Ganesan, R., Sherry, L., & Levy, B. S. (2008). Estimating taxi-out times with a reinforcement learning algorithm. *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, 1–12. https://doi.org/10.1109/DASC.2008.4702812

Belcastro, L., Marozzo, F., Talia, D., & Trunfio, P. (2016). Using scalable data mining for predicting flight delays. *ACM Transactions on Intelligent Systems and Technology*, *8*(1), 1–20. https://doi.org/10.1145/2888402

Bhadra, D. (2009). You (expect to) get what you pay for: A system approach to delay, fare, and complaints. *Transportation Research Part A: Policy and Prac-*

tice, *43*(9-10), 829–843. https://doi.org/10.1016/j.tra.2009.08.003

Bisandu, D. B., Homaid, M. S., Moulitsas, I., & Filippone, S. (2021). A deep feedforward neural network and shallow architectures effectiveness comparison: Flight delays classification perspective. *ICAAI '21: Proceedings of the 5th International Conference on Advances in Artificial Intelligence*, 1–10. https://doi.org/10.1145/3505711.3505712

Bisandu, D. B., & Moulitsas, I. (2022). A bidirectional deep lstm machine learning method for flight delay modelling and analysis. *The National Training Aircraft Symposium (NTAS)*. https://commons.erau.edu/ntas/2022/presentation/18/

Bisandu, D. B., & Moulitsas, I. (2023). A hybrid ensemble machine learning approach for arrival flight delay classification prediction using voting aggregation technique. *AIAA AVIATION 2023 Forum*. https://doi.org/10.2514/6.2023-4326

Bisandu, D. B., & Moulitsas, I. (2024). Prediction of flight delay using deep operator network with gradient-mayfly optimisation algorithm. *Expert Systems With Applications*, *247*. https://doi.org/10.1016/j.eswa.2024.123306

Bisandu, D. B., Moulitsas, I., & Filippone, S. (2022). Social ski driver conditional autoregressive-based deep learning classifier for flight delay prediction. *Neural Computing and Applications*, *34*(11), 8777–8802. https://doi.org/10.1007/s00521-022-06898-y

Borse, Y., Jain, D., Sharma, S., Vora, V., & Zaveri, A. (2020). Flight delay prediction system. *International Journal of Engineering Research & Technology (IJERT)*, *9*(3), 88–92. https://doi.org/10.17577/IJERTV9IS030148

Cai, K. Q., Zhang, J., Xiao, M. M., Tang, K., & Du, W. B. (2017). Simultaneous optimization of airspace congestion and flight delay in air traffic network flow management. *IEEE Transactions on Intelligent Transportation Systems*, *18*(11), 3072–3082. https://doi.org/10.1109/TITS.2017.2673247

Campanelli, B., Fleurquin, P., Eguíluz, V. M., Ramasco, J. J., Arranz, A., Extebarria, I., & Ciruelos, C. (2014). Modeling reactionary delays in the european air transport network. *SIDs 2014: Fourth SESAR Innovation Days*. https://sesarju.eu/sites/default/files/documents/sid/2014/SID%202014-44.pdf

Carvalho, L., Sternberg, A., Maia Gonçalves, L., Beatriz Cruz, A., Soares, J. A., Brandão, D., Carvalho, D., & Ogasawara, E. (2021). On the relevance of data science for flight delay research: A systematic review. *Transport Reviews*, *41*(4), 499–528. https://doi.org/10.1080/01441647.2020.1861123

Castaing, J., Mukherjee, I., Cohn, A., Hurwitz, L., Nguyen, A., & Müller, J. J. (2016). Reducing airport gate blockage in passenger aviation: Models and analysis. *Computers and Operations Research*, *65*, 189–199. https://doi.org/10.1016/j.cor.2014.02.011

Chen, J., De Laurentis, D., & Sun, D. (2016). Dynamic stochastic model for converging inbound air traffic. *Journal of Guidance, Control, and Dynamics*, *39*(10), 2273–2283. https://doi.org/10.2514/1.G001379

Chen, Y., Shu, L., & Wang, L. (2017). Traffic flow prediction with big data: A deep learning based time series model. *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. https://doi.org/10.1109/INFCOMW.2017.8116535

Ciruelos, C., Arranz, A., Etxebarria, I., Peces, S., Campanelli, B., Fleurquin, P., Eguiluz, V. M., & Ramasco, J. J. (2015). Modelling delay propagation trees for scheduled flights. *The 11th USA/Europe Air Traffic Management Research and Development Seminar*.

