Learning and Optimization for Reliable Airline Operations

by

Aniket Biswal

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Management Sciences

Waterloo, Ontario, Canada, 2024

© Aniket Biswal 2024

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

I am the sole author of Chapters 1 and 4

Chapter 2 of this thesis has been incorporated within a paper [16] that is submitted for publication. The paper is co-authored by myself and my supervisors, Dr. Sirisha Rambhatla and Dr. Fatma Gzara

Chapter 3 of this thesis has been incorporated within a paper [15] that will be submitted to publication. The paper is co-authored by myself and my supervisors, Dr. Sirisha Rambhatla and Dr. Fatma Gzara

I would like to acknowledge Aditya Shridhar, an undergraduate research assistant, for his assistance in running the training for the survival model and the initial version of the simulation setup.

Abstract

The aviation industry has experienced significant volatility and complexity in recent years, exacerbated by unpredictable factors such as adverse weather, airport congestion, and mechanical problems. This research aims to answer two key questions for improving airline operations: First, how can we create realistic schedules? To address this, we need to reliably predict flight segments. Second, how can we reduce costs while operating under uncertainty? One effective approach is by building robust schedules for crews. By tackling these questions, this thesis addresses two critical problems in airline operations: block time prediction and reliable crew pairing optimization. These problems are particularly challenging due to the long-term nature of prediction and planning, where limited knowledge about future conditions necessitates dealing with significant levels of uncertainty.

We first look at flight segment prediction through the lens of *block-time* – the total time from when an aircraft leaves the gate at the departure airport to when it arrives at the gate at the destination airport. Here we introduce an embedding-based neural network model to improve the long-term predictive accuracy of block time, which is essential for optimizing resource utilization and effective planning. Traditional methods in the literature often fail to capture the complex latent relationships between factors affecting block time and struggle with high-cardinality categorical data and temporal variations. Our large-scale study on the 2018 Bureau of Transportation Statistics (BTS) dataset incorporates high cardinality categorical features through embedding-based representation learning. The model's efficacy is demonstrated through rigorous experimental evaluations against ten baselines, showing significant improvements in long-term forecast accuracy for airline planning.

While flight segment prediction provides extensive knowledge about delays, it cannot be directly used to determine if a crew will successfully make a connection. This is because planning needs information about the probability that a crew can make a connection. Therefore, in the second study we introduce `SurvCG`, which integrates survival analysis into the crew pairing problem to quantify and manage uncertainties in flight schedules. Traditional approaches often assume deterministic scenarios or rely on predefined uncertainty sets that do not adequately reflect the true variability and unpredictability inherent in airline operations. `SurvCG` overcomes these limitations by incorporating survival probabilities into the optimization process, thus introducing a dynamic reliability score into the crew pairing cost function. This adjustment allows for the prioritization of crew schedules that are less likely to be disrupted by uncertainties, leading to more reliable and cost-effective operations. Additionally, this study also introduces a new metric to assess the predictive power of survival models in correctly predicting reliability scores. The efficacy of `SurvCG` is demonstrated through extensive simulations of different scenarios with varying disruptions, which show a significant reduction in propagated delay and disruptions.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my supervisors, Dr. Fatma Gzara and Dr. Sirisha Rambhatla. Their limitless patience and invaluable guidance have been instrumental in shaping my understanding of this complex subject. Our lengthy discussions transformed me into a more insightful and diligent researcher. I am profoundly thankful for their constant support, encouragement, and mentorship throughout this journey.

I would also like to thank the team at NAVBLUE for their crucial support and collaboration. The knowledge they shared and their assistance were instrumental to the completion of this thesis.

My heartfelt thanks go to my beloved family for their constant love, understanding, and encouragement. They have been my pillars of strength, standing by me through every challenge. Their unwavering belief in my abilities has been a continual source of motivation.

Lastly, I would like to acknowledge all those who contributed to this journey in ways both big and small. Their support, advice, and encouragement have been instrumental in the completion of this thesis.

Dedication

This is dedicated to the one I love.

Table of Contents

Author's Declaration ii

Statement of Contributions iii

Abstract iv

Acknowledgments vi

Dedication vii

List of Figures xii

List of Tables xv

1 Introduction 1

2 Embedding-Based Representation Learning for Forecasting Flight Characteristics[1] 6

2.1 Introduction 7
2.1.1 Contributions 12
2.2 Literature Review 13
2.2.1 Block Time Prediction 14
2.2.2 Delay Prediction 16
2.3 Proposed Approach 19
2.4 Experiments 22
2.4.1 Data Overview 22
2.4.2 Data pre-processing 28
2.4.3 Baseline Models 29
2.4.4 Neural Networks (NN) Architectures 30
2.4.5 Evaluation Metrics 33
2.5 Results and Discussion 35
2.5.1 Comparing Baselines vs. Neural Networks 35
2.5.2 Performance Evaluation and Ablations 39
2.5.3 Learned embeddings 40
2.5.4 Sensitivity Analysis 43
2.6 Conclusion 45

3 Reliable Crew Pairing Optimization **46**

3.1 Introduction 47

3.2 Preliminaries : Crew Pairing using Column Generation 50

3.3 Related Works 52

3.4 Predicting Flight Connection Reliability with Survival Analysis 54

3.4.1 Methodology for Reliability Prediction 55

3.4.2 Evaluating the Survival Analysis Model 58

3.5 Reliable Crew Pairing Framework 61

3.5.1 Survival-based Column Generation (`SurvCG`) 63

3.5.2 Nominal vs Reliable Cost Functions 65

3.5.3 Overall Framework 66

3.6 Experimental Results 67

3.6.1 Survival Model Testing and Reliability Prediction 67

3.6.2 Instance Generation 69

3.6.3 Comparison of Solutions 71

3.6.4 Simulation 75

3.7 Conclusions and Discussions 79

3.8 Appendix 83

3.8.1 C-index and where it fails 83

3.8.2 Instance Generation in Detail 83

4 Conclusions and Future Work 86

References 88

List of Figures

2.1 Segments of Flight Operation 8

2.2 Block Diagram: Harnessing Categorical Embeddings and Continuous Features for Specific Tasks 20

2.3 Block Time (ABT) Distributions for 2018. The main plot illustrates the aggregate distribution, while individual subplots capture airline-specific variations denoted by IATA codes as described in Table 2.1 24

2.4 Block Time (ABT) Variations Across Top 10 Busiest Links. Violin plots depict block time variations for 10 busiest routes, with internal lines indicating quartiles and means, showcasing both within-route variability and comparisons across routes 24

2.5 Confidence Intervals of Test RMSEs for baselines models and `NN_EMB`. The blue markers on the graph show the average Test RMSE of the models, while the error bars indicate their 95% confidence intervals. 34

2.6 Hyperparameter Impact on Test RMSE: Each point denotes the mean Test RMSE for specific batch size and learning rate combinations, averaged over three distinct random seeds. Notably, `NN_EMB` exhibits consistent performance around an RMSE of approx. 11 mins across various hyperparameter settings. 38

2.7 Embeddings from the best-performing `NN_EMB` run. Red-highlighted clusters and arrows indicate consistent patterns and sequential trends, respectively, across varying learning rates and perplexity values. 41

2.8 Comparative Train and Test Performances of models 43

2.9 Prediction error across airlines 43

2.10 Prediction error across months 44

3.1 Illustration of P-index Calculation: The top panels show q_{pred} within the acceptable range R of q_{true}, indicating a successful prediction. The bottom panels depict q_{pred} outside R, indicating an unsuccessful prediction. The P-index is the ratio of successful predictions (within R) to the total predictions across all query times. 59

3.2 Block diagram for Reliable Crew Pairing Framework 62

3.3 Effects of varying λ_1 and λ_2 on the Reliability Integrated Cost Function . . 66

3.4 Block diagram for Instance Generation: The process begins with a dataset $\mathcal{D}$ filtered by constraints $\mathcal{C}$ to produce filtered flights $\mathcal{F}_0$. An initial network $\mathcal{N}_0$ is constructed with these flights, and further pruned to create the final network $\mathcal{N}$ with adjusted arcs and costs. 69

3.5 Comparison of reliability distribution of arcs selected across solutions . . . 72

3.6 Impact of α on Deadheads 74

List of Tables

2.1 Airlines and their percent share of total flights in 2018 23

2.2 List of Variables 25

2.3 Summary Statistics Across Temporal Categorical Features (Part 1) 25

2.4 Summary Statistics Across Temporal Categorical Features (Part 2) 26

2.5 Summary Statistics Across Temporal Categorical Features (Part 3) 27

2.6 Summary Statistics Across Temporal Categorical Features (Part 4) 28

2.7 Baseline Models 29

2.8 Test Performance Comparisons: Each Test RMSE represents the RMSE for the best set of hyperparameters, averaged across randoms three distinct random seeds. 34

2.9 Tukey HSD Post-Hoc Comparisons: Showcases mean Test RMSE differences, 95% CIs, and Adj. P values, affirming the statistical significance of `NN_EMB`'s performance over baselines. 35

2.10 Best Parameters for the Different Baseline Methods 36

2.11 Overview of architecture modifications. Each Test RMSE represents the RMSE for the best set of hyperparameters, averaged across randoms three distinct random seeds. Read from top to bottom; symbols Δ, $-$, and $+$ denote change via replacement, removal, and cumulative addition, respectively. 38

3.1 Summary of Survival Analysis Methods 58

3.2 Survival model comparison 68

3.3 Comparison of Deadheading and Total Costs for Nominal and Reliable solutions ($\alpha = 2$). Changes in costs (increase or decrease, in parentheses, are percentages) compared to nominal. Headers: DFC - Deadhead Flying Cost, DCC - Deadhead Connection Cost, TFC - Total Flying Cost, TCC - Total Connection Cost, TC - Total Cost. Solutions: N - Nominal, R1 - $\lambda_1 = 10, \lambda_2 = 3$, R2 - $\lambda_1 = 20, \lambda_2 = 4$ 71

3.4 Comparison of Deadheading and Total Costs for solutions with different values of α. Changes in costs (increase or decrease) are in parentheses compared to $\alpha = 1$ and are expressed as percentages. Headers: DFC - Deadhead Flying Cost, DCC - Deadhead Connection Cost, TFC - Total Flying Cost, TCC - Total Connection Cost, TC - Total Cost. 74

3.5 Simulation Scenarios Based on Percentage Irregularity and Level of Delay. R: Regular Operations, IR: Irregular Operations. For IR runs, delay values are sampled from the specified percentile using KDE. 76

3.6 Total Propagated Delays for 75R,25IR Scenarios (70, 80, 90); N: Nominal, R: Reliable. 80

3.7 Total Propagated Delays for 50R,50IR Scenarios (70, 80, 90); N: Nominal, R: Reliable. 80

3.8 Total Propagated Delays for 25R,75IR Scenarios (70, 80, 90); N: Nominal, R: Reliable. 81

3.9 Total Propagated Delays for 100R 0 and 100IR 70 Scenarios; N: Nominal, R: Reliable. 81

Chapter 1

Introduction

The demand for rapid air transportation has surged in recent decades, intensifying air traffic flow and creating more complex operational challenges for airlines. The global aviation industry has experienced unprecedented volatility in recent years. FAA data from FY 2019-2023 reveals passenger numbers plummeting 45.5% during the COVID-19 pandemic, from 1.06 billion to 577 million, before rebounding to 1.04 billion by FY 2023. This rapid recovery, coupled with fluctuating passenger load factors - from a high of 83.36% in FY 2019 to a low of 63.84% in FY 2021, and back to 83% in FY 2023 - has continually created challenges for the airline industry [39].

On the other hand, airlines are confronted with significant operational costs, including fuel, crew, and maintenance expenses. The industry also faces challenges from stricter regulations, unpredictable demand fluctuations, and weather-related disruptions causing delays, climate change and increasing costs dramatically[61]. These factors collectively

lead to them operating under very tight profit margins [34]. Therefore, it is of critical importance to improve the profitability of schedules and reduce operational costs, which can also lead to fewer delays, reduced carbon emissions, and higher customer satisfaction. The key to realizing these improvements lies in the implementation of sophisticated and effective airline scheduling optimization techniques.

To be profitable, airlines primarily manage two key assets: aircraft and cabin crew, which directly relate to schedule planning and the crew scheduling problem [94]. Airline schedule planning encompasses four core problems [13, 92]. These include a)*schedule design*, which involves determining market service, flight frequencies, and scheduling; b)*fleet assignment*, where aircraft are allocated to specific flights; c)*aircraft maintenance routing* to ensure all maintenance requirements are met; and d)*crew scheduling*, which involves assigning crew members to flights.

These core problems are interrelated and make airline scheduling one of most challenging and crucial to make sure that all the flights happen safely and are on schedule. These operations are further complicated by inherent uncertainties in resource availability such as crews and aircraft, and operating conditions like weather. A last-minute unforeseen event like a snow storm may lead to significant disruptions that are costly for airlines and often wipe out weeks' worth of other efficiency measures [26].

As a result, planning continues leading up to execution with revisions made in light of short-term circumstances. Airlines often rely on manual adjustments to manage disruptions in day-to-day plan execution leading to costly inefficiencies. While the industry is moving towards data-driven solutions, airlines recognize that longer-term robust planning remains

a challenge. As a result, the industry is looking to leverage historical data to better understand flight characteristics, diagnose inefficiencies, and integrate with planning can greatly improve the operations and efficient use of personnel and infrastructure.

In this thesis, we focus on solving two important problems within this complex operational framework: block time prediction and reliable crew pairing optimization. Block time, the total time from when an aircraft first moves from its departure gate until it comes to a stop at the arrival gate, is a fundamental metric that influences scheduling, resource allocation, and passenger satisfaction. Error in estimations can lead to a cascade of issues, including delays, inefficient resource utilization, and increased operational costs. Conversely, reliable block time predictions enable airlines to create more realistic schedules, improve on-time performance, and enhance customer satisfaction. This study introduces an embedding-based neural network model for not only block time prediction but for any other flight segment as well, representing a novel application in aviation analytics for long-term planning. The model leverages embedding-based representation learning to capture complex non-linear relationships within high-cardinality features, improving the predictive accuracy of flight scheduling. This approach addresses the challenges of traditional encoding schemes that fail to capture intricate data relationships. Comprehensive ablation studies and comparisons with ten baseline machine learning models reveals the robustness and effectiveness of the proposed neural network architecture. The study conducts extensive evaluations using Monte Carlo simulations and t-distributed Stochastic Neighbor Embedding (t-SNE) to visualize high-dimensional representations, revealing the patterns learned by the model.

While flight segment prediction provides extensive knowledge about delays, it does not

directly inform planning, which requires knowing if a crew can still make a connection despite delays. Therefore, the second work, Reliable Crew Pairing Using Survival Analysis, integrates survival analysis into the crew pairing problem to address uncertainties in flight schedules. This study introduces the SurvCG framework, which applies survival analysis techniques to predict the reliability of flight connections based on historical data. By incorporating survival probabilities into the crew pairing optimization process, the approach enhances the robustness of crew schedules against disruptions. With these survival probabilities, we propose a cost function which, to the best of our knowledge, is the first to incorporate data-driven flight connection reliability to build schedules. This cost function is designed to ensure that low reliability incurs higher costs, and vice-versa, to inform the optimization. This dynamic adjustment leads to more realistic and flexible scheduling. Extensive simulations demonstrate the efficacy of the SurvCG framework in reducing operational disruptions and improving schedule reliability with lower total propagated and pairing delays. The study also introduces new metrics to evaluate the predictive power of the survival analysis model. and validates its effectiveness through comprehensive data-driven simulations.

This thesis contributes to the field of airline operations optimization through two primary works:

Embedding-Based Representation Learning for Forecasting Flight Characteristics

- Introduces an embedding-based neural network model for block time prediction.
- Captures complex non-linear relationships within high-cardinality features.

- Validated through comprehensive ablation studies and comparisons with ten baseline models.

Reliable Crew Pairing:

- To the best of our knowledge the first use of time-to-event models (survival analysis) in the context of flight scheduling to quantify the real-world uncertainties.
- Incorporate the data-driven reliability scores augmenting the traditional cost functions for the crew pairing optimization.
- Rigorous empirical validation through simulations that demonstrate the robustness of the proposed solution.

The thesis is structured as follows: Chapter 1 provides and introduction the thesis, with a discussion on the complexity and importance of scheduling optimization in the airline industry. This is followed by Chapter 2 and Chapter 3, each detailing the respective studies undertaken as part of this thesis. Chapter 2 discusses the development and implementation of an embedding-based representation learning model for forecasting flight block times. It details the methodology, experiments conducted, and the analysis of results, demonstrating the model's efficacy compared to baseline methods, and provides all necessary information to ensure the work is reproducible.

Chapter 2

Embedding-Based Representation Learning for Forecasting Flight Characteristics[1]

Airlines face significant challenges when building flight schedules, particularly due to unpredictable operations caused by factors such as adverse weather, airport congestion, mechanical problems, etc. One of the major components of flight scheduling is block time; accurately estimating it is crucial for optimizing resource utilization and effective planning (scale of minutes). Given that flight scheduling takes place months in advance, accurately predicting block time is a challenging task. This is largely due to the limited availability of features impacting operations on a specific day, such as weather,

[1]The following chapter has been sent for publication to *Transportation Research Record*: Biswal, A., Rambhatla, S., and Gzara, F. (2024). *Embedding-Based Representation Learning for Forecasting Flight Characteristics. Transportation Research Record*, 2024

at the time of planning. Consequently, current literature suggests that popular machine learning models are not suitable, recommending the use of statistical historical metrics. However, these methods do not a) capture the complex latent relationships between factors affecting block time, b) effectively handle high-cardinality categorical data and temporal variations, and c) only consider very small number of flights for their conclusions. We conduct, to the best of our knowledge, the first large scale study on airline on-time performance database from 2018 Bureau of Transportation Statistics (BTS), a public dataset. Specifically, our work introduces an embedding-based representation learning model to efficiently incorporate high cardinality categorical features and improve the long-term predictive capabilities of the model. These embeddings also encapsulate richer feature representations and their interactions. Complementary to these, we conduct rigorous experimental evaluations across 10 baselines and significance tests to demonstrate the advantages of using our embedding-based model to increase long-term forecast accuracy for planning. For reproducibility the code is made available at https://github.com/criticalml-uw/Embeddings-for-Block-Time-Prediction

2.1 Introduction

Flight schedule planning involves complex decision-making at longer time horizons, typically four to six months before the actual day of operations. This stage in operations planning is critical since it directly impacts resource requirements, revenues, and costs. Central to building an efficient schedule is the allocation of correct time estimates for each operational segment of a flight. By accurately estimating these durations, airlines aim to

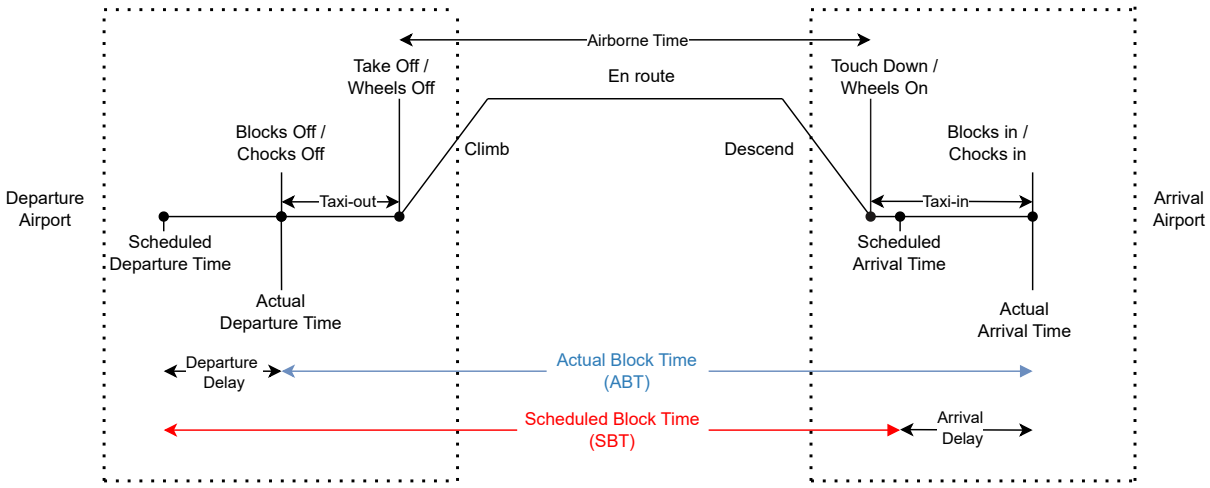


Figure 2.1: Segments of Flight Operation

align the planned schedules closely with the actual time taken on the day of operations. Here, even a 1 to 2-minute reduction in prediction error is critical, as it can determine whether a flight can connect seamlessly with the next. Achieving this level of accuracy is essential for On-Time Performance (OTP), a key metric that airlines use to demonstrate the timeliness and reliability of their operations.

An essential aspect of schedule building, which takes place months in advance, is determining the scheduled block time (SBT) of a flight, which is the time interval between the scheduled departure and scheduled arrival of a flight; See Fig. 2.1 based on [31] which shows various segments of flight operations that can impact the block time. This contrasts with the actual block time (ABT), which is the measure of time duration between blocks-off and blocks-in, as observed on the day of the flight. SBT serves as an estimate of ABT, informing the setting of departure and arrival times. The discrepancy between SBT and ABT can lead to departure and arrival delays and is crucial for robust planning and OTP.

Several challenges are encountered in SBT estimation due to the long-term prediction horizon and the dynamic nature of airline operations. Prediction takes place months in advance, making it impossible to accurately anticipate factors such as wind, weather, air traffic, and aircraft load [1, 38, 88]. As a result, only factors that are known a few months in advance such as origin and destination airports, along with time-based features (day, week, month, etc.), can be utilized in any prediction modeling. Moreover, airlines make adjustments closer to flight departure, to accommodate real-time changes which often go unrecorded. When planning, airlines typically lack information on what affects operations on the actual flight day. This limited availability of explanatory variables in the data makes achieving accurate predictability extremely challenging. Another challenge comes from the nature of the explanatory variables. Many features such as departing and arrival airports, and time-based features are categorical, are of high cardinality, and have spatio-temporal effects that are not well captured by traditional encoding schemes in machine learning. This complexity demands advanced methods for effective modeling and prediction.

