A Generalized Approach to Aircraft Trajectory Prediction via Supervised Deep Learning

Nathan Schimpf*, Eric J. Knoblock †, Zhe Wang *, Rafael D. Apaza †, Hongxiang Li *
* Dept. of Electrical and Computer Engineering
University of Louisville, Louisville, USA
Email: nathan.schimpf@louisville.edu, zhe.wang@louisville.edu, h.li@louisville.edu
† Communications and Intelligent Systems Division
NASA Glenn Research Center, Cleveland, USA
Email: eric.j.knoblock@nasa.gov, rafael.d.apaza@nasa.gov

Abstract—**As research advances diverse forms and missions of aircraft, the National Airspace System (NAS) will become increasingly crowded, limiting current communications resources to accommodate aviation operations. Ongoing research proposes a paradigm of airspace communications, such that resources are autonomously and dynamically allocated via intelligent agents; this allocation requires accurate representations of the NAS, including the predicted positions of aircraft. State-of-the-art research emphasizes the importance of a hybrid-recurrent framework for trajectory prediction and compares the impact of commonly considered weather data on prediction accuracy. However, current research has been limited in its scope of efforts, frequently utilizing a unique flight route, architecture, set of weather data, and date range. This article considers the challenges of generalizing hybrid-recurrent predictive models for flight trajectories. Results illustrate an increase in error variance when identical models are trained over a generalized set of flights; this may be mitigated with careful tuning of hyperparameters, both in the network structure and optimization algorithms. Even so, an irreducible vertical error was identified, resulting from the complex takeoff and landing procedures which can not be correlated to functions of weather or additional assumptions of aircraft behavior. Finally, the use of a test route indicates that generalized models still do not possess sufficient knowledge for general aircraft predictions, with mean error increases ranging from 70-500%. These results illustrate the need for continued efforts on improving model versatility, as well as potential limitations for spectrum allocation near airports and other centers.**

I. Introduction

Over the next decade, the number of aircraft and the diversity of missions handled by the National Airspace System (NAS) are expected to grow rapidly. In large part, this will be driven by emerging aircraft and airspace missions, such as those related to Unmanned Aircraft Systems (UAS) and Urban Air Mobility (UAM). It is anticipated that the operation of these systems will integrate with existing aviation infrastructure. From a communications perspective, this poses a significant challenge: current spectrum available to aviation communications is limited and relies on static channel allocations, predominantly for analogue voice communication.

As a result, NASA is investigating techniques for the autonomous allocation of aviation spectrum, such that NAS communications will assure quality-of-service and spectral efficiency with an increasing number of aircraft. Current approaches consider the use of artificial intelligence, specifically deep-learning solutions, which are intrinsically data-driven [1]. Toward this end, a key data product for spectrum allocation is an accurate trajectory prediction. An accurate estimation of aircraft position enables the development of key data products for allocating resource usage. From the predicted trajectory, direct inferences can be made toward localizing the aircraft to a particular sector (and associated frequency), as well as estimating the path loss between the aircraft and ground stations. By careful analysis of predicted trajectory, communication events might be inferred, as changes in flight route and altitude all require coordination between the pilot and air traffic control.

This paper revisits the challenge of 4D trajectory prediction (latitude, longitude, altitude, and time) for commercial aircraft, with experiments critically examining the selected data and architectural designs in potential deep-learning approaches. The initial work of this paper was presented in [2], which has since been expanded upon. The contributions of this complete set of research includes:

- A comparison of the efficacy of weather products used in-literature, and combinations thereof, for trajectory prediction via a hybrid-recurrent neural network. These results indicated the presence of bias when training over a single flight route, while affirming Echo Top as the best-suited holistic weather product for trajectory prediction[1].
- A comparison of current deep-learning mechanisms which are intuitively suited to the challenges of trajectory prediction. Specifically, the use of gated recurrent units (GRUs), independently recurrent neural networks (IndRNNs), and self-attention (SA) were considered. These results indicated the potential for improved accuracy using SA layers, while IndRNN may not be suited for the complexity of decision making in this task [1].
- The generalization of flight data previously used: a 5-route training dataset (approximately 1,086 flights) is proposed, with a 6th test route (approximately 578 flights) over a 2-week period (January 10th - 14th, 2019). Training with this dataset improved mean accuracy of predictive models at the cost of an increase in error

variance, particularly in horizontal error.