Dhanawade, R., Deo, M., Khanna, N., & Deolekar, R. V. (2019). Analyzing factors influencing flight delay prediction. In M. Hoda (Ed.), *2019 6th international conference on computing for sustainable global development (indiacom)*. https://ieeexplore.ieee.org/document/8991208

Ganesan, R., Balakrishna, P., & Sherry, L. (2010). Improving quality of prediction in highly dynamic environments using approximate dynamic programming. *Quality and Reliability Engineering International*, *26*(7), 717–732. https://doi.org/10.1002/qre.1127

Gopalakrishnan, K., & Balakrishnan, H. (2017). A comparative analysis of models for predicting delays in air traffic networks. *12th USA/Europe Air Traffic Management Research and Development Seminar 2017*. https://www.mit.edu/~hamsa/pubs/GopalakrishnanBalakrishnanATM2017.pdf

Guo, Z., Hao, M., Yu, B., & Yao, B. (2022). Detecting delay propagation in regional air transport systems using convergent cross mapping and complex network theory. *Transportation Research Part E: Logistics and Transportation Review*, *157*, 102585. https://doi.org/10.1016/j.tre.2021.102585

Guo, Z., Yu, B., Hao, M., Wang, W., Jiang, Y., & Zong, F. (2021). A novel hybrid method for flight departure delay prediction using random forest regression and maximal information coefficient. *Aerospace Science and Technology*, *116*. https://doi.org/10.1016/j.ast.2021.106822

Gupta, H. (2018). Evaluating service quality of airline industry using hybrid best worst method and vikor. *Journal of Air Transport Management*, *68*, 35–

47. https://doi.org/10.1016/j.jairtraman.2017.06.001

Holguín-Veras, J., Xu, N., & Bhat, C. (2012). An assessment of the impacts of inspection times on the airline industry's market share after september 11th. *Journal of Air Transport Management*, *23*, 17–24. https://doi.org/10.1016/j.jairtraman.2012.02.004

Hondet, G., Delgado, L., & Gurtner, G. (2018). Airline disruption management with aircraft swapping and reinforcement learning. *SIDS 2018: 8th SESAR Innovation Days*. https://www.sesarju.eu/sites/default/files/documents/sid/2018/papers/SIDs_2018_paper_45.pdf

Kim, Y. J., Choi, S., Briceno, S., & Mavris, D. (2016). A deep learning approach to flight delay prediction. *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC) Proceedings*. https://doi.org/10.1109/DASC.2016.7778092

Le Ny, J., & Balakrishnan, H. (2011). Feedback control of the national airspace system. *Journal of Guidance, Control, and Dynamics*, *34*(3), 832–846. https://doi.org/10.2514/1.51203

Li, Q., & Jing, R. (2022). Flight delay prediction from spatial and temporal perspective. *Expert Systems with Applications*, *205*. https://doi.org/10.1016/j.eswa.2022.117662

Li, Y., Yin, G., Zhuang, W., Zhang, N., Wang, J., & Geng, K. (2018). Compensating delays and noises in motion control of autonomous electric vehicles by using deep learning and unscented kalman predictor. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *50*(11), 4326–4338. https://doi.org/10.1109/TSMC.2018.2850367

Loughran, K., & Elliott, J. R. (2022). Unequal retreats: How racial segregation shapes climate adaptation. *Housing Policy Debate*, *32*(1), 171–189. https://doi.org/10.1080/10511482.2021.1931928

Ma, X., Tao, Z., Wang, Y., Yu, H., & Wang, Y. (2015). Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transportation Research Part C: Emerging Technologies*, *54*, 187–197. https://doi.org/10.1016/j.trc.2015.03.014

Manna, S., Biswas, S., Kundu, R., Rakshit, S., Gupta, P., & Barman, S. (2018). A statistical approach to predict flight delay using gradient boosted decision tree. *ICCIDS 2017 - International Conference on Computational Intelligence in Data Science, Short Proceedings*. https://doi.org/10.1109/ICCIDS.2017.8272656