Both over- and under-estimation of SBT is undesirable. With a shorter block time, the possibility of delayed arrival arises, leading to disruption of flight sequences, crew displacement, and additional expenses like food and lodging. In contrast, introducing buffers to block time may cause early arrivals, potentially resulting in inefficient utilization of resources such as crew allocation, leading to sub-optimal outcomes [10, 91, 55, 2]. To optimize scheduling and maximize resource utilization, airlines strive to find the most accurate SBT estimate that closely aligns with the ABT.

A broad spectrum of approaches have been employed to estimate flight block times, their components (such as taxi and flight time), and delay. These methodologies range

from machine learning techniques including Random Forest, and XGBoost, as well as deep learning models, which have been applied to predict short-term delays [76, 96, 27, 60, 12]. However, much of this work tends to focus on short-term prediction, typically hours or a day before the day of operations, and often excludes certain links or routes, suggesting a gap in broader applicability across the airline industry. There is a notable absence in the literature concerning long-term SBT prediction modeling. A recent work by Abdelghany et al. [2] concluded that machine learning models, specifically regression tree, random forest, and XGBoost have limited capability in predicting ABT. They show that these models were outperformed by historical median-based estimates of the ABT. Moreover, this study only considered a very limited set of routes (seven), which necessitates a large-scale analysis and investigation and development of more sophisticated neural network-based approaches for this critical task to effectively exploit the information available at the planning stage and enhance the predictive capabilities of the models.

Adding to the complexity of modeling is the inherent problem with traditional encoding schemes. Although straightforward and commonly used, encoding schemes like one-hot encoding have significant drawbacks. Primarily, since it adds a variable for each unique category for a feature, the model's dimensionality increases proportionally to the cardinality of the categorical variables. This escalation poses a challenge known as the *curse of dimensionality,* which demands exponentially more data for the model to remain accurate [14]. Another issue with one-hot encoding is it assumes that categories are mutually exclusive and unrelated, leading to representations that are orthogonal and equidistant. This fails to capture any natural similarity between categories, overlooking their potential associative or semantic connections [24]. For instance, consider the scenario where we're analyzing

a dataset related to Days of the month in airline operations. For a human observer it is clear that the last day of a month and and the first day of the following month are close, but once this feature is one-hot encoded we lose this natural progression. Moreover, while circular transformations to encode this closeness between days can be useful, these cannot capture additional complex relationships and similarities. Therefore learning these latent relationships is key to improve long-term forecasting, especially for a very limited feature set.

To bridge these gaps, our study adopts an embedding-based representation learning to discern intricate non-linear relationships through a neural network framework. Specifically, embeddings are task-dependent relationships that the model learns to help the downstream task. This methodology efficiently addresses the challenge of encoding categorical features with high cardinality, a task often challenging with traditional encoding schemes. Embedding techniques have significantly evolved to capture complex data relationships in various domains. Starting with RESCAL [71] and TransE [17] for knowledge graphs, these methodologies have laid the groundwork for advanced representations. DeViSE [40] further extended embeddings to map images into semantic spaces. Subsequently, methods like DeepWalk [72] and Node2Vec [44] have been developed for graph representations. The most recent work on the use of embeddings for transportation research by Arkoudi et al. [8] introduces embedding encoding for socio-characteristic variables, aligning latent representations with individuals' travel behavior choices. This progression highlights the versatility and importance of embedding models in capturing nuanced data relationships across a spectrum of applications.

This study aims to leverage embedding-based representation learning in a neural net-

work to improve flight scheduling predictions. By learning embeddings, the proposed neural network architecture is intrinsically aware of the task it is being trained for and concurrently learns the multi-dimensional relationships as the model trains. To ensure the robustness of our approach, we conducted ablation studies, leading us to the best-suited neural network architecture. Additionally, we benchmarked our approach against ten baseline ML models for a thorough comparison using Monte Carlo simulations. Furthermore, we employ t-distributed Stochastic Neighbour Embedding (t-SNE) on the learned embeddings, to visualize the high-dimensional representations revealing patterns learned by the proposed model.

2.1.1 Contributions

1. **Embedding-Based Model:** We introduce an embedding-based neural network model for block time prediction, a novel application in aviation analytics for long term planning.

2. **First large-scale study for broader generalization:** Unlike the previous studies that limit their scope to busier links or several airports, our study uses the entire network ($\sim$6 million flights across $\sim$5500 routes), ensuring our model's generalizability and applicability.

3. **Rigorous experimental validation:** We rigorously test our model against ten baseline models using three distinct seeds, ensuring model robustness, reliability, and reproducibility. We also include statistical significance results to further demonstrate the reliability of our model.

4. **Ablation Studies:** To evaluate the utility of each component of the proposed solution we conduct extensive ablations across feature compression, embedding integration, dropout, batch normalization, and non-linearity, on the predictive performance.

5. **Visualization of embeddings:** We employ t-SNE to explore the learned embeddings, and provide insights into the complex relationships learned by the model.

In the sections that follow, we present a literature review on block time and delay prediction, followed by a detailed description of the methodologies applied in this study. Subsequently, we assess the performance of our predictive models and conduct sensitivity analysis.

2.2 Literature Review

Through an extensive literature review, we uncovered a diverse range of models that address the estimation of block time, its components (i.e. taxi time, flight time), and delays. While the broader scope of this paper encompasses the long-term prediction of ABT, delays, taxi times, and other related downstream tasks to aid airline operational planning in the long term, our experimental focus and demonstrated results center on the prediction of ABT. In this section, we review the literature on prediction modeling of time segments of a flight operation, focusing on block time and delay which seem to have attracted most of the work. We begin by examining the efforts made to estimate block time. Then,we review the modeling of delay propagation and prediction while making the distinction between short-term and long-term prediction.

2.2.1 Block Time Prediction

The prediction of ABT has been the focus of two studies by Wang et al. [85] and Abdelghany et al. [2]. Wang et al. [85] proposed a stacking model that demonstrates promising generalization for the prediction of block time for various spatial-temporal instances. Abdelghany et al. [2] used three machine learning(ML) models, namely Regression Tree (RT), Random Forest Regression (RFR), and eXtreme Gradient Boosting (XGBoost) regression to know of when planning their schedules several months in advance. They found that the performance of the ML models was poorer when evaluated against a benchmarking scenario, where the median value of historical actual block times was used as an estimate. The analysis employed Bureau of Transportation Statistics (BTS) data from 2019, focusing on seven airport pairs. For the flights along these routes, the study employed outlier filtering by applying a lenient approach using Lower Bound $= Q1 - 2 \times IQR$ and Upper Bound $= Q3 + 2 \times IQR$. Here, $Q1$ represents the first quartile, marking the 25% percentile of the dataset, and $Q3$ represents the third quartile, marking the 75% percentile. The IQR (Interquartile Range) is defined as the difference between $Q3$ and $Q1$, reflecting the middle 50% of the data. This approach has the drawback of eliminating a substantial amount of valuable information, including the crucial tails of the distribution. Their findings indicate that XGBoost outperformed both RT and RFR models in terms of prediction accuracy. The study identified seasonality, aircraft type, departure/arrival hours, and traffic at the airport as the most influential variables. Furthermore, the study revealed significant variability in actual block time and its components across different airport pairs, with longer routes exhibiting greater variability. It is worth noting that the study's scope was con-

fined to only seven specific one-way routes, potentially limiting the generalizability of the findings to the broader airline network.

To enhance the understanding of Scheduled Block Time (SBT) estimation, previous studies have undertaken various analytical approaches. Coy [30] developed 2-stage statistical models corresponding to six US airlines to predict SBT. The approach involved estimating an initial block time by averaging the previous year's similar flights and using it as a prediction variable in a second-stage regression model. The models are claimed to capture over 95% of the variance, indicating a strong fit. However, their approach relied on a subset of data obtained by dropping the less frequented routes, avoiding the challenges of the long-tailed distribution of block time. In contrast, our approach harnesses the entirety of the data. Moreover, their reliance on variables such as traffic and weather conditions, which are not readily available during the planning phase, potentially limits the practical application of their model for long-term predictions.

Sohoni et al. [78] note that airlines typically determine SBT based on fixed percentiles of historical data. Expanding on this, Hao and Hansen [47] suggest that airlines commonly adopt an SBT ranging from the 65th to 75th percentile of ABT. However, Sohoni et al. [78] argued that these approaches have not yielded substantial improvements in OTP. Building upon this critique, our study introduces a machine-learning framework that is not solely dependent on historical trends but is also capable of learning from available explanatory variables. Deshpande and Arıkan [31] analyzed flight buffer selection using the newsvendor problem as an analogy, where early or late flight arrivals equate to newspaper vendors experiencing surplus or shortage. Their study revealed that buffer decisions are notably influenced by carrier types, route market shares, and specific route attributes.

Kang and Hansen [56] modeled how airlines adjust SBT to balance OTP. They analyzed five U.S. domestic airlines and found that schedulers are inclined to extend SBT by 0.38 to 0.54 minutes per 1% rise in OTP. In the same context, Wang et al. [88] compare the differences in this SBT setting behavior between major airlines in the US and China. They use econometric models based on historical ABT distributions, departure delay, as well as other factors that may drive SBTs. They perform counterfactual analysis which indicates that Chinese airlines could attain U.S.-level OTP through U.S. SBT setting practices, while U.S. airlines might experience slightly lower OTP using Chinese SBT setting methods.

Instead of relying on block time prediction, Gui et al. [45] propose a data-driven 3-stage method to enhance the estimation of expected arrival time. Their method involves identifying aircraft arrival patterns (clustering), classification (XGBoost), and estimating flight time (XGBoost). This approach primarily focuses on short-term prediction since it relies on real-time radar trajectory data, including current, historical, and traffic situation information. As a result, it is more suitable for immediate operational use rather than long-term planning which is the focus of this paper.

2.2.2 Delay Prediction

Flight delay prediction has been extensively researched in the literature ([79], [23]). A significant differentiation can be observed in the practical application of these predictions. In the long term, such predictions are employed for planning and scheduling purposes, including activities like booking slots at the airport, which are done several months in advance of the flight departure. On the other hand, in the short term, predictions are

utilized to optimize and enhance the operational efficiency of airlines in real-time.

One of the first studies that incorporated the spatio-temporal aspect into the prediction of delay was by Rebollo and Balakrishnan [76]. They define a NAS (National Airspace System) delay state at time t as a vector of the departure delays of all of the Origin-Destination (OD) pairs at time t. This delay for an OD pair at time t refers to the median delay of all the flights that fall within a 2-hour time window beginning at time t. These NAS delay observations are classified both as a characteristic delay pattern of NAS (i.e. spatial) as well as a characteristic type of day(temporal). These are then used as explanatory variables which represent the system to predict departure delays using Random Forests. They evaluated their model on the 100 most delayed links and got an average test error of 20.9 minutes. Subsequent studies by Gopalakrishnan and Balakrishnan [42] compared different approaches including (MJLS), and Artificial Neural Networks (ANN), which achieved a 94% accuracy in classifying the 100 delay links with a threshold of 60 minutes. Our study builds upon these insights, targeting a wider scope beyond the 100 most delayed links. Wang and Vaze [86] model the distribution of primary delays in the NAS, identifying a bimodal pattern with a peak at zero and declining frequency for positive delays, and use a two-phase approach due to quantile regression's limitations with this distribution. While insightful on delay probabilities, their model doesn't predict exact delay times for individual flights.

Deep Learning models, on the other hand, have excelled in capturing complex non-linear relationships in space and time [96, 60]. Researchers have transformed the multi-airport flight delay prediction into a graph representation learning task [12]. Several other efforts have also been made to develop tree-based and neural network-based models to predict

delays at the individual flight, airport, link, and network levels with a prediction horizon of minutes, hours, and days [96, 60, 27]. Another formulation that is frequently seen is the classification problem, which focuses on classifying delay on an OD link, departure/arrival at an airport, or individual flights based on predetermined delay thresholds. Alonso and Loureiro [5] define the intervals as $(-\infty, 0)$, $[0, 15)$, $(15, 30]$, $(30, 60]$, and $(60, \infty)$.The idea here is to assign delays to one of these severity levels, allowing for a more detailed understanding of delay patterns and their influence on operations.

Interestingly, most of the research around delay prediction has been restricted to the last hours or days before departure, when data sources are abundant. For example, weather data and information on the state of the airline network, both strongly associated with delay, are typically used. However, in a long-term context, only a couple of studies have addressed this challenge. Lambelho et al. [63] proposed a generic approach to assess strategic flight schedules (arrival/departure slots several months before execution) based on flight delay and cancellation predictions. Their analysis showed LightGBM outperformed other models in delay prediction, emphasizing key factors like airlines and seats in forecasting arrival delay with high accuracy and AUC metrics. Kafle et al. [55] presented an analytical model to quantify the propagated and newly formed delays. Wong et al. [91] used survival analysis to find key contributing factors for departure and arrival delays. Their results indicate that key factors affecting departure delays include turnaround buffer time, aircraft type, logistics, and weather, while arrival delays are primarily influenced by block buffer time and weather conditions.

2.3 Proposed Approach

Our objective is to develop a predictive model equipped to process high-dimensional categorical variables and intricate relationships between them for predicting delays, block time, taxi-in, taxi-out times, and other critical operational parameters. The predictive model for such tasks can be formulated as:

$$y = f(\boldsymbol{x}^{\text{cat}}, \boldsymbol{x}^{\text{cont}}), \tag{2.1}$$

where $y \in \mathbb{R}$ represents the outcome of interest (e.g., block time but this can be delay time, taxi-out time etc. for other applications), $f(\cdot)$ is the predictive function, $\boldsymbol{x}^{cont} \in \mathbb{R}^{n_{cont}}$ denotes the vector for continuous features pertinent to the downstream prediction tasks, where n_{cont} is the total number of continuous variables, and $\boldsymbol{x}^{\text{cat}} \in \mathbb{R}^{n_{cat}}$ denotes the vector for the categorical features.

The features $\boldsymbol{x}^{\text{cat}}$ and $\boldsymbol{x}^{\text{cont}}$ constitute the features influencing the outcome y. Some of these are deterministic and well-understood, yet many exhibit intricate relationships that are non-deterministic and less apparent. Specifically :

- **Temporal Variations:** Seasonal factors, like snow, may lead to longer block times due to requirements like de-icing. Weekends or peak travel times might exacerbate taxi times because of congestion.
- **Airline-specific practices:** Different carriers maintain distinct operational philosophies. An airline might value quick turnarounds, optimizing for reduced block times, while another could emphasize longer layovers to ensure a buffer against disruptions.

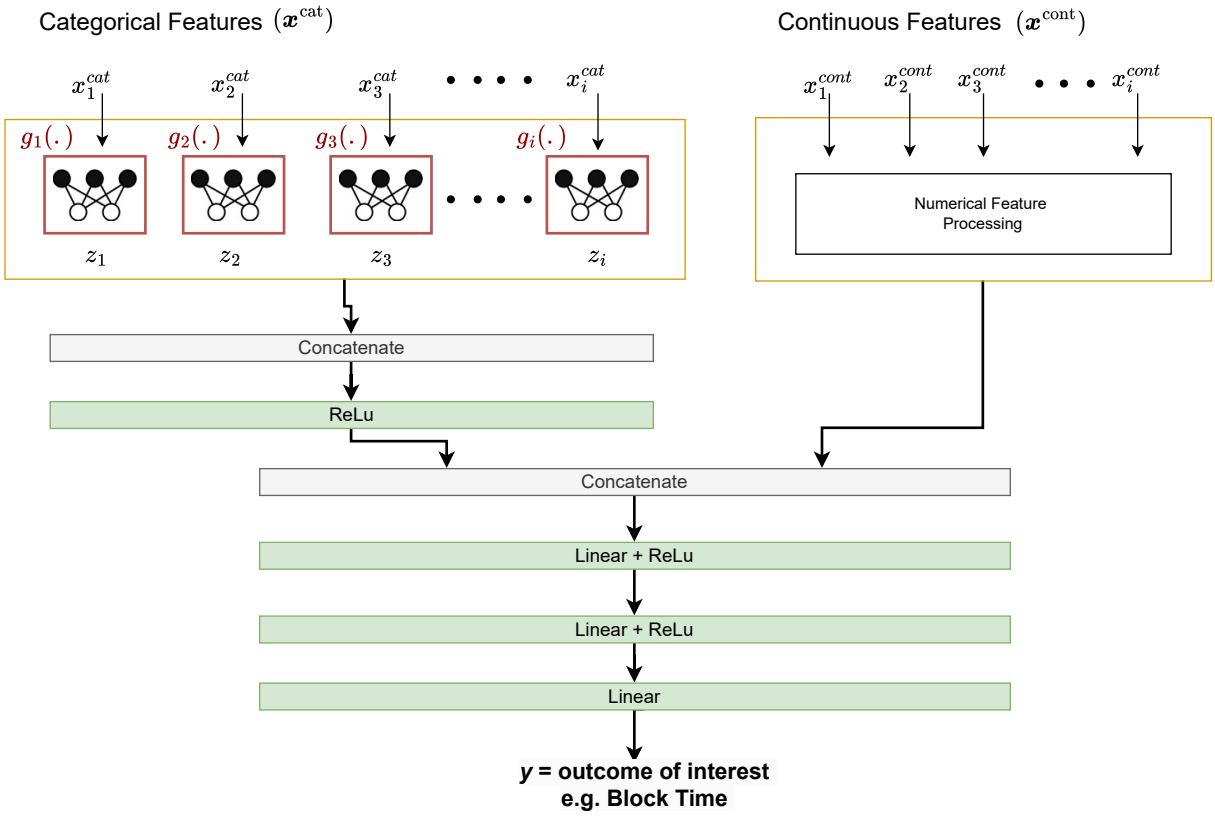


Figure 2.2: Block Diagram: Harnessing Categorical Embeddings and Continuous Features for Specific Tasks

The non-deterministic nature of these factors presents challenges when incorporating domain expertise into prediction models. To effectively capture these multifaceted interactions, we undertake a representation learning task.

To leverage the latent information encoded within the data we enrich our feature set by learning the embedding $\boldsymbol{z}_i$ from the categorical variables x_i^{cat}. To this end, we first represent x_i^{cat} as its one hot encoded counterpart $\boldsymbol{v}_i^{cat}$ (of dimensions commensurate with the categorical values x_i^{cat} can take). For instance, if the feature x_i^{cat} can take m_i unique cate-

gorical values, then $\boldsymbol{v}_i^{cat} \in \mathbb{R}^{m_i}$ is a vector of zeroes except a 1 at the location corresponding to the value x_i^{cat}.

Now, introducing $\boldsymbol{v}_i^{cat}$ we obtain the representation $\boldsymbol{z}_i = g_i(\boldsymbol{v}_i^{cat})$, where $g_i(\cdot)$ is the embedding function and $\boldsymbol{z}_i \in \mathbb{R}^{d_i}$ is the embedding of the i^{th} categorical feature. The role of the embedding function $g_i(\cdot)$ is critical. It maps each level of the categorical variable x_i^{cat} to a low-dimensional parameter vector. Embedding functions are typically linear mappings which can be represented by an embedding matrix $\boldsymbol{G}_i$ as

$$\boldsymbol{z}_i = \boldsymbol{v}_i^{\top} \cdot \boldsymbol{G}_i \tag{2.2}$$

Given this, we train a mapping $f(\cdot)$ (a neural network in this case) as:

$$y = f\left(\left[\boldsymbol{x}^{\text{cont}}, \boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{n_{cat}}\right]^{\top}\right). \tag{2.3}$$

The action of the first layer of such a neural network can be written as follows (see also Fig. 2.2).

$$\boldsymbol{h}_0 = \sigma(\boldsymbol{W}_1 \cdot \boldsymbol{x}_0 + b_1), \text{ for } \boldsymbol{x}_0 = \left[\boldsymbol{z}_1; \boldsymbol{z}_2; \ldots; \boldsymbol{z}_{n_{cat}}; \boldsymbol{x}^{\text{cont}}\right] \tag{2.4}$$

Here, the function σ denotes the activation function, which can be any non-linear function like the sigmoid, tanh, or ReLU, introducing non-linearity to the model. Note that the embedding matrix $\boldsymbol{G}_i$ is jointly optimized with the parameters of the neural network $f(\cdot)$ via backpropagation, as depicted in Fig. 2.2. This helps us to extract customized representations of input categorical variables to achieve optimal performance for the given task.

2.4 Experiments

This section outlines the dataset, preprocessing steps, the baseline models used for comparison, and the various neural network architectures explored in our study.

2.4.1 Data Overview

The data for this study was obtained from the Airline On-Time Performance Database. This database comprises scheduled actual departure, and arrival times recorded by certified US air carriers, which account for a sizable share of domestic scheduled passenger revenue. This database is compiled by the Bureau of Transportation Statistics' Office of Airline Information (BTS). It provides complete details on on-time arrival and departure dates, flight cancellations and diversions, taxi-out and taxi-in times, causes of delay and cancellation, air time, and non-stop distance.

For the purpose of block time prediction, we use the 2018 data, comprising more than 6 million flights operated by 17 airlines. A breakdown of each airline's contribution, presented as a percentage of the total flights alongside their IATA codes and names, is given in Table 2.1. Figure 2.3 provides a comprehensive visualization of the actual block times (ABT) distributions. The primary plot at the top of the grid depicts the overall distribution across all airlines, revealing a right-skewed pattern with a pronounced long tail. The subsequent plots, each labeled with an IATA code, represent the block time distributions of 17 individual airlines. The differences in these distributions are quite pronounced, reflecting the varied operational profiles of the airlines. Some airlines, potentially those servicing

shorter routes, showcase a more centered distribution, while others exhibit broader spreads, suggesting a mix of short and long-haul flights. Additionally, Figure 2.4 illustrates the block time variation across the top 10 busiest Origin-Destination (OD) pairs, highlighting the variability not only across routes but also within each route due to operating conditions, different airline practices, and other factors, further emphasizing the complex nature of predicting block times accurately.