- The fine-tuning of model hyperparameters for prior architectures which showed promise. This aimed to improve generalization and consistent training of models; in particular, larger recurrent hidden sizes and convolutional/attention hidden depths were necessary for retaining flight and weather features, respectively.
- An unsuccessful attempt to modify flight data for improved accuracy. This is included to illustrate the challenges facing accurate altitude predictions as they relate to the takeoff/landing procedures, where aircraft behaviors are adjusted to accommodate heavy congestion, rather than resulting from limitations in climb-rate modelling or changes due to weather.
- The evaluation of tuned architectures over a test flight route, which increases mean error by 70-500%, illustrating the need for generalizable architectures and a thorough investigation of transfer learning for trajectory prediction.

A. *Article Outline*

The remainder of this paper is: Section II presents a survey of relevant background information, including prior research on trajectory prediction and a brief discussion of current deep-learning mechanisms which may be suited to this research problem. Section III describes the general formulation and deep learning framework employed in this paper, as well as preliminary studies of weather products and deep learning mechanisms based on a limited number of flights collected over a single route. Section IV introduces a complete training dataset of generalized flight routes, and the challenges encountered when training and evaluating models. Section V attempts to mitigate the increased error via model tuning and adjustments to existing flight information; the section concludes by evaluating these adjusted approaches over a test route excluded from training data. Finally, research conclusions and further steps are discussed in Section VI

II. State of the Art

This section surveys relevant research to the trajectory prediction challenge. Four papers are briefly discussed, which significantly contributed to the form of trajectory prediction considered by this paper. Following this, deep learning mechanisms of interest for this task are presented.

A. *On Trajectory Prediction*

In [3], the authors sought to develop a model that could correlate historic flight data with surrounding weather data. A Hidden Markov Model was developed, where flight coordinates were associated with those of the NOAA Rapid Refresh (RAP) dataset. Using the defined trajectory as observed emissions and regions of RAP coordinates as hidden states, the HMM was trained. 594 4D trajectories were collected for one flight (DAL2173) with identical arrival and departure locations (ATL to MIA). Ayhan and Samet were the first to formulate a complete trajectory prediction model and proposed the notion of surrounding weather data (feature cubes), a concept which has widely defined later model developments. The efficacy of this approach has yet to be matched, however it is unclear if the reported accuracy is a result of the learning approach or the heavy constraints placed on flight data collection.

A more complex, deep generative network is presented in [4]. At its core, this framework generates Gaussian Mixture Models using a sequence-to-sequence paradigm for Long Short-Term Memory (LSTM). The predictions of these models are then filtered using a variety of techniques (Adaptive Kalman Filter, Beam Search, Rauch-Tung-Striebel Smoother). 3D Flight plans and 4D flight trajectories were recorded for 1,679 flights with identical arrival and departure airports (IAH to BOS). Weather data were collected from the NOAA North American Mesoscale (NAM) database, specifically Westerly and Southerly (U/V) Winds, Air Temperature, and Convective Weather. While the results were not as compelling as those in [3], Liu and Hansen present several significant concepts, including a sequence-to-sequence paradigm and efficient methods of organizing and accessing weather data. Several reasons for these poorer results may be inferred: foremost, the NAM database is limited in resolution, as each data point is interspaced at 12 km and refreshed every 6 hours; the selected flight is infrequent, and collecting the number of flights in this paper may have required a significant range of seasons and consequent weather patterns; finally, the model itself may have been unnecessarily complicated by relying on the repeated generation and sampling of Gaussian Mixture Models.