Mizufune, K., & Katsumata, S. (2019). Joint classification model of topic and polarity: Finding satisfaction and dissatisfaction factors from airport service review. *Proceedings - 2018 IEEE International Conference on Data Mining Workshops, ICDMW 2018*, 856–863. https://doi.org/10.1109/ICDMW.2018.00126

Mofokeng, T. J., & Marnewick, A. (2017). Factors contributing to delays regarding aircraft during a-check maintenance. *2017 IEEE Technology and Engineering Management Society Conference, TEMSCON 2017*, 185–190. https://doi.org/10.1109/TEMSCON.2017.7998375

Muros Anguita, J. G., & Díaz Olariaga, O. (2023). Prediction of departure flight delays through the use of predictive tools based on machine learning/deep learning algorithms. *Aeronautical Journal*, *18*(1), 1–23. https://doi.org/10.1017/aer.2023.41

Ogunsina, K., & Okolo, W. A. (2021). Artificial neural network modeling for airline disruption management. *Journal of Aerospace Information Systems*, *19*(5), 1–27. https://doi.org/10.2514/1.i011018

Orimoloye, L. O., Sung, M. C., Ma, T., & Johnson, J. E. V. (2020). Comparing the effectiveness of deep feedforward neural networks and shallow architectures for predicting stock price indices. *Expert Systems with Applications*, *139*. https://doi.org/10.1016/j.eswa.2019.112828

Osorio, C. (2019). Dynamic origin-destination matrix calibration for large-scale network simulators. *Transportation Research Part C: Emerging Technologies*, *98*, 186–206. https://doi.org/10.1016/j.trc.2018.09.023

Pejovic, T., Williams, V. A., Noland, R. B., & Toumi, R. (2009). Factors affecting the frequency and severity of airport weather delays and the implications of climate change for future delays. *Transportation Research Record*, *2139*(1), 97–106. https://doi.org/10.3141/2139-12

Petersen, K., Feldt, R., Mattsson, M., & Mujtaba, S. (2008). Systematic mapping studies in software engineering. *12th International Conference on Evaluation and Assessment in Software Engineering (EASE)*, 1–10. https://doi.org/10.14236/ewic/EASE2008.8

Pfeil, D. M., & Balakrishnan, H. (2012). Identification of robust terminal-area routes in convective weather. *Transportation Science*, *46*(1), 56–73. https://doi.org/10.1287/trsc.1110.0372

Pfeil, D. M., Balakrishnan, H., Zou, B., Hansen, M., Zonglei, L., Jiandong, W., Guansheng, Z., Zhong, Z. W., Varun, D., Lin, Y. J., Zhang, W., Kamgarpour, M., Sun, D., Tomlin, C. J., Rong, Y., Wang, J., Ding, J., Xu, Y., Dalmau, R., & Babić, O. (2008). A recursion-based approach to simulating airline schedule robustness. *Journal of Air Transport Management*, *2*(1), 1–11. https://doi.org/10.1016/j.jairtraman.2006.07.004

Piao, C., Wang, N., & Yuan, C. (2023). Rebalance weights adaboost-svm model for imbalanced data. *Com-*

putational Intelligence and Neuroscience. https://doi.org/10.1155/2023/4860536

Proenca, H. M., Klijn, R., Bäck, T., & Van Leeuwen, M. (2019). Identifying flight delay patterns using diverse subgroup discovery. In S. Sundaram (Ed.), *Proceedings of the 2018 ieee symposium series on computational intelligence (ssci 2018)* (pp. 60–67). https://doi.org/10.1109/SSCI.2018.8628933

Rebollo, J. J., & Balakrishnan, H. (2014). Characterization and prediction of air traffic delays. *Transportation Research Part C: Emerging Technologies*, *44*, 231–241. https://doi.org/10.1016/j.trc.2014.04.007

Rodríguez-Sanz, Á., Cano, J., & Rubio Fernández, B. (2022). Impact of weather conditions on airport arrival delay and throughput. *Aircraft Engineering and Aerospace Technology*, *94*(1), 60–78. https://doi.org/10.1108/AEAT-12-2020-0318

Rong, Y., Wang, J., & Ding, J. (2009). Ria-based visualization platform of flight delay intelligent prediction. In Q. Luo, J. Yi, & C. Bin (Eds.), *2009 second isecs international colloquium on computing, communication, control, and management (cccm 2009)* (pp. 94–97, Vol. 4). https://doi.org/10.1109/CCCM.2009.5267976

Saadat, M. N., & Moniruzzaman, M. (2019). Enhancing airlines delay prediction by implementing classification based deep learning algorithms. In S. Lee, R. Ismail, & H. Choo (Eds.), *Proceedings of the 13th international conference on ubiquitous information management and communication (imcom) 2019* (pp. 886–896). Springer International Publishing.