Table 2.1: Airlines and their percent share of total flights in 2018

IATA Code	**Airline Name**	**% of Total Flights**
WN	Southwest Airlines	21.91
DL	Delta Airlines	15.22
AA	American Airlines	14.65
UA	United Airlines	10.31
OO	SkyWest Airlines	8.52
B6	JetBlue Airways	4.76
AS	Alaska Airlines	4.04
YX	Midwest Airlines	3.29
NK	Spirit Air Lines	2.85
EV	ExpressJet Airlines	2.62
YV	Mesa Airlines	2.50
MQ	American Eagle Airlines	2.05
9E	Pinnacle Airlines	1.97
OH	PSA Airlines	1.75
F9	Frontier Airlines	1.91
HA	Hawaiian Airlines	1.38
VX	Virginia Airlines	0.28

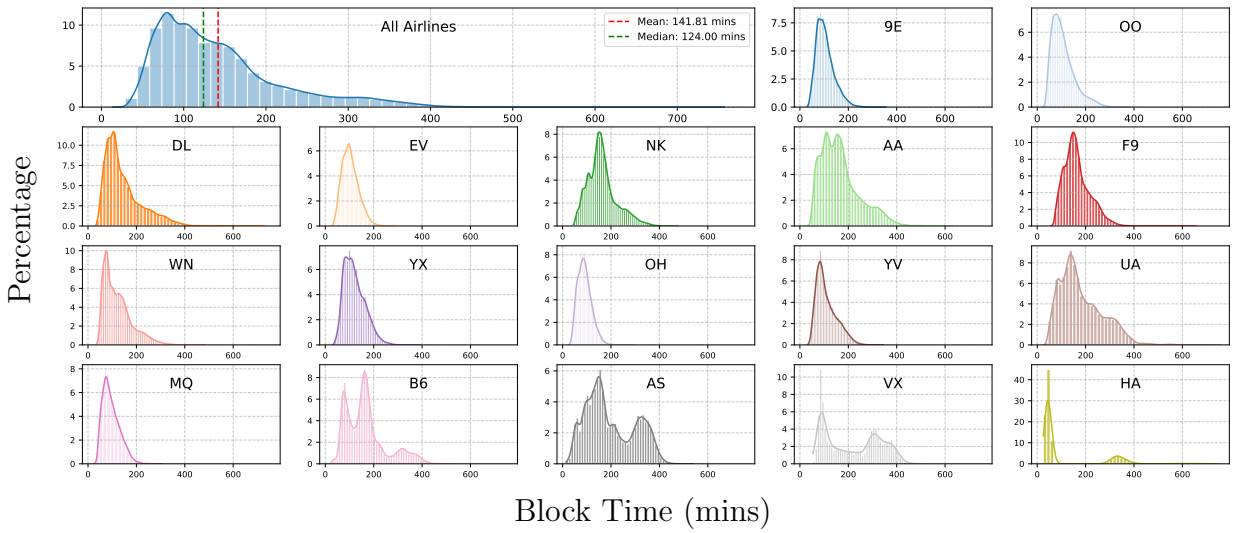


Figure 2.3: Block Time (ABT) Distributions for 2018. The main plot illustrates the aggregate distribution, while individual subplots capture airline-specific variations denoted by IATA codes as described in Table 2.1

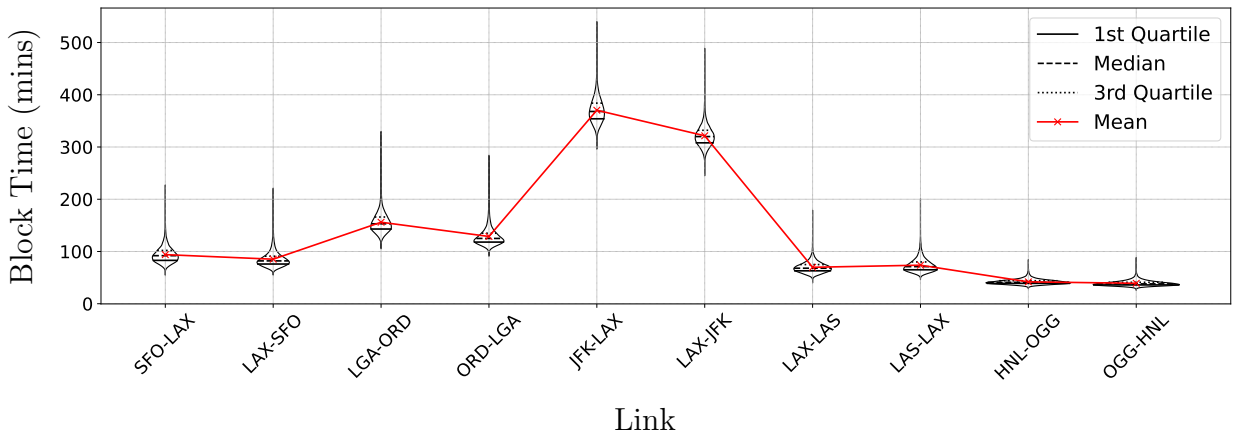


Figure 2.4: Block Time (ABT) Variations Across Top 10 Busiest Links. Violin plots depict block time variations for 10 busiest routes, with internal lines indicating quartiles and means, showcasing both within-route variability and comparisons across routes

Table 2.2: List of Variables

Sr. No.	Variable	Type	Nature
1	Quarter	Categorical	Temporal
2	Month	Categorical	Temporal
3	DayofMonth	Categorical	Temporal
4	DayOfWeek	Categorical	Temporal
5	Reporting_Airline	Categorical	-
6	OriginAirportSeqID	Categorical	Spatial
7	Origin	Categorical	Spatial
8	DestAirportSeqID	Categorical	Spatial
9	Dest	Categorical	Spatial
10	Weekend	Categorical	Temporal
11	IATA Season	Categorical	Temporal
12	FlightSequenceTails	Numerical	-
13	sin_CRSDepTime	Numerical	Temporal
14	cos_CRSDepTime	Numerical	Temporal
15	sin_CRSArrTime	Numerical	Temporal
16	cos_CRSArrTime	Numerical	Temporal
17	OriginLongitude	Numerical	Spatial
18	OriginLatitude	Numerical	Spatial
19	DestLongitude	Numerical	Spatial
20	DestLatitude	Numerical	Spatial
21	Distance	Numerical	Spatial
22	CRSElapsedTime	Numerical	Temporal

Table 2.3: Summary Statistics Across Temporal Categorical Features (Part 1)

Feature	Category	Number of flights (%)	Mean	SD	Min	Max	Med.
Quarter	1	1274079 (21.46)	145.98	77.14	16	739	129
	2	1394307 (23.49)	145.72	77.43	14	757	128
	3	1567716 (26.41)	140.98	75.81	15	736	122
	4	1700866 (28.65)	136.23	72.75	14	723	117

Table 2.4: Summary Statistics Across Temporal Categorical Features (Part 2)

Feature	**Category**	**Number of flights (%)**	**Mean**	**SD**	**Min**	**Max**	**Med.**
Month	January	459645 (7.74)	142.31	75.52	16	728	125
	February	374254 (6.30)	148.33	78.34	16	739	132
	March	440180 (7.41)	147.83	77.63	16	711	132
	April	490286 (8.26)	142.05	75.84	16	757	124
	May	451195 (7.60)	146.97	78.05	16	717	129
	June	452826 (7.63)	148.45	78.35	14	680	131
	July	469832 (7.91)	149.08	78.71	16	736	132
	August	542624 (9.14)	140.93	76.02	16	681	122
	September	555260 (9.35)	134.18	72.35	15	689	115
	October	586714 (9.88)	134.04	72.02	14	684	115
	November	554906 (9.35)	136.26	72.51	17	704	118
	December	559246 (9.42)	138.51	73.67	17	723	120

Table 2.5: Summary Statistics Across Temporal Categorical Features (Part 3)

Feature	**Category**	**Number of flights (%)**	**Mean**	**SD**	**Min**	**Max**	**Med.**
	1	190670 (3.21)	142.32	76.28	16	711	124
	2	191264 (3.22)	142.06	76.28	17	757	124
	3	187919 (3.17)	142.20	76.26	16	717	124
	4	190092 (3.20)	140.68	75.48	17	694	122
	5	197608 (3.33)	140.74	75.48	16	728	122
	6	193019 (3.25)	141.90	75.98	17	705	124
	7	193890 (3.27)	142.00	75.99	15	723	124
	8	193516 (3.26)	141.63	75.67	16	718	123
	9	197497 (3.33)	142.04	75.27	16	699	124
	10	193379 (3.26)	141.74	75.36	18	692	124
	11	197560 (3.33)	141.66	75.55	17	690	123
	12	200615 (3.38)	141.33	75.10	16	705	123
	13	194056 (3.27)	141.62	75.42	17	697	123
	14	193841 (3.26)	141.97	75.58	16	684	124
	15	192032 (3.23)	143.43	76.36	14	736	125
DayofMonth	16	198422 (3.34)	141.56	75.57	16	739	123
	17	193884 (3.27)	141.73	75.62	17	701	124
	18	198076 (3.34)	140.62	75.16	16	728	122
	19	201924 (3.40)	141.29	75.51	16	704	123
	20	196443 (3.31)	142.97	76.11	16	691	125
	21	195247 (3.29)	142.29	75.67	14	696	124
	22	189929 (3.20)	142.28	76.22	17	711	124
	23	196369 (3.31)	141.16	75.51	17	698	123
	24	195446 (3.29)	141.55	75.61	16	693	123
	25	196821 (3.32)	142.00	75.85	17	700	124
	26	201084 (3.39)	142.19	75.84	16	711	124
	27	193075 (3.25)	142.09	75.88	17	708	124
	28	194765 (3.28)	142.47	75.80	18	712	124
	29	183566 (3.09)	141.29	75.59	17	714	123
	30	186109 (3.13)	140.68	75.31	17	704	122
	31	108850 (1.83)	143.23	76.77	18	693	125

Table 2.6: Summary Statistics Across Temporal Categorical Features (Part 4)

Feature	Category	Number of flights (%)	Mean	SD	Min	Max	Med.
DayOfWeek	Monday	892504 (15.03)	141.52	75.49	14	757	123
	Tuesday	850995 (14.33)	140.53	75.17	16	699	122
	Wednesday	861131 (14.50)	140.70	75.26	16	711	122
	Thursday	872859 (14.70)	141.91	75.91	14	725	123
	Friday	880776 (14.84)	141.77	75.81	15	739	123
	Saturday	733982 (12.36)	144.25	76.58	16	714	127
	Sunday	844721 (14.23)	142.35	75.97	16	736	124
Weekend	No	4358265 (73.41)	141.29	75.53	14	757	123
	Yes	1578703 (26.59)	143.23	76.26	16	736	126
IATASeason	Winter	2357187 (39.7)	141.42	75.21	16	739	123
	Summer	3579781 (60.3)	142.06	76.07	14	757	124

2.4.2 Data pre-processing

Accurately capturing the periodic naturre of temporal attributes is critical for modeling the nuances of airline operations. These patterns, especially in the context of timestamps like `CRSArrTime` and `CRSDepTime`, influence metrics like block times in airline operations. Tables 2.3, 2.4, 2.5 and 2.6 provide a comprehensive summary statstics of the temporal categorical features, highlighting the variability of ABT of flights across different time frames. To encode the cyclical pattern in time, we circularly transformed timestamps through the application of sine and cosine functions, enabling them to be represented as angles within a circular space. This approach ensures continuity between the endpoints of a cycle, allowing the subsequent models to recognize and leverage these cyclical dependencies. Recognizing the inter-dependencies of flights in a sequence and to capture the compounding delays, we also introduce `FlightSequenceTails`, which is a numerical attribute that encodes the order of flights based on their flight date and tail number, providing insights into the

cascading delays.

2.4.3 Baseline Models

In predictive modeling, especially when dealing with tabular datasets comprising categorical and continuous features, the choice of algorithm plays a pivotal role in predictive performance. Our choice of baseline models reflects a spectrum of algorithms that have traditionally demonstrated robust performance on tabular data, and they were selected to encompass both linear and non-linear complexities inherent in the data. Additionally, these models range from ones that directly handle regularization to counteract potential overfitting to the ensemble and boosting methods known for their high accuracy in diverse scenarios. The models presented in Table 2.7 were considered.

Table 2.7: Baseline Models

Model	**Description**
Linear	Linear without any regularization.
Lasso	Linear with L1 regularization.
Ridge	Linear with L2 regularization.
Elastic Net	Linear with both L1 and L2 regularization.
Decision Tree	Non-linear using a single tree.
Random Forest	Ensemble of decision trees.
HGBoost	Gradient boosting with histogram-based trees.
XGBoost	Tree-based gradient boosting with level-wise growth.
LightGBM	Tree-based gradient boosting with leaf-wise growth.
AdaBoost	Boosting with multiple weak learners.

For handling the categorical features, one-hot encoding is employed. In the context of Generalized Linear Models (GLMs), this encoding allows each category to be treated

distinctly, with models fitting specific coefficients for each category. For tree-based models, the encoding directly impacts the decision-making process. Tree splits will be determined based on the presence or absence of the encoded categorical variables, thus influencing how the model interprets and acts upon the data. Having experimented with the traditional modeling approaches, we now set our sights on the vast potential of neural network-based architectures.

2.4.4 Neural Networks (NN) Architectures

Baselines using one-hot encoding. We analyze various configurations of neural network architectures. First, we consider architectures that do not include an embedding layer. For these models, we begin with one-hot encodings of the categorical variables as described in proposed approach section, and directly feed these into the neural network, without the embedding layer for the categorical variables. Since the embedding layer is a linear layer, we also choose this to be a linear layer for a fair comparison. Here, the dimension that the first layer maps to becomes an important parameter. To analyze the impact of compression due to the action of the first layer, we analyze the following variants.

`NN_OHE_25 :` This variant introduces a compression to keep 75% of the input dimensions (i.e. a 25% compression). Formally for each categorical variable x_i^{cat}, we obtain the corresponding one-hot encoding $\boldsymbol{v}_i^{cat}$ as described in proposed approach section, and then

concatenate all one-hot encoded categorical representations to form $\boldsymbol{x}_{ohe}$ as

$$\boldsymbol{x}_{ohe} = \left[\boldsymbol{v}_1^{cat}; \boldsymbol{v}_2^{cat}; \ldots; \boldsymbol{v}_{n_{cat}}^{cat}\right]$$
$$\boldsymbol{x}_{lin} = \boldsymbol{W}_{cat}\boldsymbol{x}_{ohe} + b_{\text{cat}}$$

where $\boldsymbol{W}_{cat}$ and b_{cat} are the parameters of the linear layer. The following concatenated representation integrates the continuous features:

$$\boldsymbol{x} = [\boldsymbol{x}_{\text{lin}}; \boldsymbol{x}_{\text{cont}}] \tag{2.5}$$

Then, $\boldsymbol{x}$ is transformed using 2 fully connected layers as:

$$\boldsymbol{h}_0 = \sigma(\boldsymbol{W}_1\boldsymbol{x} + b_0), \text{ and } \boldsymbol{h}_1 = \sigma(\boldsymbol{W}_2\boldsymbol{h}_0 + b_1) \tag{2.6}$$

where $\boldsymbol{h}_0$ and $\boldsymbol{h}_1$ are the outputs of the first and second fully connected layers respectively. Finally, the output o is computed as:

$$\text{o} = \boldsymbol{W}_3\boldsymbol{h}_1 + b_2, \tag{2.7}$$

where $\boldsymbol{W}_1, \boldsymbol{W}_2$, and $\boldsymbol{W}_3$ are weight matrices and b_0, b_1, and b_2 capture bias corresponding to each layer.

`NN_OHE_50 :` Building upon `NN_OHE_25`, we experimented with a more aggressive compression strategy, retaining 50% of the initial dimensions post one-hot encoding. The objective is to extract crucial features.

Using Embeddings. Embeddings provide a dense representation of categorical variables. Instead of using a sparse matrix as in one-hot encoding, embeddings map each category to a point in a continuous vector space. This allows the model to place similar categories closer together, learning the relationships between categories during training.

`NN_EMB` **:** Drawing inspiration from the structure set by `NN_OHE_50` architecture, where categorical data was initially represented using one-hot encoding, in `NN_EMB_D_B` we pivot to a more sophisticated approach by replacing the one-hot encoded layer with an embedding layer. Given the input feature vector, $\boldsymbol{x}^{cat}$, and $\boldsymbol{x}^{cont}$ for categorical and continuous variables, respectively, we obtain the embeddings $\boldsymbol{z}_i$ for each categorical variable x_i^{cat} as described in the proposed approach section. The obtained embeddings are concatenated and then passed through a ReLU layer to introduce non-linearity, after which they are concatenated with continuous features to form $\mathbf{x}_0$ as shown in (2.4); the overall procedure is shown in Fig. 2.2. Then it is passed through 2 fully connected layers as in (2.6) with the input being $\boldsymbol{x}_0$.

`NN_EMB_CORE`**:** Expanding on the `NN_EMB` framework, this architecture introduces two key modifications: batch normalization and dropout layers. These changes were aimed at optimizing the learning process and enhancing the model's generalization capabilities.

`NN_EMB_D_B`**:** This architecture removes the non-linearity immediately following the embedding layer in `NN_EMB_CORE`. The approach intends to ascertain if the initial non-linearity was perhaps preemptively capturing or distorting the relationships within the data.

2.4.5 Evaluation Metrics

The primary metric used for evaluation of model performance is Root Mean Square Error (RMSE), defined as:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.8}$$

where y_i, and $\hat{y}_i$ are the actual and the predicted block times, respectively, and n is the number of observations. This choice aligns with the precedent set by notable studies in the field [46, 48, 76, 96, 12], which have also employ RMSE to gauge the prediction accuracy in airline operations and delays.

Further, to ascertain the statistical significance of the differences observed in the performance of the multiple models, an analysis of variance (ANOVA) was employed. We make use of p-values and confidence intervals (CIs) to determine whether the observed differences could have occurred by chance, with p-values below 0.05 indicating statistically significant differences.

In addition to RMSE, the Mean Absolute Error (MAE) was utilized for the sensitivity analysis, defined as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.9}$$

Unlike RMSE, MAE does not square the errors, thus providing a direct average error magnitude without unduly penalizing larger errors. This metric is useful for evaluating model performance in practical scenarios, reflecting the operational impact of prediction inaccuracies.

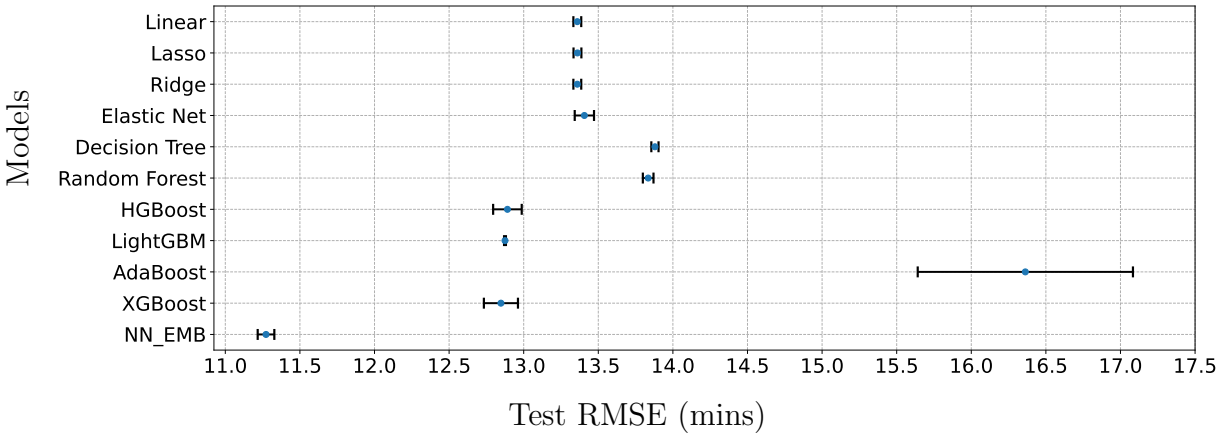


Figure 2.5: Confidence Intervals of Test RMSEs for baselines models and `NN_EMB`. The blue markers on the graph show the average Test RMSE of the models, while the error bars indicate their 95% confidence intervals.

Table 2.8: Test Performance Comparisons: Each Test RMSE represents the RMSE for the best set of hyperparameters, averaged across randoms three distinct random seeds.

Model Category	**Model Name**	**Test RMSE mins (S.D.)**
Baseline Models	Linear	13.359 (0.011)
	Lasso	13.360 (0.011)
	Ridge	13.359 (0.011)
	Elastic Net	13.407 (0.026)
	Decision Tree	13.880 (0.010)
	Random Forest	13.834 (0.014)
	HGBoost	12.891 (0.039)
	LightGBM	12.875 (0.003)
	AdaBoost	16.362 (0.290)
	XGBoost	12.847 (0.046)
Neural Networks	`NN_OHE_25`	20.651 (14.543)
	`NN_OHE_50`	12.622 (0.027)
	`NN_EMB_D_B`	13.211 (0.218)
	`NN_EMB_CORE`	13.556 (0.452)
	`NN_EMB`	**11.273 (0.022)**

Table 2.9: Tukey HSD Post-Hoc Comparisons: Showcases mean Test RMSE differences, 95% CIs, and Adj. P values, affirming the statistical significance of `NN_EMB`'s performance over baselines.