Taking inspiration from the previous paper, the Pang, Hao, Hu, and Liu presented a convolutional-LSTM hybrid network to predict aircraft trajectories [5]. This model presents a basis for hybrid-recurrent networks: weather features are extracted and represented through a series of convolutional and dense layers, while supplemented with the aircraft location prior to the provided cube. The combination of abstracted weather data and prior aircraft position are fed into an LSTM layer to predict the aircraft's position. LSTM layers were selected for recurrence to mitigate the vanishing gradient challenge of training traditional Recurrent Neural Networks (RNNs). Feature cubes were generated from Echo Top measurements, and flight data was collected for a total of 2,528 flights of identical arrival and departure points (JFK to LAX) over the dates November 1st, 2018 through February 5th, 2019. Initial research focused on 3D trajectory predictions (ignoring altitude), and reported efficacy in terms of improved error (described by Euclidean Norms) over that of the flight plan. Pang *et al.* reported the accuracy of 47% of all flight plans were improved by their predictive model, on average by 12.3%. While this efficacy appears satistying, this relative metric is not directly comparable to other research; the two papers prior reported efficacy in terms of a horizontal and vertical error (units of nautical miles and ft), with no reference to the error of related flight plans. As a result, this paper should be more critically contextualized in other research.

Finally, Ma and Tian discuss the prediction of aircraft based solely on prior Automatic Dependent Surveillance-Broadcast

[1]Contributions were initially presented in [2].

(ADS-B) data [6]. The research considers single and multi-point forecasting of 4D trajectory using sequences of prior 4D data, as well as ground speed and heading information. Three models are presented for this task, one convolutional (CNN), one recurrent (LSTM), and one CNN-LSTM hybrid. The results reinforce the usefulness and importance of prior design choices in hybrid-recurrent networks, while also offering some qualitative understandings and intuition. Data were collected for approximately 397,000 flights from Qingdao to Beijing, providing the largest dataset of all considered in existing research. However, again, the metrics reported in this paper are not directly comparable to those described in prior research; those presented here are standard error metrics within deep learning (Mean Absolute Percentage Error, Mean-Squared Error, etc.), but not relevant to describing the usefulness of a trajectory prediction model.

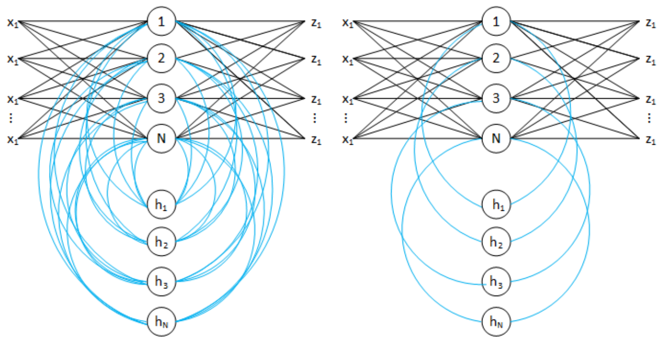
Over the course of the existing body-of-research, several individual advances have been made; a core formulation and approach illustrated the viability of machine learning for this task in [3]; An initial deep learning framework was developed in [4]; A more robust deep learning approach was formulated in [7] and [5], while the architecture was validated with [6], which provided a starting point for this research. However, a clear synthesis of each contribution cannot be directly made. Each contribution considered different datasets, both in their selection of weather products and in collecting along singular, distinct flight routes. Additionally, contributions often demonstrated the efficacy of their approach with different metrics. To advance the state of aircraft trajectory prediction, two major contributions are noted in this research: 1) the generalization of the predictive task, experimenting with a variety of datasets, flights, and models to contextualize other research, and 2) a search for an accurate, generalizable deep learning model that suits this task.

B. Model Architectures

Beyond those mentioned in prior research, three components have the potential to improve current hybrid-recurrent designs. This section presents these components and a modular hybrid-recurrent framework.

Similar in concept to LSTM, Gated Recurrent Units (GRUs) mitigate the issue of vanishing gradients with the addition of gating mechanisms to regulate input and retained information. However, their implementation accomplishes this task with fewer parameters and hidden states than LSTM, theoretically allowing for faster training and improved performance. The two cells perform similarly on a variety of tasks, often without a clearly optimal choice [8].

As another technique for recurrence, Independently Recurrent Neural Networks (IndRNNs) resolve the vanishing gradient problem without the addition of gating mechanisms, resulting in a network with fewer parameters and potentially greater memory [9]. In traditional recurrent neural networks (and gated designs such as LSTM and GRU), the hidden states associated with a recurrent layer are connected to each cell in the layer; in contrast, IndRNN proposes a restriction on the connections, such that each cell in a layer is connected to only one hidden state. This approach prevents hidden states from being harshly penalized and neglected, solving the vanishing gradient problem so long as a sufficiently small optimizer learning rate is selected.