Shabanpour, N., Razavi-Termeh, S. V., Sadeghi-Niaraki, A., Choi, S. M., & Abuhmed, T. (2022). Integration of machine learning algorithms and gis-based approaches to cutaneous leishmaniasis prevalence risk mapping. *International Journal of Applied Earth Observation and Geoinformation*, *112*. https://doi.org/10.1016/j.jag.2022.102854

Sharifzadeh, S., Chiotellis, I., Triebel, R., & Cremers, D. (2016). Learning to drive using inverse reinforcement learning and deep q-networks. *Conference on Neural Information Processing Systems (NIPS 2016)*. https://arxiv.org/pdf/1612.03653

Tan, Y. E., Teong, K. S., Shabbir, M., Foo, L. K., & Chua, S. L. (2018). Modelling flight delays in the presence of class imbalance. *AICC '18: Proceedings of the 2018 Artificial Intelligence and Cloud Computing International Conference*, 186–191. https://doi.org/10.1145/3299819.3299847

Venkatesh, V., Arya, A., Agarwal, P., Lakshmi, S., & Balana, S. (2017). Iterative machine and deep learning approach for aviation delay prediction. *2017 4th IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics (UPCON)*. https://doi.org/10.1109/UPCON.2017.8251111

Wei, X., Wagner, M., Christiano, E. R. A., Shattuck, P., & Yu, J. W. (2014). Special education services received by students with autism spectrum disorders from preschool through high school. *Journal of Special Education*, *48*(3), 167–179. https://doi.org/10.1177/0022466913483576

Wu, Y., Mei, G., & Shao, K. (2022). Revealing influence of meteorological conditions and flight factors on delays using xgboost. *Journal of Computational Mathematics and Data Science*. https://doi.org/10.1016/j.jcmds.2022.100030

Xiao, Y., Liu, J. J., Hu, Y., Wang, Y., Lai, K. K., & Wang, S. (2014). A neuro-fuzzy combination model based on singular spectrum analysis for air transport demand forecasting. *Journal of Air Transport Management*, *39*, 1–11. https://doi.org/10.1016/j.jairtraman.2014.03.004

Yap, M., Luo, D., Cats, O., van Oort, N., & Hoogendoorn, S. (2019). Where shall we sync? clustering passenger flows to identify urban public transport hubs and their key synchronization priorities. *Transportation Research Part C: Emerging Technologies*, *98*, 433–448. https://doi.org/10.1016/j.trc.2018.12.013

Yazdi, M. F., Kamel, S. R., Chabok, S. J. M., & Kheirabadi, M. (2020). Flight delay prediction based on deep learning and levenberg-marquart algorithm. *Journal of Big Data*, *7*(1), 1–28. https://doi.org/10.1186/s40537-020-00380-z

Ye, B., Liu, B., Tian, Y., & Wan, L. (2020). A methodology for predicting aggregate flight departure delays in airports based on supervised learning. *Sustainability*, *12*(7). https://doi.org/10.3390/su12072749

Yu, B., Guo, Z., Asian, S., Wang, H., & Chen, G. (2019). Flight delay prediction for commercial air transport: A deep learning approach. *Transportation Research Part E: Logistics and Transportation Review*, *125*, 203–221. https://doi.org/10.1016/j.tre.2019.03.013

Yu, Y., Chen, H., Yuan, L., & Zhang, B. (2021). Flight delay classification warning based on evolutionary under-sampling bagging ensemble learning. In Y. Xing (Ed.), *Proceedings of the fifth international conference on traffic engineeering and transportation systems (ictets 2021)*. https://doi.org/10.1117/12.2619725