Comparison	**ΔMean**	**95% CI**	**Adj. P**
vs. Linear	2.087	(1.823, 2.350)	<0.0001
vs. Lasso	2.088	(1.824, 2.351)	<0.0001
vs. Ridge	2.087	(1.823, 2.350)	<0.0001
vs. Elastic Net	2.134	(1.870, 2.398)	<0.0001
vs. Decision Tree	2.607	(2.343, 2.871)	<0.0001
vs. Random Forest	2.562	(2.298, 2.825)	<0.0001
vs. HGBoost	1.618	(1.355, 1.882)	<0.0001
vs. LightGBM	1.602	(1.338, 1.866)	<0.0001
vs. AdaBoost	5.089	(4.826, 5.353)	<0.0001
vs. XGBoost	1.575	(1.311, 1.838)	<0.0001

2.5 Results and Discussion

In this section, we present and discuss the results of block time prediction models. Both the baselines and the neural network models were subjected to an identical experimental framework. The data was distributed into training, validation, and testing subsets, taking 64%, 16%, and 20% of the total respectively. Each model underwent training, validation, and testing across three distinct random seeds.

2.5.1 Comparing Baselines vs. Neural Networks

Table 2.8 presents the comparative performances of the baseline models and neural network architectures. The methods can be categorized into three distinct clusters: traditional regression approaches, tree-based algorithms, and neural network-based models. Linear,

Table 2.10: Best Parameters for the Different Baseline Methods

Method	**Best Parameters**
Linear	None
Lasso	alpha: 0.0122
Ridge	alpha: 0.0180
Elastic Net	alpha: 0.0155, l1_ratio: 0.8977
Decision Tree	max_depth: 5, min_samples_split: 5
Random Forest	n_estimators: 73, max_depth: 5
HGBoost	learning_rate: 0.1990, max_depth: 5, max_leaf_nodes: 31
LightGBM	learning_rate: 0.1923, num_leaves: 22, max_depth: 5
AdaBoost	n_estimators: 22, learning_rate: 0.1649
XGBoost	learning_rate: 0.1993, max_depth: 5, colsample_bytree: 0.8131

Lasso, and Ridge regressions display nearly identical performances with minimal variations. ElasticNet, while similar, shows slightly poorer performance and greater variability.This is reflected in their test error confidence intervals (CIs), which overlap as shown in Figure 2.5. Conversely, Decision Tree and Random Forest CIs overlap, indicating similar performances, as do the CIs for HGBoost, LGBM, and XgBoost. However, `NN_EMB`'s CI does not overlap with those of the other models, suggesting its performance differs significantly. The best parameters for each method are shown in Table 2.5.

On the neural networks front, both `NN_EMB` and `NN_OHE_50` consistently outperform all the baseline models. It is evident that `NN_EMB` holds a distinct edge over `NN_OHE_50`. With a test RMSE of 11.273 mins, it not only yields a better prediction accuracy compared to the 12.622 mins of `NN_OHE_50`, but also lesser variation in performance. Consequently, we select `NN_EMB` as the optimal neural architecture for benchmarking against the baseline models.

To determine the statistical significance of the improvement in prediction error by the NN_EMB, an Analysis of Variance (ANOVA) was conducted. This ANOVA compared the test RMSE across ten baseline models and the NN_EMB. The results revealed a significant effect of the model on Test RMSE ($F(10, 22) = 535.55$, $p<0.0001$), indicating marked differences in performance among the evaluated models. Upon confirming significant differences, Tukey's HSD post-hoc analysis was applied to perform pairwise comparisons of the means, with the Family Wise Error Rate (FWER) controlled at 0.05. Table 2.9 details these comparisons, presenting the differences in means (ΔMean) of test RMSE across seeds, the 95% confidence interval for each difference, and the adjusted p-value (Adj. P) for statistical significance. Given that the adjusted p-value is less than 0.05, we can conclude with 95% confidence that the differences in mean test RMSE between all baseline models and NN_EMB are statistically significant.

The better performance of NN_EMB can be attributed to the utilization of embeddings, particularly when tasked with encapsulating features of high cardinality. Traditional encoding schemes often struggle in this regard. Techniques such as one-hot encoding, for instance, operate under the assumption that each category is independent of others, suggesting no inherent similarity between varying categories. This can often be a simplistic and inaccurate representation, as it assumes categories to be orthogonal. Another notable advantage of using embeddings is the adaptability they offer. For instance, if a new airport is incorporated into the dataset, the learned embeddings can be fine-tuned to accommodate this new data, ensuring model robustness and adaptability.

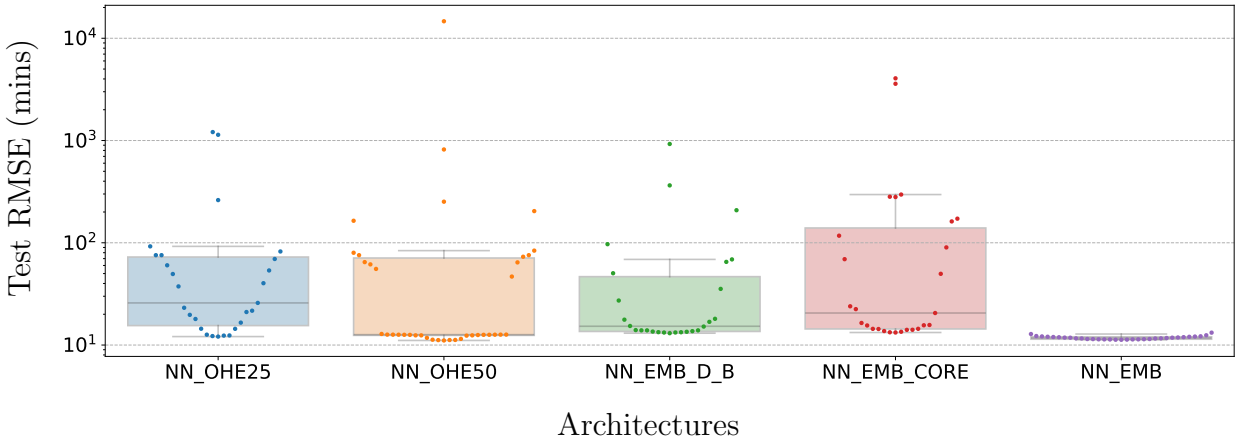


Figure 2.6: Hyperparameter Impact on Test RMSE: Each point denotes the mean Test RMSE for specific batch size and learning rate combinations, averaged over three distinct random seeds. Notably, `NN_EMB` exhibits consistent performance around an RMSE of approx. 11 mins across various hyperparameter settings.

Table 2.11: Overview of architecture modifications. Each Test RMSE represents the RMSE for the best set of hyperparameters, averaged across randoms three distinct random seeds. Read from top to bottom; symbols Δ, $-$, and $+$ denote change via replacement, removal, and cumulative addition, respectively.

Name	**Modifications**	**Test RMSE mins (S.D.)**
`NN_OHE_25`	Initial (OHE with 25% compression)	20.651 (14.543)
`NN_OHE_50`	Δ 50% Compression	12.622 (0.027)
`NN_EMB`	Δ Embeddings	**11.273 (0.022)**
`NN_EMB_CORE`	+ Dropouts + Batch Norms	13.556 (0.452)
`NN_EMB_D_B`	$-$ ReLU post embedding	13.211 (0.218)

2.5.2 Performance Evaluation and Ablations

We performed a series of modifications on the initial model employing OHE and 25% compression to understand the effect of adding, removing, and replacing elements of the architectures. The outcomes of these variations are summarized in Table 2.11. It is worth noting the initial performance of `NN_OHE_25` improves significantly when retaining only 50% of the initial dimensions after one-hot encoding. `NN_OHE_50` focuses on more critical features, thereby reducing noise and improving performance.

Transitioning from OHE to embeddings marked a significant shift, facilitating `NN_EMB_D_B` with a more nuanced representation of the data. This lead to marked improvement in performance, specifically the RMSE was enhanced from 12.622 minutes in `NN_OHE_50` to 11.273 minutes in `NN_EMB`, highlighting the efficacy of embeddings in capturing intricate data relationships.

However, when further modifications were introduced to `NN_EMB`, particularly dropouts and batch norms, to form the `NN_EMB_CORE` architecture, there was a discernible degradation in performance. This suggests that the additional regularizing layers introduce unnecessary complexities that outweigh their benefits for this 2-layer network. Notably, upon removing the non-linearity post the embedding layer, the performance of the `NN_EMB_D_B` architecture improved compared to `NN_EMB_CORE`. Thereby indicating that in this case, additional non-linearity does not help in capturing the relationships better.

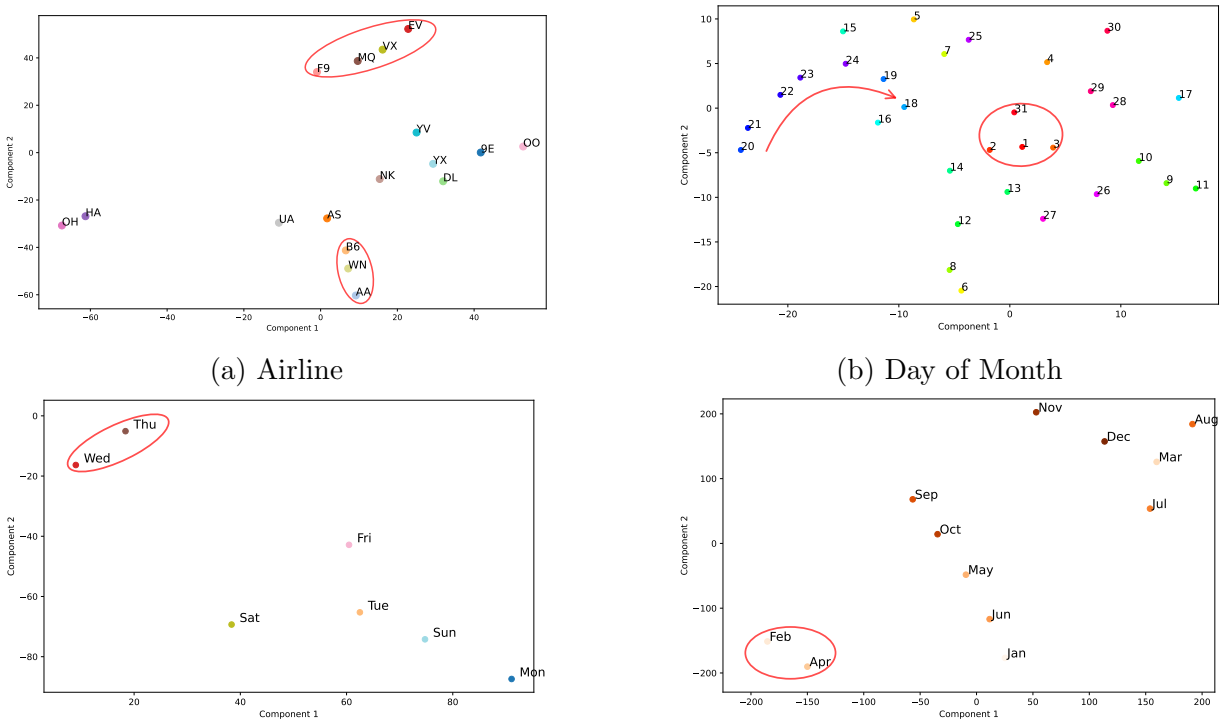
Fig. 2.8 compares the test performance across different architectures, where individual points on the graph represent distinct runs for each architecture. `NN_EMB` stands out with a uniform performance, demonstrating a test RMSE close to 11 minutes, irrespective of

batch size and learning rate variations. This indicates that it is a robust architecture that generalizes well across different configurations. On the contrary, NN_OHE_25 reveals a more varied performance spectrum.

Meanwhile, NN_EMB_D_B exhibits a notably tighter interquartile range in comparison to NN_OHE_25, NN_OHE_50, and NN_EMB_CORE. This suggests that NN_EMB_D_B offers more consistent performance, with less sensitivity to hyperparameter fluctuations. In contrast, NN_OHE_50 displays a pronounced clustering of data points around its first quartile, indicating inclinations towards a specific performance range. Among all the architectures, NN_EMB stands out not only for its superior performance but also for its remarkable consistency. The data points for this architecture are notably clustered along a line representing the lowest RMSE, indicating that it consistently achieves this performance regardless of minor variations.

2.5.3 Learned embeddings

In the implemented neural network architectures, NN_EMB_D_B, NN_EMB_CORE, and NN_EMB leverage embedding layers to process categorical variables. As the neural network is trained for block time prediction, these embeddings emerge as rich, multi-dimensional representations that capture intricate correlations within our dataset. For our visualization, we specifically utilize the embeddings from the best-performing run, notably from NN_EMB. To transform these embeddings into a more interpretable 2-dimensional space, we employ t-distributed Stochastic Neighbor Embedding (t-SNE) [82]. Essentially, t-SNE gauges similarities between points in the high-dimensional space and maps them into a



(a) Airline

(b) Day of Month

(c) Day of Week

(d) Month

Figure 2.7: Embeddings from the best-performing `NN_EMB` run. Red-highlighted clusters and arrows indicate consistent patterns and sequential trends, respectively, across varying learning rates and perplexity values.

lower-dimensional space, ensuring similar data points cluster together. This non-linear projection technique unveils patterns or clusters that might remain hidden in the original high-dimensional space. Utilizing this technique on our airline embeddings, we obtain a 2-D representation of airlines as depicted in Figure 2.7a annotated with IATA labels (see Table 2.1 for airline names). We observed that American Airlines (AA), Southwest Airlines (WN), and JetBlue Airways (B6) form a close-knit cluster, revealing shared operational characteristics among them. Additionally, the entities F9, MQ, YX, and EV form a separate group, which is distinctly highlighted in red in the figure. Further, We observed that American Airlines (AA), Southwest Airlines (WN), and JetBlue Airways (B6) form a close-knit cluster. Similarly, the entities Frontier Airlines (F9), American Eagle Airlines (MQ), Mesa Airlines (YX), and ExpressJet Airlines (EV) also form a cluster. Both these clusters are distinctly highlighted in red in the figure, indicative of analogous operational attributes amongst their respective members.

The embeddings for the days of the month, as seen in Figure 2.7b, reveal distinct cyclical patterns. The days 31,1,2 and 3 are closely clustered, with a clear sequential trend observed from days 20 to 24. These patterns are consistent across various t-SNE configurations, highlighting the cyclical nature of block time predictions throughout the month. Thursdays and Wednesdays, as shown in Figure 2.7c, cluster together across different perplexity values and learning rates. While the other days the other days are spread out, demonstrating a significant deviation from this cluster. February and April clustering together in the month embeddings, as showcased in Figure 2.7d, suggest similar flight patterns or operational behaviors during these months. However, there might be seasonal travel demands, holidays, or airline promotional activities that dictate the pattern visible and one one would need

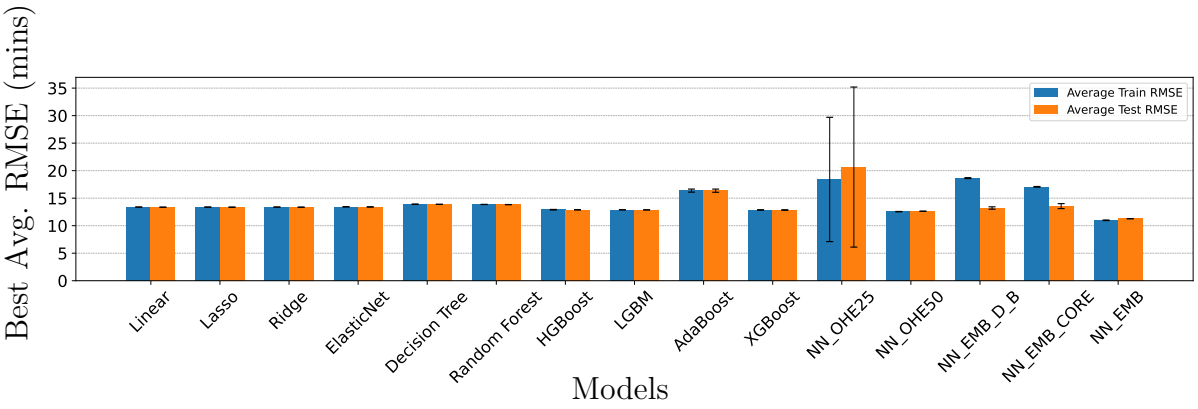


Figure 2.8: Comparative Train and Test Performances of models

to delve deeper into external factors to derive concrete insights.

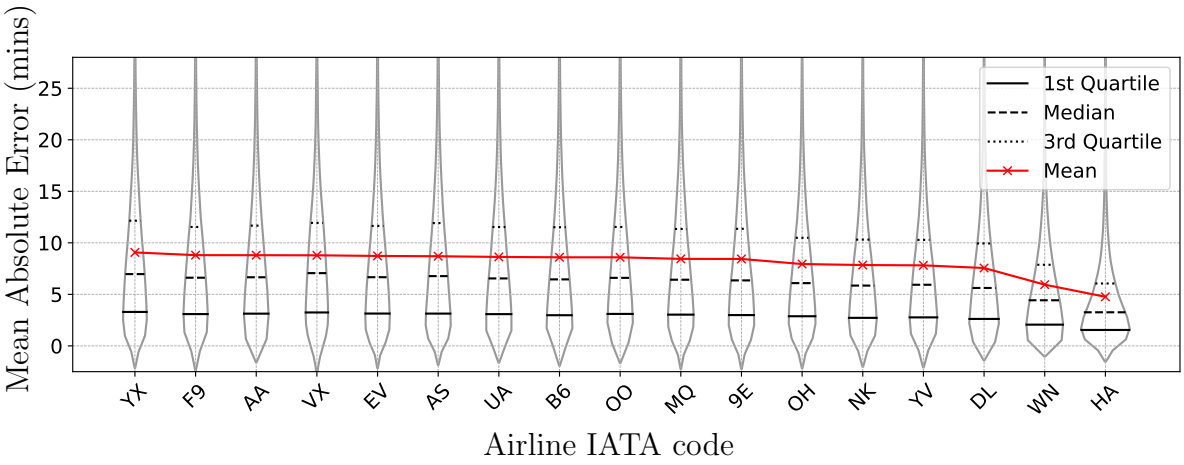


Figure 2.9: Prediction error across airlines

2.5.4 Sensitivity Analysis

We also carried out a sensitivity analysis for the proposed model: `NN_EMB` as detailed below.

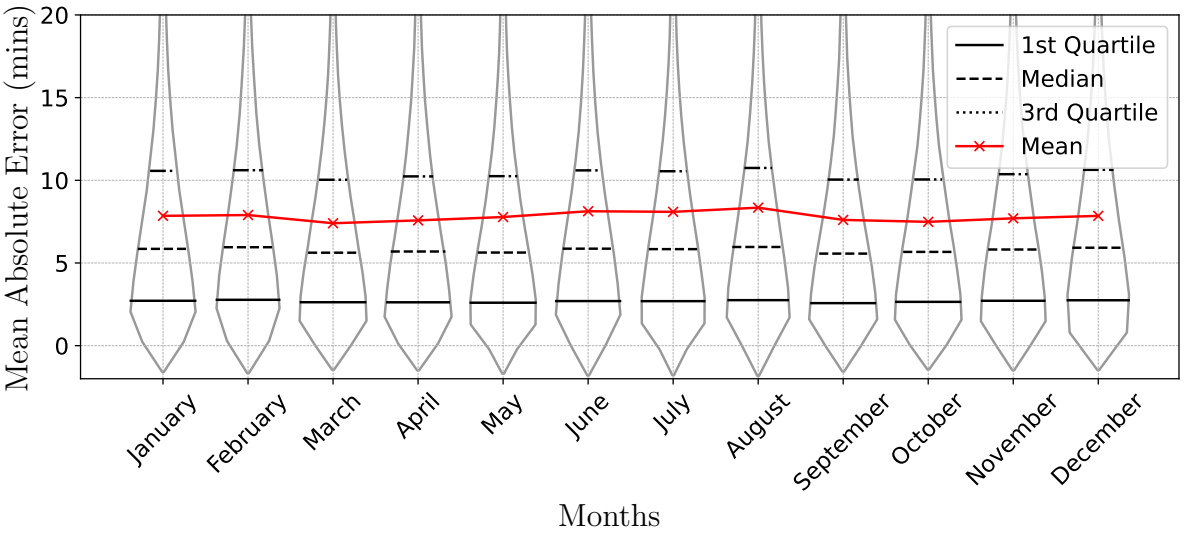


Figure 2.10: Prediction error across months

Error distribution across Airlines : Using the computed absolute errors in predictions on the test set across different airlines, it became evident that certain airlines, such as Hawaiian Airlines (HA), Southwest Airlines (WN), and Delta Airlines (DL) had relatively lower absolute errors. This suggests that the `NN_EMB` model was especially proficient at making predictions for these airlines. The distribution of the errors for each airline are shown in Figure 2.9.

Error distribution with temporal features : No specific patterns could be discerned from the analysis of errors across months. As seen in Figure 2.10 the distributions of errors are almost identical, similar observation was found for Day of Week. This shows that `NN_EMB` consistency in performance is unaffected by days or months.

2.6 Conclusion

Airline operations, by their very nature, operate under several uncertainties spanning various flight phases. For long-term planning, tasks such as predicting delays, determining block time, and assessing taxi-in and taxi-out durations, among others, thus take center stage for improving operational efficiencies. To this end, we focused on devising a predictive model tailored to address these challenges for a large scale data (a first to the best of out knowledge). Our exploration led us to the core insight: traditional encoding methods often fall short when handling the inter-dependencies and high cardinality of categorical variables. This prompted us to rigorously investigate the role of embeddings in improving forecasting capabilities. Among the neural network architectures explored, embedding-based neural network architecture – `NN_EMB`, demonstrated superior performance and consistency properties. In conclusion, our findings emphasize the role of embedding-based representation learning in predictive modeling, especially to capture the complex relationship in high cardinality features. Future explorations incorporating additional airline operations priors, and diverse data sources, remain exciting possibilities to further improve model robustness.