(a) Traditional RNN (b) IndRNN

Fig. 1. Comparative Visualization RNN layers

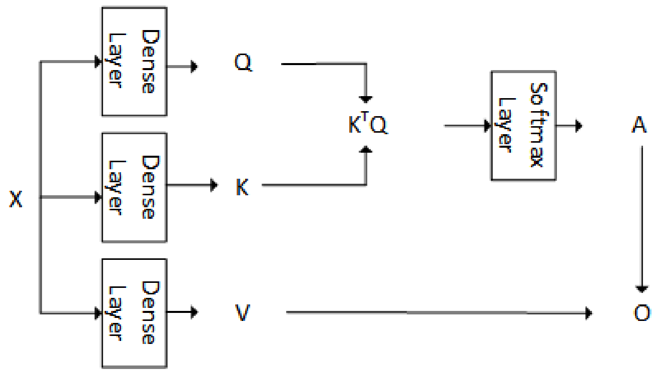


Fig. 2. Functional Diagram of Self-Attention Layer

While attention mechanisms have existed for years in the field of natural language processing, their application to other fields (such as computer vision) has only been considered recently [10]. For our discussion, soft self-attention (SA) is considered as in Figure 2 and equations 1 through 5 [11]. A complete data sequence is received and transformed into key, query, and value datasets. In the discussed attention, a sense of locality is embedded in the data via softmax layer, whose input is a matrix multiplication of transformed key and query datasets. Findings indicate that self-attention is a capable supplement following convolutional layers when extracting patterns from 2D data [10].

$$Q = \sigma(\frac{W_Q^T X}{\sqrt{d_{out}}}) \tag{1}$$

$$K = \sigma(W_K^T X) \tag{2}$$

$$V = \sigma(W_V^T X) \tag{3}$$

$$A = softmax(K^T Q) \tag{4}$$

$$O = AV \tag{5}$$

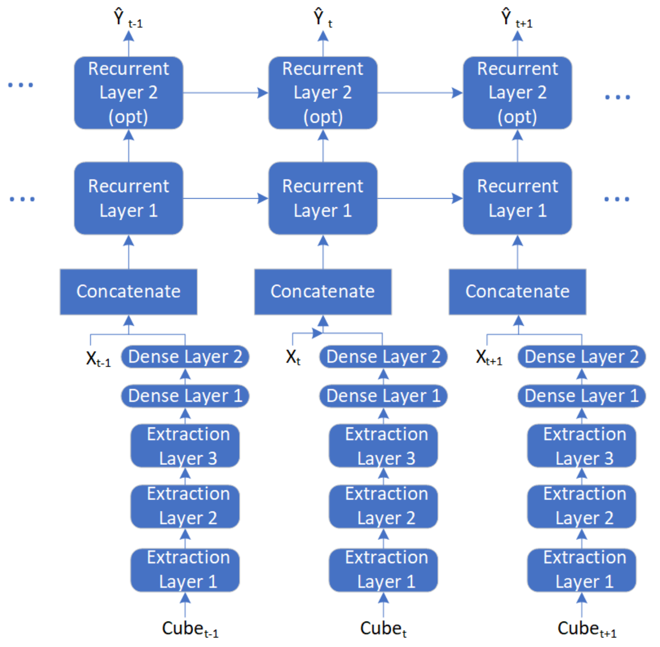


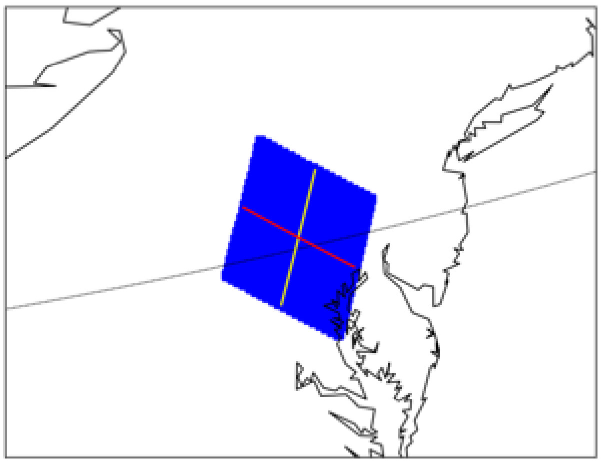
Fig. 3. Framework for Hybrid-Recurrent Network

III. System Model

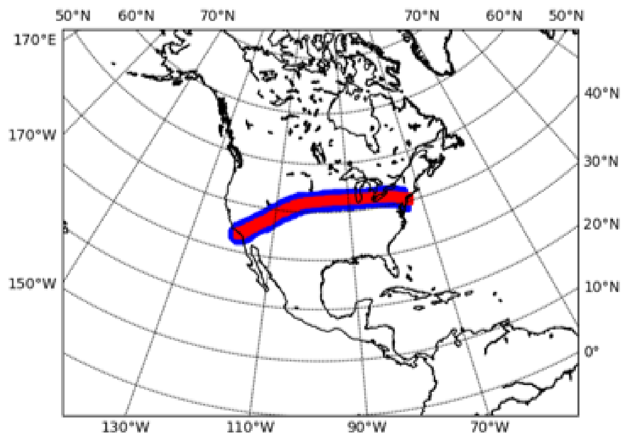
For this research, trajectory prediction is treated as a sequence-to-sequence translation problem: a complete sequence of flight plan information and weather data following that flight plan (feature cubes) are provided as inputs to the models, which predict the complete flight trajectory. The architecture of prediction abstracts from the hybrid network of convolutional and Long Short-Term Memory (LSTM) layers to a general hybrid-recurrent architecture, unrolled in Figure 3. Feature cubes are abstracted through a sequence of extraction layers (convolution or self-attention), as well as two dense layers. The resulting abstraction is fed alongside the anticipated position from the flight plan into a recurrent network (LSTM, GRU, or IndRNN). The output of the recurrent network is treated as an estimate ($\hat{Y}$) of the actual trajectory (Y), which can be used in estimating loss and updating model parameters.

The preprocessing of feature cubes, flight plan, and flight trajectory information all follow the algorithms developed in [5], with two additions: weather data which may vary by altitude is also collected one level above and below the current altitude (generating a 20x20x3 cube for each flight plan point). Additionally, all data have been linearly interpolated at a one-minute interval; this restricts the data to a useful format for recurrent neural networks, where entries are regularly-dispersed and the total sequence length is within the experimental memory constraints of LSTM/GRU [9].

This framework for trajectory prediction is intentionally broad to allow for both its data inputs and model components to be assessed for their fitness to the general challenge. The following subsections analyze both aspects of the framework over a limited dataset prior to the main contributions of this work. Flight and weather data were limited to a single route,



(a) Case I



(b) Case II

Fig. 4. Visualization of collecting a single feature cube (top) and set of feature cubes (bottom, blue) along a filed flight plan (bottom, red)

KLAX-KJFK, over the two-week period of January 10th-24th, 2019; this resulted in a total of 379 available, valid flights. For all experiments, the setup consisted of

- A workstation using Ryzen 1950X processor and dual Nvidia RTX2080 graphics cards, with Ubuntu 20.04 LTS serving as its operating system.
- Data preprocessing and deep learning models as python projects in [12] and [13].
- Flight and Echo Top data collection are accomplished via access to NASA's Sherlock Data Warehouse, which maintains a repository of data for air traffic management [14].
- Additional weather data are collected directly in from NOAA High Resolution Rapid Refresh (HRRR) cloud portals or in coordination with Massachusetts Institute of Technology Lincoln Labs Corridor via their Corridor Integrated Weather Services (CIWS) [15].

A. Data Items

The general framework emphasizes the use of feature cubes, but not what features might be captured by the cubes. This section discusses five potential weather features based on their use in-literature, then presents a brief assessment of their effectiveness for supervised deep learning problems.

Table I summarizes the weather data sources and popular products that have been previously utilized in literature. Each source provides gridded data at a regular update interval, offering sufficient weather information to support flight prediction. In order to alleviate potential forecasting error, all weather data is collected based on their current (actual) measurement. Data sources include Corridor Integrated Weather Service (CIWS), a set of radar-based weather features that are tailored to air traffic management tasks; NOAA High Resolution Rapid Refresh (HRRR) and Rapid Refresh (RAP), hourly measurements of 14 high-impact weather features; and NOAA North American Mesoscale (NAM), a general database of over 60 measurements taken to track changes in climate. Because of their maximized spatial and temporal resolution for respective weather products, CIWS and HRRR are used in analyzing relevant weather products.