Chapter 3

Reliable Crew Pairing Optimization

Optimization tasks for planning and scheduling in various sectors, such as aviation and healthcare, are fraught with uncertainties that can easily disrupt operations and inflate costs. Column generation (CG) is a well-established solution methodology for large-scale optimization, often assuming deterministic operations [90]. Moreover, recent works generally assume that the uncertainties in operational parameters either have known distributions or can be described by uncertainty sets, which is often not true in practice [77, 7, 89]. This study introduces `SurvCG`, to the best of our knowledge, a first approach that leverages survival analysis to quantify uncertainties and integrate them into the CG algorithm.

In Chapter 2, we address the problem of creating realistic schedules by predicting flight segments like block time, taxi time and delays reliably. However, predicting flight segments alone is insufficient for planning, as we also need to understand if a crew will make a connection. It may seem that using distributions constructed from historical flight

arrival data could suffice, but this approach falls short in data-scarce scenarios where predicting the probability of on-time arrival is crucial. Therefore, we need a way to learn from historical flight arrival data to understand if connections will be made. To this end, we propose the use of time-to-event models, which, to the best of our knowledge, have never been used in aviation planning and scheduling. This integration enhances the robustness of crew pairing solutions by incorporating reliability predictions.

In this chapter, we cover crew pairing preliminaries and review related works before introducing our method for predicting flight connection reliability using survival analysis. We evaluate the predictive power of these models with a new metric and integrate the predictions into a reliable crew pairing framework, optimizing with a column generation algorithm. We then describe the process of instance generation and present our experimental and simulation results, demonstrating that reliability-based solutions significantly reduce total propagated delays in schedules.

3.1 Introduction

Optimization problems across diverse industries from aviation and logistics to manufacturing and healthcare, are exposed to the inherent uncertainties of operational environments. For instance, airline schedules are often disrupted by unpredictable weather, leading to delays and increased operational costs [11, 93]. Similarly, in manufacturing, production lines are frequently halted by equipment failures, [4] and in healthcare, fluctuating patient needs can strain resources unexpectedly [69].

In the context of airline operations, the crew pairing problem (CPP) directly confronts these uncertainties. CPP involves determining the most efficient assignment of crews to flights while complying with regulations and minimizing costs. The CPP is a highly constrained NP-hard combinatorial optimization problem [9, 66, 33]. The CPP is typically modelled as a set-covering or set-partitioning problem, and, the state-of-art methods use a column generation(CG) algorithm to solve its linear relaxation [98, 75].

Traditionally, the CPP is solved by optimizing planned costs based on known schedules, assuming fixed flight times and no delays, well before the actual day of operations [37, 36]. However, the actual costs can be considerably different from their planned values since these schedules are often disrupted by unforeseen issues such as maintenance problems or adverse weather conditions [7]. Such disruptions are not only costly for airlines but also lead to increased carbon emissions and reduced customer satisfaction [51].

While accurately estimating block times can enhance airlines' on-time performance and schedule reliability, it also leads to higher planned resource expenses, such as increased aircraft and crew utilization costs [78]. Additionally, even if a flight is predicted to be delayed, it can still be included in crew pairings if the schedule has sufficient buffer time to accommodate potential delays. Understanding the historical reliability of specific flight connections relative to their buffer times allows airlines to incorporate and quantify uncertainty in schedule building. By integrating this uncertainty into the planning process, airlines can create more resilient schedules that balance reliability with efficient resource utilization.

In the literature, uncertainty in the CPP is typically captured by flight delays, which

are either drawn from a set of potential expected delays bounded within specified intervals [66], or from a range based on one-year historical flight delay data [7]. This approach is limiting as it may not accurately reflect reality. Moreover, predicting expected delays is exceptionally difficult, if not impossible, since airlines at the planning stage lack information of what affects operations on the flight day. Furthermore, even a minor error in delay prediction can be the critical factor in whether a connection is made or missed.

To address these challenges, we employ survival analysis to quantify the uncertainty of connection between flights. Our approach, `SurvCG`, utilizes survival analysis to model time-to-event data and predict the likelihood of a flight arriving within the necessary time window, determining the reliability score of a connection based on historical flight characteristics. Furthermore, we augment the traditional cost function by incorporating a parametric function that enhances the costs according to the predicted reliability score. This adjusted cost reflects not only the planned schedule costs, but also the historical reliability of the flight connection based on the current schedule. By integrating this reliability score, which varies from 0 to 1. We then use a parametric cost function to penalize the less reliable connection so that they are not picked in the pairing optimization process.

Another integral part of our work is the introduction of a new metric, P-index, to assess the ability of survival models to predict reliability scores. This metric employs a non-parametric estimator to evaluate the accuracy of the model's probability estimations for specific times, ensuring more precise decision-making in selecting connections. Unlike traditional metrics in survival analysis, which focus on evaluating the discriminative ability of the model, P-index, provides a more nuanced assessment, facilitating better integration

of reliability into the scheduling process.

Our comprehensive framework encompasses several stages: generating instances, predicting reliability scores, finding optimal pairings using Column Generation (CG), and validating the results through data-driven simulation.

Contributions

1. A novel application of survival analysis to quantify uncertainties in flight schedules.
2. Modification of traditional cost functions in crew pairing optimization to include uncertainty measures.
3. A new metric to evaluate the predictive power of survival analysis models in the context of reliability scores.
4. Empirical validation through simulations that demonstrate the robustness of the proposed solution.

3.2 Preliminaries : Crew Pairing using Column Generation

The crew pairing problem (CPP) is formulated as a set partitioning problem, where $\mathcal{P}$ represents the set of all possible pairings. A pairing is defined as a sequence of flights, separated by rest periods, starting and ending at a crew base. The objective of the CPP is to select a subset of feasible pairings from the set, $\mathcal{P}$ such that each scheduled flight in the

set of flights $\mathcal{F}$ is included in exactly one chosen pairing. Each pairing $p \in \mathcal{P}$ has a cost c_p, and $a_{ip} = 1$ if flight $i \in \mathcal{F}$ is covered by pairing $p \in \mathcal{P}$ and 0 otherwise. The decision variable α_p equals 1 if pairing $p \in \mathcal{P}$ was selected, and 0 otherwise. The formulation is given as:

$$\text{[P]: minimize} \quad \sum_{p \in \mathcal{P}} c_p \alpha_p \tag{3.1}$$

$$\text{subject to} \quad \sum_{p \in \mathcal{P}} a_{ip} \alpha_p = 1 \quad \forall i \in \mathcal{F} \tag{3.2}$$

$$\alpha_p \in \{0, 1\} \quad \forall p \in \mathcal{P} \tag{3.3}$$

The objective is to minimize the total cost of the selected pairings while ensuring that each flight is covered by exactly one pairing. The constraint (3.2) ensures that every flight $i \in \mathcal{F}$ is included in exactly one pairing, guaranteeing full coverage of all scheduled flights. The constraint (3.3) enforces the binary nature of the decision variable α_p, indicating whether a particular pairing $p \in \mathcal{P}$ is selected or not.

Solving the linear programming relaxation of this problem [P] requires enumerating the set of all feasible parings $\mathcal{P}$. And, it is impractical to enumerate these due to the vast number of potential pairings; even in smaller airline instances, the number of feasible pairings can reach billions.

Instead of explicitly enumerating all feasible pairings, we use the column generation technique to solve the problem. It is a method where instead of considering all feasible pairings at once, the method starts with a subset of pairings and iteratively adds new pairings that can improve the objective function. This is done by solving a restricted

master problem (RMP) and a sub-problem (SP): the RMP, is solved first using a smaller subset of pairings while the sub-problem identifies potential new pairings that when added to the RMP could improve the optimal objective value of the RMP.

3.3 Related Works

Traditional crew pairing problem are typically deterministic and do not account for potential disruptions. However, airlines frequently encounter irregularities such as adverse weather conditions, carrier delays, and maintenance issues. To address these uncertainties, the crew pairing problem can be approached using various methodologies, including stochastic programming and robust optimization. Stochastic models incorporate the influence of random factors, aiming to leverage this information to develop solutions that are more robust to disruptions.

Yen and Birge [95] present a model that aims to reduce the propagated delay by minimizing the expected value of the total costs through a stochastic programming approach. However, while their model is robust, it faces significant computational challenges as the size of considered scenarios increases. Ionescu and Kliewer [53] presented a stochastic optimization model focused on crew swap opportunities to enhance flexibility. Duck et al. [35] formulated a combined stochastic crew scheduling and aircraft assignment model, integrating both aspects to improve operational resilience.

Robust models assume that problem parameters vary within a known interval and focus on the worst-case scenario, without making any assumptions on the distribution of these

parameters. This is done by the introduction of the robust terms that increase the constraints and variables – leading to a more computationally complex problem. Additionally, they require precise calibration of uncertainty parameters (robustness parameters); failing to do so can result in solutions that do not consistently align with real-world scenarios For example, Antunes et al. [7] developed a robust crew pairing model that accounts for the full complexity of crew pay structures and various dimensions of schedule robustness. Their model incorporates delay propagation and additional costs due to disruptions, aiming to generate crew pairing solutions that minimize total propagated delays and operational infeasibilities. Similarly, Lu and Gzara [66] proposed a robust optimization model that includes robust terms that account for the cost of deviation in the objective function and resource constraints. Their approach uses Lagrangian relaxation to solve the resulting nonlinear optimization problem, aiming to minimize crew costs under the worst-case scenario.

It is crucial to account for uncertainty when building a schedule for crew pairing optimization. Current methods often rely on historical delay distribution or predefined uncertainty sets, but these approaches fail to fully capture the unpredictable nature of airline operations. Survival analysis, a statistical method originally developed for medical research to analyze the time until an event occurs (such as death or survival) [52, 41], is well-suited for this purpose. Several studies have employed survival analysis for diverse tasks, including modelling of traffic incident duration ([70],[50],[65]), predictive maintenance tasks ([84],[83]).

While survival analysis is extensively applied in healthcare and engineering, its application in aviation is relatively novel. This method enables the study of delay characteristics

over time and provides insights into the factors associated with longer or shorter delays. However, there is limited research available that utilizes survival analysis for delay prediction. For instance, Wong et al. [91] employed a Cox proportional hazard (Cox PH) model to analyze both arrival and departure delays for a Taiwanese domestic airline, identifying key factors contributing to departure delays and using hazard ratios to assess the likelihood of recovering from delays. Kim et al. [59] compared departure delays between international and domestic flights in South Korea's air transportation system, utilizing Kaplan-Meier (KM) estimates and Cox PH models to reveal that international flights tend to experience minimal delays and that longer turnaround buffers are associated with fewer delays.

3.4 Predicting Flight Connection Reliability with Survival Analysis

This section introduces the methodology for using survival analysis to predict the reliability of flight connections. It explains the concept of Time to Event (TTE) and how it is used to model the likelihood of a flight arriving within the necessary time window for a subsequent connection. The section outlines the features used to represent each flight and describes the survival function and its estimation using the Kaplan-Meier estimator and parametric models. This section aims to provide a detailed understanding of the survival analysis techniques employed in this study and their application to predicting flight connection reliability.

3.4.1 Methodology for Reliability Prediction

In this study, we use survival analysis to determine the probability that a flight i can connect to a subsequent flight j departing at the scheduled departure time SDT_j. The Time to Event (TTE) for a flight i is defined as $TTE_i = AAT_i - SDT_i$, where, SDT_i is the scheduled departure time and AAT_i is the actual arrival time.

Example: Consider a flight from New York (JFK) to Los Angeles (LAX) with a scheduled departure time (SDT_i) of 08:00 AM EST and an actual arrival time (AAT_i) of 11:30 AM PST. Convert the scheduled departure time to PST: SDT_i (PST) = 08:00 AM EST− 3 hours = 05:00 AM PST. The TTE for this flight is: TTE_i = 11:30 AM PST−05:00 AM PST = 6.5 hours.

Each flight i is represented by a tuple $(\mathbf{x}_i, y_i, D_i)$, where $\mathbf{x}_i$ includes features like origin, destination, and aircraft type/model, y_i is the TTE, and D_i is the event indicator. The event D_i is 1 if the flight lands by time t, otherwise 0. The survival function for a flight with feature vector $\mathbf{x}$ at time t is given by:

$$\begin{aligned} S(t \mid \mathbf{x}) &= \mathbb{P}(\text{flight landing beyond time } t \mid \text{flight's feature vector is } \mathbf{x}) \\ &= \mathbb{P}(T > t \mid \mathbf{X} = \mathbf{x}) \\ &= 1 - \mathbb{P}(\text{flight landing within time } t \mid \text{flight's feature vector is } \mathbf{x}) \\ &= 1 - \mathbb{P}(T \leq t \mid \mathbf{X} = \mathbf{x}) \end{aligned} \tag{3.4}$$

Here, $\mathbf{X}$ and T are random variables corresponding to a feature vector $\mathbf{x}$, and the associated Time to Event (TTE).

Predicting the reliability r_{ij} of two flights, i and j in a sequence, involves estimating the probability that the flight i (with feature vector $\mathbf{x_i}$) will arrive in time for the subsequent flight j to depart at SDT_j. Specifically, we calculate the probability that i will land in time t_q from the SDT_i. Using 3.4 we can estimate r_{ij} by querying the estimated survival function $\widehat{S}(t \mid \mathbf{x_i})$ as

$$r_{ij} = \mathbb{P}(T \leq t_q \mid \mathbf{X} = \mathbf{x_i}) = 1 - \widehat{S}(t_q \mid \mathbf{x_i}), \text{where } t_q = SDT_j - SDT_i - \delta_{min} \tag{3.5}$$

The survival function can be estimated using the non-parametric Kaplan-Meier estimator [57]:

$$\widehat{S}(t) = \prod_{i=1}^{L} \left(1 - \frac{d_i}{n_i}\right)^{1\{t_i \leq t\}}, \text{where } d_i = \sum_{j=1}^{n} 1\{y_j = t_i\} D_j, n_i = \sum_{j=1}^{n} 1\{y_j \geq t_i\} \tag{3.6}$$

where $t_1, t_2, \ldots, t_L$ are unique times of landing of flight, d_i denotes the number of flights that landed at time t_i, n_i be the number of flights that could possibly land at time t_i and $1\{\cdot\}$ is the indicator function. The Kaplan Meier estimator 3.6 can be used to derive the $\widehat{S}(t)$ from empirical data at observed event times. However, when predicting for time points where no events have been observed, it becomes a limitation. Hence, we use parametric models that can provide continuous survival estimates that extend beyond the observed event times.

The most popular survival analysis model, Cox proportional hazards model [29] allows for examining the effect of the covariates/features on the TTE. The Cox model can be

expressed by a hazard function $h(t)$, estimated as follows:

$$h(t \mid \mathbf{x}) = h_0(t) \exp\left(\beta^T x\right) \quad \text{– } \textit{proportional hazards assumption} \tag{3.7}$$

where h_0 is the non-parametric baseline hazard, and $\beta \in \mathbb{R}^d$ is the feature weighting vector. Both h_0 and β are parameters that need to be estimated. Even when the feature vector $\mathbf{x}$ varies, the hazard function remains proportional to $h_0(t)$. By integrating the hazard function $h(t \mid \mathbf{x})$ with respect to time, we obtain the cumulative hazard function:

$$H(t \mid \mathbf{x}) = \int_0^t h(\tau \mid \mathbf{x})\, d\tau \tag{3.8}$$

This cumulative hazard function allows us to calculate the survival function:

$$S(t \mid \mathbf{x}) = \exp(-H(t \mid \mathbf{x})) = [S_0(t)]^{\exp(\beta^T \mathbf{x})} \tag{3.9}$$

The proportionality assumption implies that all the possible survival curves are powers of the baseline curve $S_0(t)$. This assumption is very restrictive, since survival curves for different $\beta^T x$ would have to be either uniformly better or worse than $S_0(t)$. However, this may not hold for real world problems, in our context the survival curves for various routes (origin-destination pairs) often intersect, indicating that the proportional hazards assumption is violated. This observation highlights the need for a more flexible model where the proportionality constraint can be relaxed.

Table 3.1: Summary of Survival Analysis Methods

Method	**Model Type**	**Prop. Constraint**	**Main Benefit**
Cox Proportional Regression [29]	Continuous	Yes	Most Interpretable
DeepSurv [58]	Continuous	Yes	Handles non-linearity (Uses NN for reg Cox)
Cox-Time [62]	Continuous	No	Extends Cox Reg beyond prop. hazards
Cox-CC [62]	Continuous	Yes	Proportional version of Cox-Time
Random Survival Forests [54]	Continuous	No	Handles interactions and non-linearity
DeepHit [64]	Discrete	No	Best discriminative ability (C-index)

3.4.2 Evaluating the Survival Analysis Model

The most commonly used evaluation metric for survival models is concordance index, also known as C-index [49]. It quantifies the rank correlation between the actual Time to Event (TTE) and the model's predictions. However, the C-index does not capture nuances in model performance over time, making it particularly less effective for evaluating models that violate the proportional hazards' assumption (see 3.8.1 for more details).

Hence, one of the metric we use is the time-dependent C-index given by Antolini et al. [6], similar to the approach used by Kvamme et al. [62] to evaluate survival models. This metric estimates the probability that the predicted TTE for flight i is less than that for flight j, given that flight i's TTE is less than or equal to flight j's TTE.

$$C^{td} = P(\hat{S}(T_i \mid x_i) < \hat{S}(T_j \mid x_j) \mid T_i \leq T_j, D_i = 1). \tag{3.10}$$

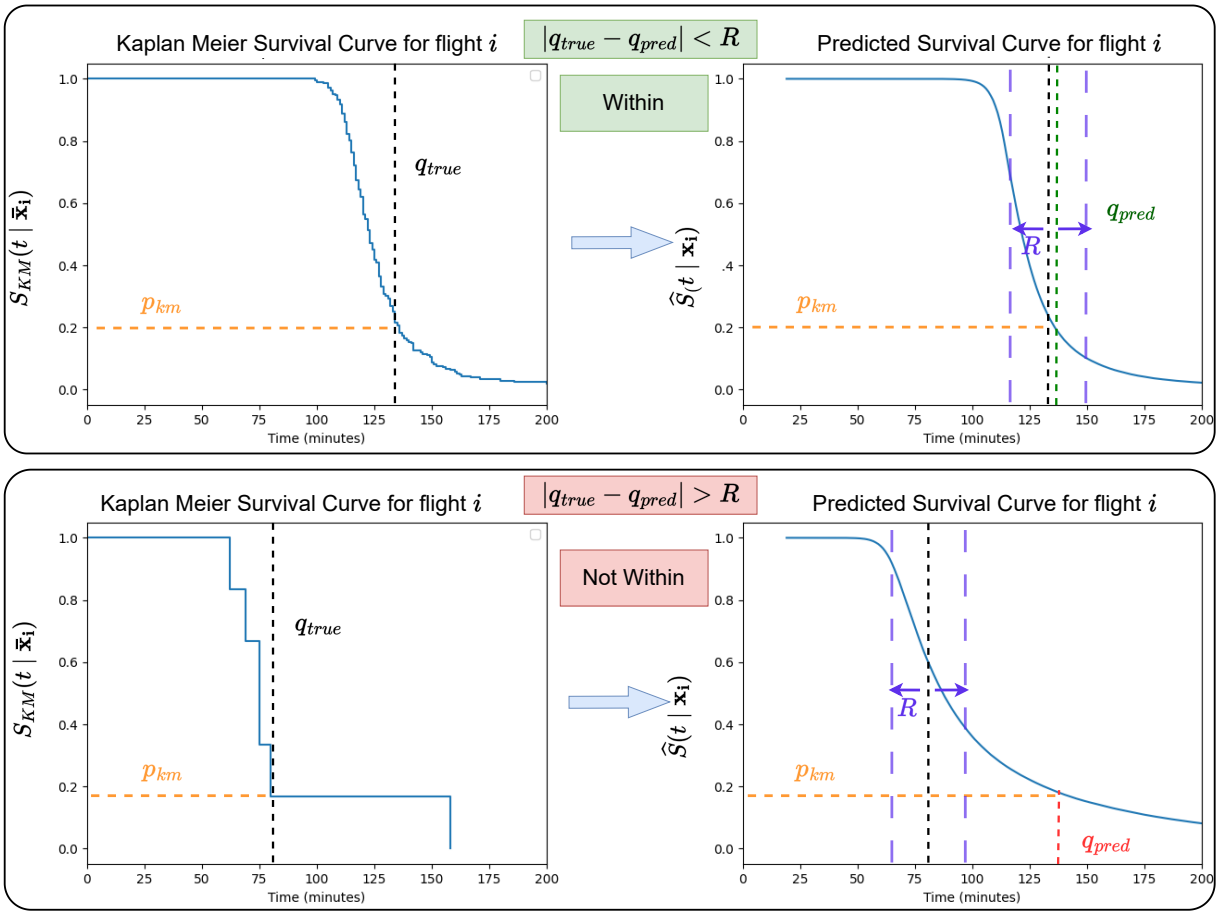


Figure 3.1: Illustration of P-index Calculation: The top panels show q_{pred} within the acceptable range R of q_{true}, indicating a successful prediction. The bottom panels depict q_{pred} outside R, indicating an unsuccessful prediction. The P-index is the ratio of successful predictions (within R) to the total predictions across all query times.

where $\hat{S}(t \mid x)$ denotes the estimated survival function, T_i and T_j are the TTEs for flights i and j, respectively, and x_i and x_j are their corresponding feature vectors. $D_i = 1$ indicates that the flight i has landed. The time dependent C-index (C^{td}) serves as a discriminative metric that assess the ability of the model to correctly predict the order of TTE over time points.

However, our interest extends beyond the discriminative ability of the model; we focus on the accuracy with which probabilities derived from the predicted survival curve are mapped to specific times. As illustrated by the equation 3.5, the reliability of a connection r_{ij} is obtained by querying time t_q against the estimated survival function $\widehat{S}(t \mid \mathbf{x})$. This mapping is extremely crucial, as errors in the probability estimation for specific times has significant impact on making the decision to choose a connection over the other.

To evaluate the accuracy of this mapping, we introduce a new metric called P-index that measures the quality of predictions, defined as:

$$P\text{-index} = \frac{\sum_{i=1}^{n} \mathbf{1}(|q_{\text{pred},i} - q_{\text{true},i}| \leq R)}{n} \tag{3.11}$$

where $\mathbf{1}$ is an indicator function that evaluates to 1 if the predicted query time $q_{\text{pred},i}$ and the actual query time $q_{\text{true},i}$ are within a predefined error margin R, and 0 otherwise. Here, $q_{\text{true},i}$ is the estimate of the event time for flight i based on the non-parametric Kaplan-Meier estimator $\hat{S}_{KM}(q_{\text{true}}|\bar{x}_i)$ for all $\bar{x}_i \subseteq x_i$. The $q_{\text{pred},i}$ is determined by the time at which the estimated survival probability by the model $\hat{S}(t|x_i)$ matches the Kaplan-Meier survival probability $p_{\text{KM}} = \hat{S}_{KM}(q_{\text{true}}|\bar{x}_i)$. Figure 3.1 illustrates cases where $q_{\text{pred},i}$ is within the predefined error margin R (denoted as "within") and not within ("not within").

Additionally, to quantify the deviations we compute the Mean Absolute Error (MAE) for predictions within, above, and below the actual query times:

$$\text{MAE}_{\text{within}} = \frac{\sum_{i=1}^{n} |q_{\text{pred},i} - q_{\text{true},i}| \cdot \mathbf{1}(|q_{\text{pred},i} - q_{\text{true},i}| \leq R)}{N_{\text{within}}} \tag{3.12}$$

$$\text{MAE}_{\text{above}} = \frac{\sum_{i=1}^{n} |q_{\text{pred},i} - q_{\text{true},i}| \cdot \mathbf{1}(q_{\text{pred},i} > q_{\text{true},i})}{N_{\text{above}}} \tag{3.13}$$

$$\text{MAE}_{\text{below}} = \frac{\sum_{i=1}^{n} |q_{\text{pred},i} - q_{\text{true},i}| \cdot \mathbf{1}(q_{\text{pred},i} < q_{\text{true},i})}{N_{\text{below}}} \tag{3.14}$$

where N_{within} is the number of predictions within the acceptable range R, N_{above} is the number of predictions above q_{true}, and N_{below} is the number of predictions below q_{true}. Having established how to predict and evaluate the reliability score of a connection using survival analysis, we now turn to integrating these reliability scores into the crew pairing optimization process.

3.5 Reliable Crew Pairing Framework

This section presents the core contribution of the study: the integration of survival analysis into the column generation approach for solving the Crew Pairing Problem (CPP). It describes the modeling of CPP as a set partitioning problem, the formulation of the reliability-integrated cost function, and the overall framework for generating and evaluating optimal crew pairings. The section also introduces the concept of nominal versus reliable cost functions and the impact of reliability scores on the cost function. This section aims to demonstrate how the proposed `SurvCG` approach enhances the robustness of

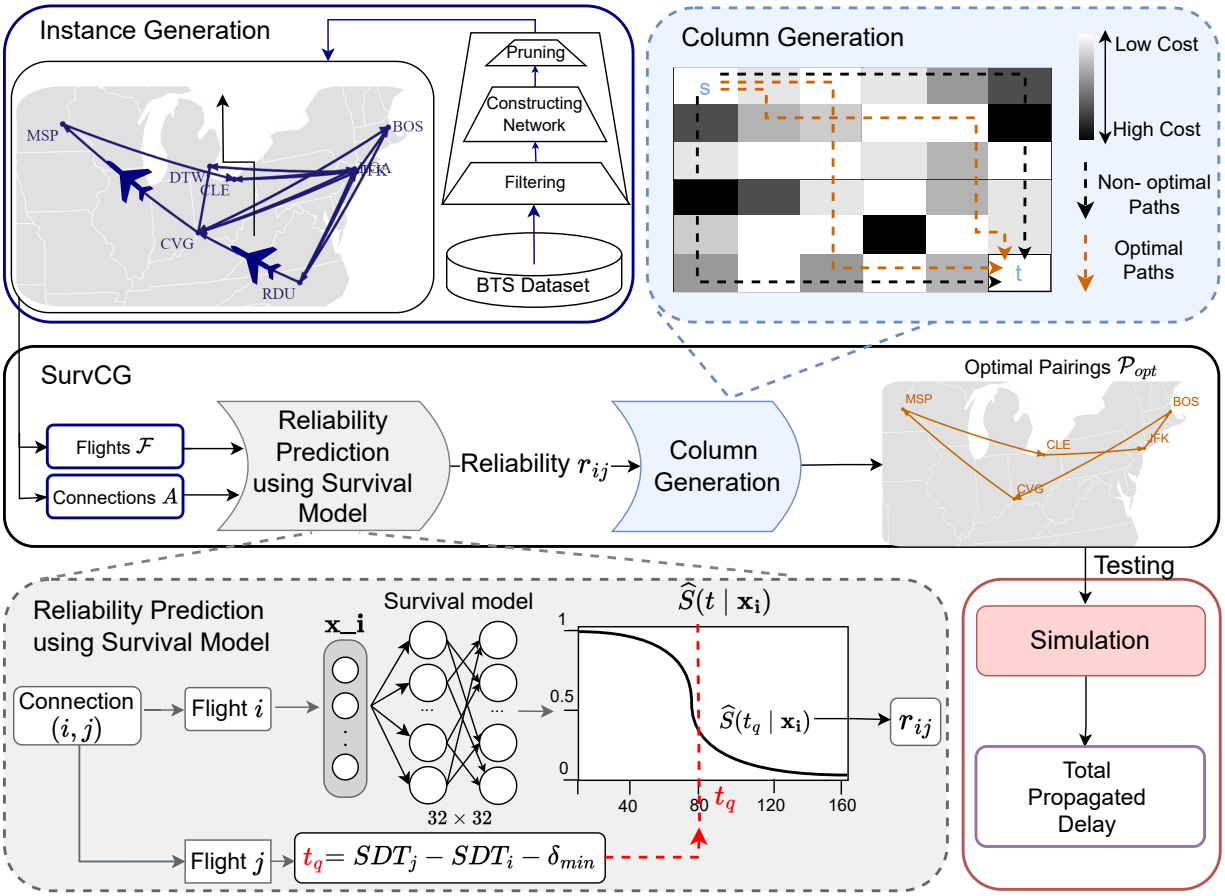


Figure 3.2: Block diagram for Reliable Crew Pairing Framework

crew pairing solutions by incorporating reliability predictions.

3.5.1 Survival-based Column Generation (SurvCG)

Building on the set-partitioning formulation discussed in Section 3.2, we generalize the problem for K crews with two significant modifications. Firstly, the cost of pairing p for crew k, c_p^k, is now represented by $\phi(c_p^k)$, incorporating reliability through a reliability cost function ϕ. Secondly, we modify the constraints to allow deadhead flights, enhancing the flexibility of crew scheduling.

To model the crew pairing problem for a set of K crews., we use a flight network $G = (N, A)$; where N represents all possible nodes, including the origin O, destination D, and flight nodes $N \setminus \{O, D\}$. Each flight node is defined by its origin and destination airports, departure, and arrival times. The set A comprises three types of arcs: arc (o_k, i) if crew k can start its schedule with flight i, arc (i, d_k) if crew k can end its schedule with flight i, and arc (i, j) for sequential flights i and j where the minimums sit time is met. The set P^k represents all possible pairings for crew k, where each pairing p in P^k has an associated cost c_p^k. The binary parameter a_{ip} is 1 if flight i is covered by pairing p and 0 otherwise, ensuring accurate flight coverage. The decision variable x_p^k is binary, 1 if pairing

p is selected for crew k and 0 otherwise.

$$\text{[MP]: minimize} \quad \sum_{k \in K} \sum_{p \in P^k} \phi(c_p^k) x_p^k \tag{3.15}$$

$$\text{subject to} \quad \sum_{k \in K} \sum_{p \in P^k} a_{ip} x_p^k \geq 1 \quad \forall i \in \mathcal{F} \tag{3.16}$$

$$\sum_{p \in P^k} x_p^k = 1 \quad \forall k \in K \tag{3.17}$$

$$x_p^k \in \{0, 1\} \quad \forall k \in K, \forall p \in P^k \tag{3.18}$$

The objective function (3.15) minimizes the total cost of selecting K crew pairings using the reliability-integrated cost function $\phi(.)$. Constraints (2) ensure that each flight i in the set of flights $\mathcal{F}$ is covered by at least one pairing. Constraint (3) makes sure that each crew k is assigned exactly one pairing from their respective set P^k.

The function ϕ, modifies the cost to reflect not only the time spent (assumed to be cost) but also the reliability of the flights within a chosen pairing. The cost c_p^k for each pairing p and crew k, and its reliability-integrated cost, are defined on the arcs A of the flight network as:

$$c_p^k = \sum_{(i,j) \in p} c_{ij}, \quad \forall p \in P^k; \quad \phi(c_p^k) = \sum_{(i,j) \in p} (\phi_{ij}(c_{ij})), \quad \forall p \in P^k \tag{3.19}$$

where c_{ij} represents the cost of including arc (i, j) in the selected pairing p.

3.5.2 Nominal vs Reliable Cost Functions

The nominal cost of including an arc (i, j), denoted as c_{ij} is calculated based on the elapsed time of flight i and the sit time between the two consecutive flights i and j, expressed as:

$$c_{ij} = (c_i^e + \alpha \cdot c_{ij}^c), \text{where } c_i^e = SET_i, c_{ij}^c = SDT_j - SAT_i \tag{3.20}$$

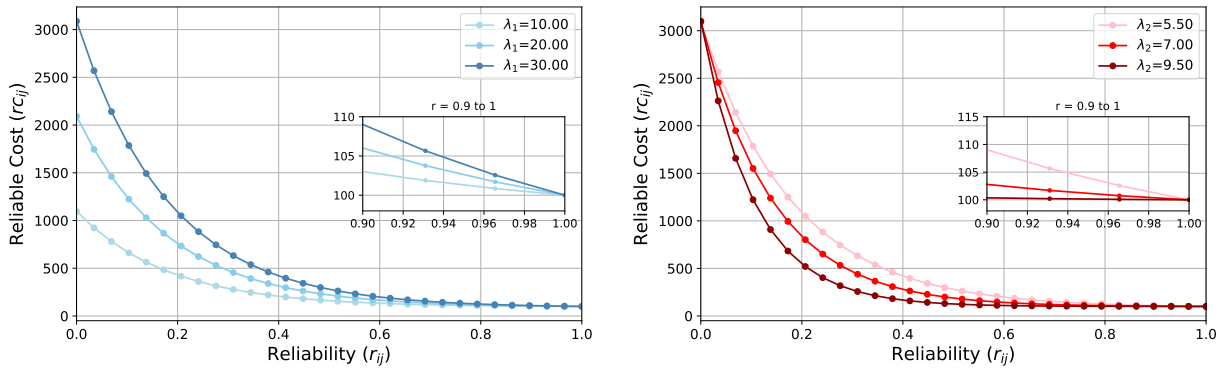
Here, α is the scaling factor for sit times. A higher α penalizes longer sit times, thus making pairings with shorter layovers between flights more attractive from a cost perspective. Specifically, α reflects the operational priorities, such as reducing crew downtime or enforcing extended layovers.

The nominal cost does not take into account the reliability r_{ij} of two flights i and j in a sequence. And hence ignores the certainty of the connection being disrupted by delays. To ensure that the pairings are robust against real-world operational variability, we introduce a reliability integrated cost function ϕ. This exponential function modifies the nominal costs to incorporate a heavier penalty for the less reliable flights.

The new reliability-integrated cost can be expressed as:

$$rc_{ij} = \phi_{ij}(c_{ij}) = c_{ij}(\lambda_1 e^{-\lambda_2 r_{ij}} + 1) - va_{ij}, \text{where } va_{ij} = c_{ij}(\lambda_1 e^{-\lambda_2}) \tag{3.21}$$

where λ_1 is the parameter that adjusts how significantly the reliability impacts the adjusted costs. A higher value of λ_1 increases the sensitivity of the cost adjustment to changes in r_{ij}. Meanwhile, λ_2 the parameter controls the rate of exponential decay, how rapidly the costs escalate as r_{ij} decreases. A larger λ_2 results in a steeper decay curve, which more



(a) Varying λ_1 for $\lambda_2 = 5.5$

(b) Varying λ_2 for $\lambda_1 = 30$

Figure 3.3: Effects of varying λ_1 and λ_2 on the Reliability Integrated Cost Function

aggressively penalizes lower r_{ij} values. The term va_{ij} is the vertical adjustment, shifts the cost such that reliability-integrated cost does not deviate from the nominal cost under ideal conditions ($r_{ij} = 1$). The effects of varying λ_1 and λ_2 on the reliability-integrated cost function are illustrated in Figure 3.3.

3.5.3 Overall Framework

An overview of our reliable crew pairing framework is illustrated by Figure 3.2. The procedure initiates with instance generation where flights and potential connections spanning over a fixed period of time are used to construct a network, as discussed in Section 3.6.2. The next step within the `SurvCG` leverages a survival model to predict a reliability score r_{ij} for every connection from flights i to j, as depicted in the Reliability Prediction using Survival Model block of Figure 3.2. This model is trained on historical data, assessing the likelihood of successful connections.

Next, we employ column generation (CG) to solve the Crew Pairing Problem (CPP),

where potential crew pairings are generated and evaluated using either nominal or reliability-integrated cost function. This process results in an optimal set of pairings $\mathcal{P}_{opt}$. The details of the column generation are described in Section 3.2. In the final phase, Testing, the framework simulates each $p \in \mathcal{P}_{opt}$ to assess the practical effectiveness of our solutions. This simulation evaluates total propagated delay and examines the

This thorough evaluation ensures that the theoretical optimizations hold under practical scenarios, providing actionable insights into operational reliability. The simulation process is explained in Section 3.6.3.

3.6 Experimental Results

This section presents the empirical validation of the proposed `SurvCG` approach. It describes the dataset used for training and testing the survival analysis models, the hyperparameter tuning process, and the performance comparison of different models. The section details the instance generation process, the setup for comparing solutions, and the simulation framework used to evaluate the practical effectiveness of the solutions. This section aims to provide evidence of the robustness and efficacy of the `SurvCG` approach through quantitative metrics and simulation outcomes.

3.6.1 Survival Model Testing and Reliability Prediction

Dataset: We train the survival analysis model using the 2019 Bureau of Transportation Statistics (BTS) On-Time Performance dataset for flight operations of Endeavor Air [21].

Table 3.2: Survival model comparison

Model	lr	b	C^{td}	P**-index**	$\mathbf{MAE_{within}}$	$\mathbf{MAE_{above}}$	$\mathbf{MAE_{below}}$
CoxPH	0.001	16	0.787	0.494	8.057	38.594	10.261
DeepSurv	0.0001	64	0.795	0.532	8.440	48.374	8.090
CoxTime	0.0001	32	**0.807**	**0.911**	**4.839**	**21.227**	**4.730**

There are 97294 total flights, of which 80% are used for training and 20% are used for testing. Specifically, for each flight i in the data, the feature set $\mathbf{x}_i$ consists of spatio-temporal attributes and aircraft information including the day of week, aircraft age/model, origin, destination, and scheduled departure/arrival.

Model Implementation: We implement DeepSurv [58], CoxPH [29], and CoxTime [62] models. DeepSurv and CoxPH assumes proportionality, whereas CoxTime does not.

Hyperparameter Tuning and Performance Comparison: We perform hyperparameter tuning for each model to find an optimal combination of learning rate lr, and batch size b. For evaluation, we primarily consider the P-index(3.11) for evaluating model performance and also consider the C^{td} (3.10). The optimal hyperparameter configurations and evaluation results are shown in Table 3.2.

Reliability Prediction: We use CoxTime [62] for reliability prediction due to its superior performance across all quantitative metrics. Notably, CoxTime achieved a P-index of 0.911, which is indicates its robust capability to accurately predict survival functions. Further, this precision is critical for predicting whether a delayed flight will still meet the minimum sit time (δ_{min}) for the crew to transition to the next flight.

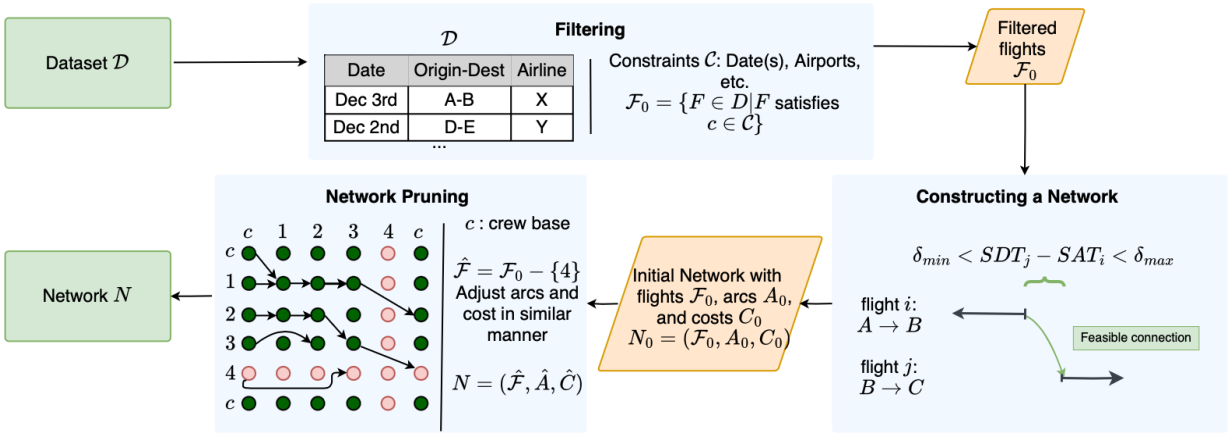


Figure 3.4: Block diagram for Instance Generation: The process begins with a dataset $\mathcal{D}$ filtered by constraints $\mathcal{C}$ to produce filtered flights $\mathcal{F}_0$. An initial network $\mathcal{N}_0$ is constructed with these flights, and further pruned to create the final network $\mathcal{N}$ with adjusted arcs and costs.

3.6.2 Instance Generation

To set up a crew pairing experiment, we first generate an instance of crew operations consisting of a closed network of flights and connections over a fixed period of time. The crew starts and finishes at a specified crew base. Specifically, an instance is described by a network N containing a set of flights $\hat{\mathcal{F}}$, connections $\hat{A}$, and costs $\hat{C}$. We create an instance for December 2-5, 2019, using the flight operations of Endeavor Air between all the airports in the network on these dates, with John F. Kennedy International Airport (JFK) as the crew base. The following subsections explain the process in greater detail.

Filtering: During the filtering phase, we specify space and time constraints to select relevant flights from the dataset. The constraints include the specific dates and the crew base for starting and ending operations. This phase aims to narrow down the vast dataset to a manageable subset that is relevant to the instance we want to create. By applying

these constraints, we extract a filtered set of flights $\mathcal{F}_0$ from the dataset $\mathcal{D}$, such that $\mathcal{F}_0 = \{i \in \mathcal{D} | i \text{ satisfies } \mathcal{C}\}$. This step ensures that only flights within the specified dates and that either start or end at the crew base are included in the instance.

Constructing a Network: In the network construction phase, we form connections between the filtered flights. The connections represent possible pairings of flights that a crew can operate within the given constraints. The network includes nodes for the crew base and connections between flights that are feasible based on time constraints. Specifically, a connection between flight i and flight j is feasible if the destination of i matches the origin of j, and the time difference between the scheduled departure time of flight j and the scheduled arrival time of flight i falls within the allowable sit time range. The cost for a connection is calculated as $SDT_j - SAT_i + SET_i$, where SDT is the scheduled departure time, SAT is the scheduled arrival time, and SET is the scheduled elapsed time. For the detailed algorithm to construct the network, see Section 3.8.2.

Network Pruning: The pruning phase ensures that the network remains practical and feasible for crew pairings. During this phase, we remove redundant or infeasible flights and connections that do not contribute to viable pairings. This is done by identifying and retaining only those flights that can form a continuous path from the origin to the destination crew base. Flights that do not participate in any such path are pruned out. For the detailed algorithm to prune the network, see Section 3.8.2.

3.6.3 Comparison of Solutions

Given an instance, we run the CG algorithm for the nominal cost function with $\alpha = 2$, as defined in Equation (3.20), meaning that the sit time is penalized twice compared to the elapsed time. For the reliable solutions, we use two configurations with $\lambda_1 = 10, \lambda_2 = 3$ and $\lambda_1 = 20, \lambda_2 = 4$, as defined in Equation (3.21). We obtain the optimal crew pairings $\mathcal{P}_{opt}$ for each crew $k \in K$.

Cost comparison: We compute the cost of the solutions based on the nominal arc cost (3.20) with $\alpha = 1$ to ensure nominal and reliable solutions are comparable. The constraint (3.16) allows for a flight node to be covered more than once, resulting in deadheading, where crew members are transported as passengers, incurring additional costs. Reliable solutions show an increase in deadhead flying costs by up to +5.93% compared to the nominal solution. However, these reliable solutions exhibit significant reductions in deadhead connection costs, with decreases up to -13.58%. This results in marginally lower total costs by up to -0.003%, as seen in Table 3.3.

Table 3.3: Comparison of Deadheading and Total Costs for Nominal and Reliable solutions ($\alpha = 2$). Changes in costs (increase or decrease, in parentheses, are percentages) compared to nominal. Headers: DFC - Deadhead Flying Cost, DCC - Deadhead Connection Cost, TFC - Total Flying Cost, TCC - Total Connection Cost, TC - Total Cost. Solutions: N - Nominal, R1 - $\lambda_1 = 10, \lambda_2 = 3$, R2 - $\lambda_1 = 20, \lambda_2 = 4$

Soln.	**DFC**	**DCC**	**TFC**	**TCC**	**TC**
N	2948.0	17436.0	29233.0	214933.0	244166.0
R1	3123.0 (5.93)	16155.0 (-7.35)	29408.0 (0.60)	214750.0 (-0.09)	244158.0 (-0.003)
R2	3123.0 (5.93)	15015.0 (-13.58)	29408.0 (0.60)	214750.0 (-0.09)	244158.0 (-0.003)

The number of deadheads in the nominal $\alpha = 2$ solution is 28, based on the flight

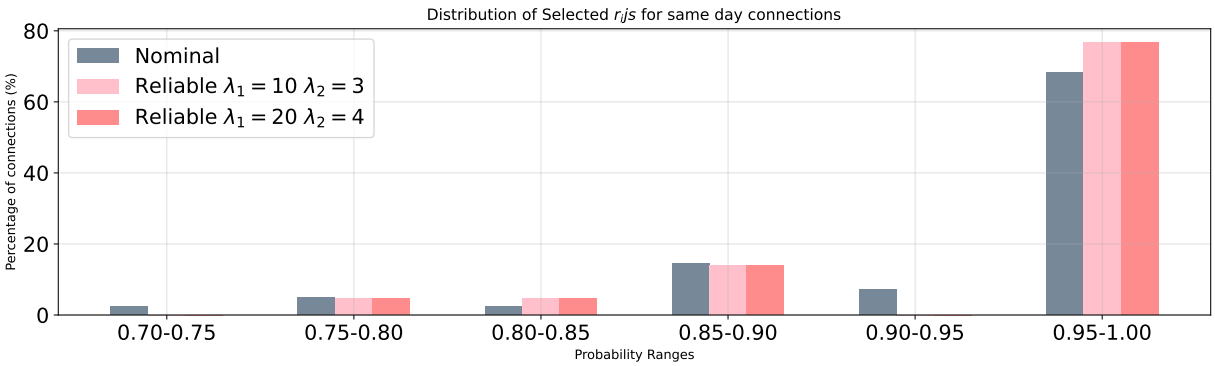


Figure 3.5: Comparison of reliability distribution of arcs selected across solutions

frequency count (230 nodes covered once, 2 nodes twice, 9 nodes thrice, and 2 nodes four times). For the reliable solution $\alpha = 2, \lambda_1 = 10, \lambda_2 = 3$, the number of deadheads is 24 (232 nodes covered once, 7 nodes thrice, 2 nodes four times, and 2 nodes five times). The configuration $\alpha = 2, \lambda_1 = 20, \lambda_2 = 4$ also results in 24 deadheads. The deadheading is lower for reliable solutions as reflected by deadhead costs, Additionally, severity of deadheading into a node is also much lower for reliable solutions.

Analysis of connection choices: Based on the solutions obtained from CG, we compare the reliability distribution of arcs selected for same-day connections (Fig. 3.5). Reliable solutions avoid selecting connections in the lower reliability range (0.70-0.75), unlike the nominal solution. In the highest reliability range (0.95-1.00), reliable solutions select a significantly higher percentage of arcs. This demonstrates that reliable solutions are more effective in prioritizing highly reliable connections, ensuring better overall performance by focusing on arcs with higher reliability.

Sensitivity analysis of α: The analysis of costs across different values of α reveals some trends and insights. For $\alpha = 0.5$, the costs associated with deadheading, both flying and

connection, are significantly lower than for other values of α. This is reflected in the minimal deadhead fly cost (-84.30%) and deadhead connection cost (-85.49%) compared to $\alpha = 1$. Additionally, the total flying cost and total connection cost for $\alpha = 0.5$ are also relatively low, with a slight increase in total connection cost (0.65%), as seen in Table 3.4.

As α increases to 1, the costs rise substantially. For $\alpha = 2$, the deadhead fly cost increases by 77.91%, and the deadhead connection cost rises by 31.62%. The total flying cost also shows a moderate increase (+4.63%), while the total connection cost decreases slightly (-0.58%).

For higher values of α, such as 3, 4, and 5, the deadhead fly costs and connection costs continue to rise significantly, with $\alpha = 3$ showing the highest increase in deadhead fly cost (+87.01%) and deadhead connection cost (+29.41%). The total flying and connection costs exhibit similar patterns of moderate increases and decreases. Overall, the analysis indicates that lower values of α (e.g., 0.5) result in significantly reduced deadheading costs, whereas higher values of α lead to increased costs and operational complexity.

For $\alpha = 0.5$, the flight coverage frequency shows minimal deadheads, with most nodes covered only once or twice, resulting in just 2 deadheads. As α increases to 1 and beyond, the pattern shifts, with more nodes being covered multiple times, leading to a higher number of deadheads. Specifically, $\alpha = 1$ results in 15 deadheads, while $\alpha = 2$ results in 26 deadheads. Notably, for $\alpha = 3$, $\alpha = 4$, and $\alpha = 5$, the number of deadheads stabilizes at 28, indicating that increasing α beyond 3 does not significantly impact the deadheading pattern but rather maintains a consistent level of complexity and overlap in the coverage patterns.

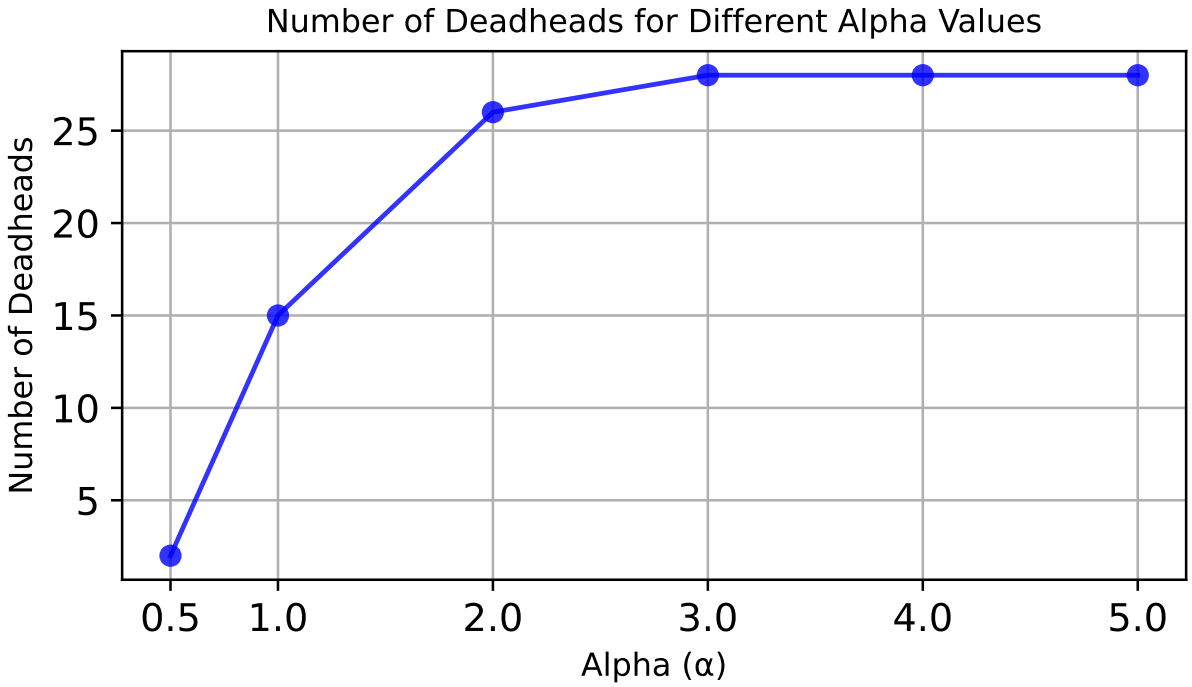


Figure 3.6: Impact of α on Deadheads

Table 3.4: Comparison of Deadheading and Total Costs for solutions with different values of α. Changes in costs (increase or decrease) are in parentheses compared to $\alpha = 1$ and are expressed as percentages. Headers: DFC - Deadhead Flying Cost, DCC - Deadhead Connection Cost, TFC - Total Flying Cost, TCC - Total Connection Cost, TC - Total Cost.

α	**DFC**	**DCC**	**TFC**	**TCC**	**TC**
1	1657	13244	27942	216189	244131
0.5	260 (-84.30)	1922 (-85.49)	26545 (-5.00)	217586 (0.65)	244131 (0.00)
2	2948 (77.91)	17436 (31.62)	29233 (4.63)	214933 (-0.58)	244166 (0.01)
3	3099 (87.01)	17130 (29.41)	29384 (5.18)	214777 (-0.65)	244161 (0.01)
4	3119 (88.17)	17406 (31.48)	29404 (5.23)	214727 (-0.68)	244131 (0.00)
5	3119 (88.17)	16211 (22.38)	29404 (5.23)	214762 (-0.66)	244166 (0.01)

3.6.4 Simulation

Setup: Given a solution, either nominal or reliable ($\lambda_1 = 20, \lambda_2 = 4$), containing a set of optimal crew pairings $\mathcal{P}_{opt}$, where each pairing $p \in \mathcal{P}_{opt}$ contains flights $\mathcal{F}_p$, we simulate by obtaining an actual elapsed time for $i \in \mathcal{F}_p$, actual elapsed time AET_i is given by:

$$AET_i = SET_i + \epsilon_i \tag{3.22}$$

where ϵ_i is a sample from the distribution $\mathcal{KDE}_i^{delay}$, which is generated using Kernel Density Estimation (KDE) on the matched flights' *ArrDelay* without the *LateAircraftDelay*. This approach follows [7], which simulates by approximating the delay to the actual flight arrival time minus scheduled flight arrival time minus late aircraft delay, to avoid recounting the delay. The *LateAircraftDelay* category represents the delays due to propagation through aircraft connections, which will be simulated by the times, and we do not want to add delays again.

The matched flights are identified based on three criteria:

- *Origin:* The airport from which the flight departs.
- *Destination:* The airport to which the flight arrives.
- *Time of Day of the Scheduled Arrival:* The time of day when the flight is scheduled to arrive. This can be segmented into different periods, such as morning (06:00 AM - 11:59 AM), afternoon (12:00 PM - 04:59 PM), evening (05:00 PM - 10:59 PM), and night (11:00 PM - 05:59 AM).

Design of Experiments: We simulate across multiple scenarios by varying two controls: the percentage of irregular operations and the severity of these delays/irregular operations. These variables determine from where the ϵ_i will be sampled for the matched flights. Each flight is simulated across 100 runs. The percentage of irregular operations (% IR) indicates the proportion of runs (realizations of the pairings) that experience irregularities, while the level of delay specifies the severity of these irregularities. For scenarios involving irregular operations (IR), the delay values are sampled from a specified pth using Kernel Density Estimation (KDE). The set of 11 scenarios based on these two parameters is summarized in the Table 3.5

Table 3.5: Simulation Scenarios Based on Percentage Irregularity and Level of Delay. R: Regular Operations, IR: Irregular Operations. For IR runs, delay values are sampled from the specified percentile using KDE.

Scenario Notation	**Description**
100R 0	100% of runs are R
75R, 25IR 70	75% of runs are R, 25% from >70 percentile
75R, 25IR 80	75% of runs are R, 25% from >80 percentile
75R, 25IR 90	75% of runs are R, 25% from >90 percentile
50R, 50IR 70	50% of runs are R, 50% from >70 percentile
50R, 50IR 80	50% of runs are R, 50% from >80 percentile
50R, 50IR 90	50% of runs are R, 50% from >90 percentile
25R, 75IR 70	25% of runs are R, 75% from >70 percentile
25R, 75IR 80	25% of runs are R, 75% from >80 percentile
25R, 75IR 90	25% of runs are R, 75% from >90 percentile
100IR 70	100% sample delay from >70 percentile

Procedure: The crew pairing simulation process calculates actual elapsed times (AET) for each flight by adjusting scheduled times (SET) with sampled delay values. The algorithm checks for infeasible pairings and computes total propagated delays and infeasible

connections (Algorithm 1).

Algorithm 1: Crew Pairing Simulation

for *each crew* $k \in K$ **do**
  Let F_k denote the ordered array of flights covered by crew k;
  for *each flight* $i \in F_k$ **do**
    if $i = 1$ **then**
      $ADT_i = SDT_i$;
    else
      $ADT_i = \max(SDT_i, AAT_{i-1} + \Delta)$;
    end
    $AET_i = SET_i + \epsilon_i$, where $\epsilon_i \sim \mathcal{KDE}_i^{delay}$(matched flights);
    $AAT_i = ADT_i + AET_i$;
    if $i < |F_k|$ **then**
      if $SDT_{i+1} - AAT_i < \Delta$ **then**
        pairing is infeasible;
      else
        pairing is feasible;
      end
    end
    $\text{pgd}_i = \begin{cases} \Delta - (SDT_{i+1} - AAT_i), & \text{if } SDT_{i+1} - AAT_i < \Delta \\ 0, & \text{otherwise} \end{cases}$;
  end
  Calculate Total Propagated Delay: $\sum \text{pgd}_i$;
end

Metrics: We assess the total propagated delays to evaluate our simulation outcomes. Total Propagated Delay (TPGD) measures the total delay carried over from one flight segment to the next. This metric helps in understanding the cascading effect of delays on airline schedules.

The propagated delay for a flight i is defined as:

$$\text{pgd}_i = \begin{cases} \Delta - (SDT_{i+1} - AAT_i), & \text{if } SDT_{i+1} - AAT_i < \Delta \\ 0, & \text{otherwise} \end{cases}$$

where:

- SDT_{i+1} is the Scheduled Departure Time of the next flight.
- AAT_i is the Actual Arrival Time of the current flight.
- Δ is the minimum sit time or buffer allowed between flights.

The Total Propagated Delay (TPGD) over all the flights in the solution is calculated as:

$$\text{TPGD} = \sum \text{pgd}_i$$

Results: We simulate both nominal and reliable solutions for the instances separately for 100 seeds each and analyze the results for both reliable and nominal solution across 11 discussed scenarios. In analyzing the simulations results across various scenarios, reliable crew pairing solutions outperform nominal solutions, particularly in scenarios with higher irregular operations and higher levels of delay. The scenarios with 75% regular operations and 25% irregular operations (75R, 25IR) at various severity levels (70th, 80th, and 90th percentiles) represent situations where delays were greater than pth percentile used to simulate delays. These scenarios are designed to test the robustness of crew pairing solutions

under mixed operational conditions, highlighting how well they handle varying degrees of irregularity.

When comparing reliable and nominal solutions in the 75R, 25IR in Table 3.8 scenarios across different severities of delay, it becomes evident that reliable solutions generally outperform nominal ones, especially at higher percentiles. For the 75R, 25IR-70 scenario, the Total Propagated Delay (TPGD) at the 99th percentile for reliable solutions is 1130.47, while for nominal solutions it is significantly higher at 2928.25.

In the 75R, 25IR-80 scenario, the trend continues with reliable solutions showing a TPGD of 1223.89 at the 99th percentile, compared to 2998.11 for nominal solutions. Even as the severity of delays increases, reliable solutions maintain a lower TPGD, demonstrating their effectiveness in mitigating the impact of severity of delays. In the 75R, 25IR-90 scenario, the TPGD at the 99th percentile for reliable solutions is 1309.16 mins, whereas for nominal solutions it is much higher at 3037.12 mins. Reliable solutions consistently show better performance across these mixed operational scenarios, with a significant reduction in TPGD at higher percentiles compared to nominal solutions. The reliable solutions show an improvement of a minimum of 1000 mins of TPGD for 98th, 99th and 100th percentiles.

3.7 Conclusions and Discussions

In conclusion, this study introduces a novel approach to addressing the Crew Pairing Problem (CPP) under uncertainty by integrating survival analysis into the column generation algorithm. Traditional optimization methods often assume deterministic operations and

Table 3.6: Total Propagated Delays for 75R,25IR Scenarios (70, 80, 90); N: Nominal, R: Reliable.

pth	75R,25IR 70		75R,25IR 80		75R,25IR 90	
	N	R	N	R	N	R
90	358.80	394.40	476.00	515.70	1080.90	831.10
91	382.65	409.07	583.91	680.31	1119.24	922.99
92	461.76	430.40	858.80	734.00	1155.52	939.24
93	484.52	437.03	891.98	745.62	1218.96	1011.49
94	535.16	480.20	905.96	901.20	1526.22	1022.86
95	808.60	734.00	921.75	920.65	1562.25	1100.00
96	896.36	741.96	1011.32	934.96	1639.68	1121.36
97	964.64	938.58	2822.09	986.11	2930.61	1179.86
98	2893.70	1119.20	2922.12	1120.12	3015.38	1241.38
99	2928.25	1130.47	2928.98	1176.07	3037.12	1309.16
100	2953.00	1276.00	3026.00	1282.00	3346.00	1325.00

Table 3.7: Total Propagated Delays for 50R,50IR Scenarios (70, 80, 90); N: Nominal, R: Reliable.

pth	50R,50IR 70		50R,50IR 80		50R,50IR 90	
	N	R	N	R	N	R
90	402.30	424.60	530.00	681.40	1591.70	1217.80
91	406.80	430.45	558.26	745.93	1677.56	1288.81
92	425.64	437.32	595.00	823.28	2030.44	1298.60
93	436.43	480.94	871.91	908.82	2088.97	1317.56
94	483.86	708.88	884.42	935.94	2346.04	1349.06
95	531.40	757.20	891.70	983.40	2454.50	1727.80
96	881.60	822.60	905.64	1011.16	3037.84	1770.44
97	896.27	934.68	923.85	1043.80	3588.64	1979.63
98	944.76	994.74	1054.12	1200.66	3678.24	2203.68
99	2901.00	1283.56	2930.11	1290.31	3886.91	2678.17
100	3693.00	2032.00	3733.00	2113.00	3977.00	2695.00

Table 3.8: Total Propagated Delays for 25R,75IR Scenarios (70, 80, 90); N: Nominal, R: Reliable.

pth	25R,75IR 70		25R,75IR 80		25R,75IR 90	
	N	R	N	R	N	R
90	425.80	679.90	594.80	817.80	1591.70	1217.80
91	440.20	710.32	809.81	834.99	1677.56	1288.81
92	536.04	757.20	822.24	849.96	2030.44	1298.60
93	801.56	795.68	871.91	908.05	2088.97	1317.56
94	813.32	818.60	885.80	927.28	2346.04	1349.06
95	881.20	832.70	919.10	1011.45	2454.50	1727.80
96	898.76	924.68	1025.44	1040.80	3037.84	1770.44
97	1235.57	990.74	1257.88	1084.96	3588.64	1979.63
98	1469.66	1048.38	1469.66	1117.66	3678.24	2203.68
99	2542.62	1125.16	2543.02	1208.14	3886.91	2678.17
100	3693.00	2032.00	3733.00	2113.00	3977.00	2695.00

Table 3.9: Total Propagated Delays for 100R 0 and 100IR 70 Scenarios; N: Nominal, R: Reliable.

pth	100R 0		100IR 70	
	N	R	N	R
90	237.40	253.00	801.30	710.80
91	241.00	266.41	804.45	757.60
92	245.72	312.28	814.76	795.92
93	300.77	328.47	881.28	818.70
94	313.76	351.54	885.66	837.66
95	359.20	423.35	912.65	991.90
96	403.36	734.00	1237.76	1049.76
97	473.35	739.64	1491.35	1116.39
98	937.52	922.22	2894.20	1131.94
99	2534.97	934.86	2960.4	1283.56
100	2928.00	1119.00	3693.00	2032.00

fall short of capturing real-world uncertainties of connections when evaluating pairings. Our approach, SurvCG, leverages survival analysis to predict the reliability of flight connections and integrates these reliability scores into the crew pairing cost function. This augments the cost functions such that the costs reflect how reliable or unreliable a pairing is, along with the planned costs.

The comprehensive reliable crew pairing framework begins with instance generation, then SurvCG, and finally simulating the reliable and nominal solutions. SurvCG effectively reduces both Total Propagated Delay (TPGD) compared to nominal solutions. Reliable solutions also show a significant reduction in higher percentiles across various scenarios, demonstrating their robustness under mixed and highly irregular conditions. For instance, in the 75R, 25IR-70 scenario in Table 3.8, reliable solutions save 1797.78 minutes in TPGD at the 99th percentile. Similar trends are observed in the 75R, 25IR-80 and 75R, 25IR-90 scenarios, with savings of 1955.12 minutes and 1724.70 minutes, respectively.

Future research can explore the use of more priors to determine whether they can improve the prediction accuracy further. Additionally, formulating a cost function that is more realistic in terms of crew cost, rather than just minutes, can enhance the optimization process. One limitation of this approach is that the optimal pairings obtained on solving the CPP using column generation algorithm depends on how accurate the reliability predictions are from the survival model.

3.8 Appendix

3.8.1 C-index and where it fails

$$C = \frac{\sum_{i \neq j} 1\{\hat{\eta}_i < \hat{\eta}_j\} 1\{T_i > T_j\}}{\sum_{i \neq j} 1\{T_i > T_j\}} \tag{3.23}$$

where $\hat{\eta}_i$ and $\hat{\eta}_j$ are the predicted risk scores associated with flights i and j, T_i and T_j are their respective TTEs. For survival models, that do not operate under proportional hazards assumption, we use predicted median or mean TTE instead of risk scores. The higher the C-index, the better the survival model's ability to correctly predict the rank (which will arrive earlier or later) a flight over another based on given characteristics.

3.8.2 Instance Generation in Detail

Algorithm 2: Constructing a network

Input : Flights $\mathcal{F}$, crew base $\rfloor$
Output : Network $\mathcal{N}$
Parameters: Pairing time bound τ

Function `construct_network(`$\mathcal{F}$, $\rfloor$`):`
  Create $\mathcal{A}, \mathcal{C} \in \mathbb{R}^{(|\mathcal{F}_0|+2)\times(|\mathcal{F}_0|+2)}$ to hold arcs (connections) and costs
  for $i = 0$ **to** $|\mathcal{F}|$ **do**
    for $j = 0$ **to** $|\mathcal{F}|$ **do**
      if $(\mathcal{D}_i == \mathcal{O}_j) \wedge (\delta_{min} \leq \mathcal{SDT}_j - \mathcal{SAT}_i \leq \delta_{max})$ **then**
        $\mathcal{A}_{ij} = 1$
        $\mathcal{C}_{ij} = \mathcal{SDT}_j - \mathcal{SAT}_i + \mathcal{SET}_i$
      end
    end
    if $\mathcal{O}_i == \rfloor$ **then**
      $\mathcal{A}_{0i} = 1$
      $\mathcal{C}_{0i} = 0$
    end
    if $\mathcal{D}_i == \rfloor$ **then**
      $\mathcal{A}_{i|\mathcal{F}|} = 1$
      $\mathcal{C}_{i|\mathcal{F}|} = 0$
    end
  end
  $\mathcal{N} = (\mathcal{F}, \mathcal{A}, \mathcal{C})$
  return $\mathcal{N}$

Algorithm 3: Pruning

Input : Network $\mathcal{N}_0$, Crew base $\rfloor$
Output : Pruned flights $\hat{\mathcal{F}}$

Function `prune`($\mathcal{N}_0$, $\rfloor$)**:**
  $\mathcal{F}_0$ = flight nodes in $\mathcal{N}_0$
  $\mathcal{F}_s = \{i \in \mathcal{F}_0 | \mathcal{O}_i == \rfloor\}$
  $\mathcal{F}_t = \{i \in \mathcal{F}_0 | \mathcal{D}_i == \rfloor\}$
  $\hat{\mathcal{F}} = \{\}$, stores flights in pruned network
  while $|\mathcal{F}_s| > 0$ **do**
    $i = \mathcal{F}_s.pop()$
    γ_i = path from $i \rightarrow \mathcal{F}_t$
    if γ_i *exists* **then**
      $\mathcal{F}_0 = \mathcal{F}_0 - \gamma_i$
      $\mathcal{F}_s = \mathcal{F}_s - \gamma_i$
      $\hat{\mathcal{F}} = \hat{\mathcal{F}} \cup \gamma_i$
    end
  end
  $\mathcal{N} = (\hat{\mathcal{F}}, \hat{\mathcal{A}}, \hat{\mathcal{C}})$
  return $\mathcal{N}$

Chapter 4

Conclusions and Future Work

This thesis has introduced and evaluated two innovative approaches aimed at optimizing airline operations by addressing critical challenges in block time prediction and crew pairing under uncertainty. These problems are particularly challenging due to the long-term nature of prediction and planning, where limited knowledge about future conditions necessitates dealing with significant levels of uncertainty.

The first part of the research tackled the challenge of creating realistic schedules by reliably predicting flight segments. To address this, we focused on block time prediction, introducing an embedding-based neural network model. Traditional methods often fail to capture the complex latent relationships between factors affecting block time and struggle with high-cardinality categorical data and temporal variations. Our large-scale study on the 2018 Bureau of Transportation Statistics (BTS) dataset demonstrated significant improvements in forecast accuracy compared to ten baseline models. The model's robust-

ness and reliability were validated through rigorous experimental evaluations and ablation studies.

The second part of the research aimed to reduce costs while operating under uncertainty by building robust schedules for crews. To achieve this, we introduced `SurvCG`, a novel framework that integrates survival analysis into the crew pairing optimization process. Traditional approaches often assume deterministic scenarios or rely on predefined uncertainty sets that do not adequately reflect the true variability and unpredictability inherent in airline operations. `SurvCG` dynamically adjusted crew pairing costs based on reliability scores, resulting in more robust and cost-effective schedules. By incorporating survival probabilities, `SurvCG` provides a dynamic reliability score into the crew pairing cost function, prioritizing crew schedules that are less likely to be disrupted by uncertainties, thus enhancing overall schedule reliability.

Future research can build upon these findings by incorporating airline operations priors. Developing more realistic cost functions that consider a broader range of operational costs, can enhance the optimization process. Additionally, implementing and testing the proposed models in real-time operational environments will be crucial to assess their scalability and practical applicability.

References

[1] Ahmed Abdelghany and Khaled Abdelghany. *Modeling applications in the airline industry*. Routledge, 2016.

[2] Ahmed Abdelghany, Vitaly S. Guzhva, and Khaled Abdelghany. The limitation of machine-learning based models in predicting airline flight block time. *Journal of Air Transport Management*, 107, 3 2023.

[3] Divyam Aggarwal, Dhish Kumar Saxena, Thomas Bäck, and Michael Emmerich. Real-world airline crew pairing optimization: customized genetic algorithm versus column generation method. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 518–531. Springer, 2023.

[4] Mahyar Alimian, Mohammad Saidi-Mehrabad, and Armin Jabbarzadeh. A robust integrated production and preventive maintenance planning model for multi-state systems with uncertain demand and common cause failures. *Journal of Manufacturing Systems*, 50:263–277, 2019.

[5] Hugo Alonso and António Loureiro. Predicting flight departure delay at porto airport:

A preliminary study. In *2015 7th International Joint Conference on Computational Intelligence (IJCCI)*, volume 3, pages 93–98. IEEE, 2015.

[6] Laura Antolini, Patrizia Boracchi, and Elia Biganzoli. A time-dependent discrimination index for survival data. *Statistics in medicine*, 24(24):3927–3944, 2005.

[7] David Antunes, Vikrant Vaze, and António Pais Antunes. A robust pairing model for airline crew scheduling. *Transportation science*, 53(6):1751–1771, 2019.

[8] Ioanna Arkoudi, Rico Krueger, Carlos Lima Azevedo, and Francisco C Pereira. Combining discrete choice models and neural networks through embeddings: Formulation, interpretability and performance. *Transportation Research Part B: Methodological*, 175:102783, 2023.

[9] Ayyuce Aydemir-Karadag, Berna Dengiz, and Ahmet Bolat. Crew pairing optimization based on hybrid approaches. *Computers & Industrial Engineering*, 65(1):87–96, 2013.

[10] Michael Ball, Cynthia Barnhart, Martin Dresner, Mark Hansen, Kevin Neels, AR Odoni, Everett Peterson, Lance Sherry, Antonio Trani, and Bo Zou. Total delay impact study: a comprehensive assessment of the costs and impacts of flight delay in the united states. 2010.

[11] Michael Ball, Cynthia Barnhart, George Nemhauser, and Amedeo Odoni. Air transportation: Irregular operations and control. *Handbooks in operations research and management science*, 14:1–67, 2007.

[12] Jie Bao, Zhao Yang, and Weili Zeng. Graph to sequence learning with attention mechanism for network-wide multi-step-ahead flight delay prediction. *Transportation Research Part C: Emerging Technologies*, 130, 9 2021.

[13] Cynthia Barnhart and Amy Cohn. Airline schedule planning: Accomplishments and opportunities. *Manufacturing & service operations management*, 6(1):3–22, 2004.

[14] Richard Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1 edition, 1957.

[15] Aniket Biswal, F. Gzara, and S. Rambhatla. Reliable crew pairing. 2024.

[16] Aniket Biswal, S. Rambhatla, and F. Gzara. Embedding-based representation learning for forecasting flight. 2024.

[17] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, volume 26, pages 2787–2795, 2013.

[18] Stephane Bratu and C. Barnhart. Flight operations recovery: New approaches considering passenger recovery. *Journal of Scheduling*, 9:279–298, 2006.

[19] Jan K Brueckner, Achim I Czerny, and Alberto A Gaggero. Airline mitigation of propagated delays via schedule buffers: Theory and empirics . 2021.

[20] Jan K. Brueckner, Achim I. Czerny, and Alberto A. Gaggero. Airline schedule buffers and flight delays: A discrete model. *Economics of Transportation*, 26-27:100218, 6 2021.

[21] Bureau of Transportation Statistics. On-time: Reporting carrier on-time performance (1987-present). https://www.transtats.bts.gov/DL_SelectFields.aspx?gnoyr_VQ=FGJ&QO_fu146_anzr=b0-gvzr, 2024. Accessed: 2024-05-20.

[22] Luis Cadarso and Raúl de Celis. Integrated airline planning: Robust update of scheduling and fleet balancing under demand uncertainty. *Transportation Research Part C: Emerging Technologies*, 81:227–245, 2017.

[23] Leonardo Carvalho, Alice Sternberg, Leandro Maia Goncalves, Ana Beatriz Cruz, Jorge A Soares, Diego Brandão, Diego Carvalho, and Eduardo Ogasawara. On the relevance of data science for flight delay research: a systematic review. *Transport Reviews*, 41(4):499–528, 2021.

[24] Patricio Cerda, Gaël Varoquaux, and Balázs Kégl. Similarity encoding for learning with dirty categorical variables. *Machine Learning*, 107(8-10):1477–1494, 2018.

[25] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, June 2002.

[26] Zhenhua Chen and Yuxuan Wang. Impacts of severe weather events on high-speed rail and aviation delays. *Transportation Research Part D: Transport and Environment*, 2019.

[27] Sun Choi, Young Jin Kim, Simon Briceno, and Dimitri Mavris. Prediction of weather-induced airline delays based on machine learning algorithms. In *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pages 1–6. IEEE, 2016.

[28] J. Clausen, A. Larsen, J. Larsen, and Natalia J. Rezanova. Disruption management in the airline industry - concepts, models and methods. *Comput. Oper. Res.*, 37:809–821, 2010.

[29] David R Cox. Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202, 1972.

[30] Steven Coy. A global model for estimating the block time of commercial passenger aircraft. *Journal of Air Transport Management*, 12:300–305, 11 2006.

[31] Vinayak Deshpande and Mazhar Arıkan. The impact of airline flight schedules on flight delays. *Manufacturing & Service Operations Management*, 14(3):423–440, 2012.

[32] Muhammet Deveci and Nihan Çetin Demirel. Evolutionary algorithms for solving the airline crew pairing problem. *Computers & Industrial Engineering*, 115:389–406, 2018.

[33] Muhammet Deveci and Nihan Cetin Demirel. A survey of the literature on airline crew scheduling. *Engineering Applications of Artificial Intelligence*, 74:54–69, 2018.

[34] Yida Ding, Sebastian Wandelt, Guohua Wu, Yifan Xu, and Xiaoqian Sun. Towards efficient airline disruption recovery with reinforcement learning. *Transportation Research Part E: Logistics and Transportation Review*, 179:103295, 2023.

[35] Viktor Dück, Lucian Ionescu, Natalia Kliewer, and Leena Suhl. Increasing stability of crew and aircraft schedules. *Transportation research part C: emerging technologies*, 20(1):47–61, 2012.

[36] Abdelrahman EE Eltoukhy, Felix TS Chan, and Sai Ho Chung. Airline schedule planning: a review and future directions. *Industrial Management & Data Systems*, 117(6):1201–1243, 2017.

[37] Güneş Erdoğan, Mohamed Haouari, Melda Örmeci Matoglu, and Okan Örsan Özener. Solving a large-scale crew pairing problem. *Journal of the Operational Research Society*, 66:1742–1754, 2015.

[38] Terence Ping Ching Fan. Schedule creep–in search of an uncongested baseline block time by examining scheduled flight block times worldwide 1986–2016. *Transportation Research Part A: Policy and Practice*, 121:192–217, 2019.

[39] Federal Aviation Administration. Air traffic by the numbers 2024, 2024. Accessed: 2024-07-01.

[40] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. In *Advances in Neural Information Processing Systems*, volume 26, pages 2121–2129, 2013.

[41] Brandon J. George, S. Seals, and I. Aban. Survival analysis and regression models. *Journal of Nuclear Cardiology*, 21:686–694, 2014.

[42] Karthik Gopalakrishnan and Hamsa Balakrishnan. A comparative analysis of models for predicting delays in air traffic networks. ATM Seminar, 2017.

[43] Ram Gopalan and Kalyan T Talluri. Mathematical models in airline schedule planning: A survey. *Annals of Operations Research*, 76(0):155–185, 1998.

[44] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 855–864. ACM, 2016.

[45] Xuhao Gui, Junfeng Zhang, Zihan Peng, and Chunwei Yang. Data-driven method for the prediction of estimated time of arrival. *Transportation Research Record*, 2675(12):1291–1305, 2021.

[46] Lu Hao and Mark Hansen. Flight time predictability: Concepts, metrics, and impact on scheduled block time. Technical report, 2013.

[47] Lu Hao and Mark Hansen. How airlines set scheduled block times. In *10th USA/Europe Air Traffic Management Research and Development Seminar, Chicago IL*, 2013.

[48] Lu Hao and Mark Hansen. Block time reliability and scheduled block time setting. *Transportation research part B*, 69:98–111, 11 2014.

[49] Frank E Harrell, Robert M Califf, David B Pryor, Kerry L Lee, and Robert A Rosati. Evaluating the yield of medical tests. *Jama*, 247(18):2543–2546, 1982.

[50] Ahmad Tavassoli Hojati, Luis Ferreira, Simon Washington, Phil Charles, and Ameneh Shobeirinejad. Modelling total duration of traffic incidents including incident detection and recovery time. *Accident Analysis & Prevention*, 71:296–305, 2014.

[51] Fei Huang, Dequn Zhou, Jin-Li Hu, and Qunwei Wang. Integrated airline productivity performance evaluation with co2 emissions and flight delays. *Journal of Air Transport Management*, 84:101770, 2020.

[52] J. In and Dong Kyu Lee. Survival analysis: Part i — analysis of time-to-event. *Korean Journal of Anesthesiology*, 71:182 – 191, 2018.

[53] Lucian Ionescu and Natalia Kliewer. Increasing flexibility of airline crew schedules. *Procedia-Social and Behavioral Sciences*, 20:1019–1028, 2011.

[54] Hemant Ishwaran, Udaya B Kogalur, Eugene H Blackstone, and Michael S Lauer. Random survival forests. 2008.

[55] Nabin Kafle and Bo Zou. Modeling flight delay propagation: A new analytical-econometric approach. *Transportation Research Part B: Methodological*, 93:520–542, 2016.

[56] Lei Kang and Mark Hansen. Behavioral analysis of airline scheduled block time adjustment. *Transportation Research Part E: Logistics and Transportation Review*, 103:56–68, 2017.

[57] Edward L Kaplan and Paul Meier. Nonparametric estimation from incomplete observations. *Journal of the American statistical association*, 53(282):457–481, 1958.

[58] Jared L. Katzman, Uri Shaham, Alexander Cloninger, and et al. Deepsurv: personalized treatment recommender system using a cox proportional hazards deep neural network. *BMC Medical Research Methodology*, 18:24, 2018.

[59] Myeonghyeon Kim and Jiheon Bae. Modeling the flight departure delay using survival analysis in south korea. *Journal of Air Transport Management*, 91:101996, 2021.

[60] Young Jin Kim, Sun Choi, Simon Briceno, and Dimitri Mavris. A deep learning approach to flight delay prediction. In *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pages 1–6. IEEE, 2016.

[61] KPMG. Oem and fleet challenges - aviation leaders report 2024, 2024. Accessed: 2024-07-22.

[62] Håvard Kvamme, Ørnulf Borgan, and Ida Scheel. Time-to-event prediction with neural networks and cox regression. *Journal of machine learning research*, 20(129):1–30, 2019.

[63] Miguel Lambelho, Mihaela Mitici, Simon Pickup, and Alan Marsden. Assessing strategic flight schedules at an airport using machine learning-based flight delay and cancellation predictions. *Journal of Air Transport Management*, 82, 1 2020.

[64] Changhee Lee, William Zame, Jinsung Yoon, and Mihaela Van Der Schaar. Deephit: A deep learning approach to survival analysis with competing risks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[65] Xiaobing Li, Jun Liu, Asad Khattak, and Shashi Nambisan. Sequential prediction for large-scale traffic incident duration: Application and comparison of survival models. *Transportation research record*, 2674(1):79–93, 2020.

[66] Da Lu and Fatma Gzara. The robust crew pairing problem: model and solution methodology. *Journal of Global Optimization*, 62:29–54, 2015.

[67] Chandrasekhar S Manchiraju, Milind G Sohoni, and Vinayak Deshpande. It's not simply luck: The impact of network strategy, schedule padding, and operational improvements on domestic on-time performance in the us airline industry, 2022.

[68] Chris Mayer and Todd Sinai. Why do airlines systematically schedule their flights to arrive late. *The Wharton School, University of Pennsylvania*, 2003.

[69] Amirhossein Moosavi and Sadoullah Ebrahimnejad. Scheduling of elective patients considering upstream and downstream units and emergency demand using robust optimization. *Computers & Industrial Engineering*, 120:216–233, 2018.

[70] Doohee Nam and Fred Mannering. An exploratory hazard-based analysis of highway incident duration. *Transportation Research Part A: Policy and Practice*, 34(2):85–102, 2000.

[71] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 809–816, 2011.

[72] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 701–710. ACM, 2014.

[73] Hugo M. Proenca, Ruben Klijn, Thomas Bäck, and Matthijs Van Leeuwen. Identifying flight delay patterns using diverse subgroup discovery. *Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018*, pages 60–67, 1 2019.

[74] Nikolas Pyrgiotis, Kerry M. Malone, and Amedeo Odoni. Modelling delay propagation within an airport network. *Transportation Research Part C: Emerging Technologies*, 27:60–75, 2013.

[75] Frédéric Quesnel, Guy Desaulniers, and François Soumis. A branch-and-price heuristic for the crew pairing problem with language constraints. *European Journal of Operational Research*, 283(3):1040–1054, 2020.

[76] Juan Jose Rebollo and Hamsa Balakrishnan. Characterization and prediction of air traffic delays. *Transportation Research Part C: Emerging Technologies*, 44:231–241, 2014.

[77] A. Schaefer, Ellis L. Johnson, A. Kleywegt, and G. Nemhauser. Airline crew scheduling under uncertainty. *Transp. Sci.*, 39:340–348, 2005.

[78] Milind Sohoni, Yu-Ching Lee, and Diego Klabjan. Robust airline scheduling under block-time uncertainty. *Transportation Science*, 45(4):451–464, 2011.

[79] Alice Sternberg, Jorge Soares, Diego Carvalho, and Eduardo Ogasawara. A review on flight delay prediction. *arXiv preprint arXiv:1703.06118*, 2017.

[80] Yinong Su, Kexin Xie, Hongjian Wang, Zhe Liang, W. Chaovalitwongse, and P. Pardalos. Airline disruption management: A review of models and solution methods. *Engineering*, 2021.

[81] Hajime Uno, Tianxi Cai, Michael J. Pencina, Ralph B. D'Agostino, and L. J. Wei. On the c-statistics for evaluating overall adequacy of risk prediction procedures with censored survival data. *Statistics in Medicine*, 30(10):1105–1117, 2011.

[82] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11), 2008.

[83] Wim JC Verhagen and Lennaert WM De Boer. Predictive maintenance for aircraft components using proportional hazard models. *Journal of Industrial Information Integration*, 12:23–30, 2018.

[84] Wlamir Olivares Loesch Vianna and Takashi Yoneyama. Predictive maintenance optimization for aircraft redundant systems subjected to multiple wear profiles. *IEEE Systems Journal*, 12(2):1170–1181, 2017.

[85] Chunzheng Wang, Minghua Hu, Lei Yang, and Zheng Zhao. Improving the spatial-temporal generalization of flight block time prediction: A development of stacking models. *Journal of Air Transport Management*, 103:102244, 2022.

[86] Shitong Wang and Vikrant Vaze. Modeling probability distributions of primary delays in the national air transportation system. *Transportation Research Record*, 2569(1):42–52, 2016.

[87] Teng Wang, Yufan Zheng, and Haiwen Xu. A review of flight delay prediction methods. *Proceedings - 2022 2nd International Conference on Big Data Engineering and Education, BDEE 2022*, pages 135–141, 2022.

[88] Yanjun Wang, Ying Zhou, Mark Hansen, and Christopher Chin. Scheduled block time setting and on-time performance of us and chinese airlines—a comparative analysis. *Transportation Research Part A: Policy and Practice*, 130:825–843, 2019.

[89] Xin Wen, Hoi-Lam Ma, Sai-Ho Chung, and Waqar Ahmed Khan. Robust airline crew scheduling with flight flying time variability. *Transportation Research Part E: Logistics and Transportation Review*, 144:102132, 2020.

[90] W. Wilhelm. A technical review of column generation in integer programming. *Optimization and Engineering*, 2:159–200, 2001.

[91] Jinn-Tsai Wong and Shy-Chang Tsai. A survival model for flight delay propagation. *Journal of Air Transport Management*, 23:5–11, 2012.

[92] Cheng-Lung Wu. Improving airline network robustness and operational reliability by sequential optimisation algorithms. *Networks and Spatial Economics*, 6:235–251, 2006.

[93] Cheng-Lung Wu. *Airline operations and delay management: Insights from airline economics, networks and strategic schedule planning*. Routledge, 2016.

[94] Yifan Xu, Sebastian Wandelt, and Xiaoqian Sun. Airline scheduling optimization: literature review and a discussion of modelling methodologies. *Intelligent Transportation Infrastructure*, 3:liad026, 2024.

[95] Joyce W Yen and John R Birge. A stochastic programming approach to the airline crew scheduling problem. *Transportation Science*, 40(1):3–14, 2006.

[96] Bin Yu, Zhen Guo, Sobhan Asian, Huaizhu Wang, and Gang Chen. Flight delay prediction for commercial air transport: A deep learning approach. *Transportation Research Part E: Logistics and Transportation Review*, 125:203–221, 5 2019.

[97] Gang Yu and Jian Yang. Optimization applications in the airline industry. *Handbook of Combinatorial Optimization: Volume1–3*, pages 1381–1472, 1998.

[98] Bahadır Zeren and Ibrahim Özkol. A novel column generation strategy for large scale airline crew pairing problems. *Expert Systems with Applications*, 55:133–144, 2016.