TABLE I
SUMMARY OF WEATHER DATASETS

Weather Database	Used in	Relevant Variables	Update Period	Resolution
Corridor Integrated Weather Service (CIWS)	[5]	Vertically Integrated Liquid (VIL) Echo Top	Current 2.5 Min Forecast 5 Min	1.85 km (1 nmi)
North American Mesoscale (NAM)	[4]	Humidity Temperature Wind Speed (U) Wind Speed (V)	6 Hours	12 km (6.48 nmi)
Rapid Refresh (RAP) High Resolution Rapid Refresh (HRRR)	[3]	Humidity Temperature Wind Speed (U) Wind Speed (V)	1 Hour	RAP 13 km (7.01 nmi) HRRR 3 km (1.61 nmi)

Assessing the effectiveness of weather products and combinations thereof is conducted in two parts: a correlation analysis of products of interest, and supervised training of an identical neural network using varied weather data. Candidate weather data includes Echo Top and Vertically Integrated Liquid (VIL) products generated by CIWS, as well as atmospheric temperature and westerly (U) and southerly (V) wind components from NOAA HRRR. To reduce the number of combinations considered, a limited cross-correlation of weather products is first performed; the results of this cross-correlation determined which combination of weather products supplement one another sufficiently to justify the computational cost of providing additional data to the models. From here, the individual weather products and selected combinations thereof will be used to train a hybrid-recurrent model of fully convolutional layers and a single LSTM layer.

Figure 5 presents the results of the limited cross-correlation. Limitations stemmed from two factors: a difference in coverage area and fidelity between the two weather sources (CIWS and HRRR), as well as the computational complexity of a complete cross-correlation (shifting over the complete 2D space of the data). As a result, the figure represents only a cropped-and-interpolated portion of the continental United States coverage, with only a direct overlap of the two regions considered. These results should therefore be treated as indicative, but not wholly representative of the relationships between products.

After correlating the selected products, Echo Top and VIL data were each found to be weakly correlated to other products; this is likely due to the sparsity of Echo Top and VIL data at any given time. By contrast, Temperature and Wind Components were all moderately correlated with one another; as a result, combinations should predominantly consider only one of the three HRRR products. Therefore, the model training will consider nine combinations of datasets: Echo Top, VIL, Temperature, U Wind, and V Wind Component will be considered individually; Combinations will also consider Echo Top and VIL, Echo Top and Temperature, and VIL and Temperature. Finally, Temperature and V Wind Component will be tested in combination, to verify the assumption of performance based on their higher correlation.

After training the products and product combinations above, prediction results are summarized in Table II, with percent improvement comparisons drawn against Echo Top, which is treated as the default due its use in [5]. Several trends can be observed: Without considering combinations of multiple products, Echo Top provides the lowest horizontal error; this is expected, due to the nature of the measurement being tailored to air traffic management, alongside its sparsity. VIL performed notably worse than most products, despite its sparsity and correlation to convective weather. This reflects how VIL represents only the presence of liquid, not whether it is indicative of humidity, precipitation, type of precipitation, etc. - not all of which require a flight reroute. While no NOAA data product could provide an improved horizontal accuracy, the use of any of the three provided degrees of improvement to vertical accuracy predictions - likely because of data being altitude-varying. Notably, V Wind Components provided the greatest improvement in vertical accuracy; it is believed this reflects a skew in flight data, where a notable percent of arrivals and departures occurred with a significant north/south component (parallel to V Wind), while the en-route flight had a majority east/west component (parallel to U Wind).

No combinations of products provided sufficient improvements in accuracy to justify additional data processing and model complexity. However, challenges with preprocessing, specifically effective data normalization, may have inhibited the usefulness of some products. In particular, prior experiments without normalizing temperature data yielded horizontal accuracies much closer to those of Echo Top, making temperature a viable choice for predictive modelling. For the remainder of this paper, Echo Top data will be uses in the preprocessing and generation of feature cubes for model inputs.

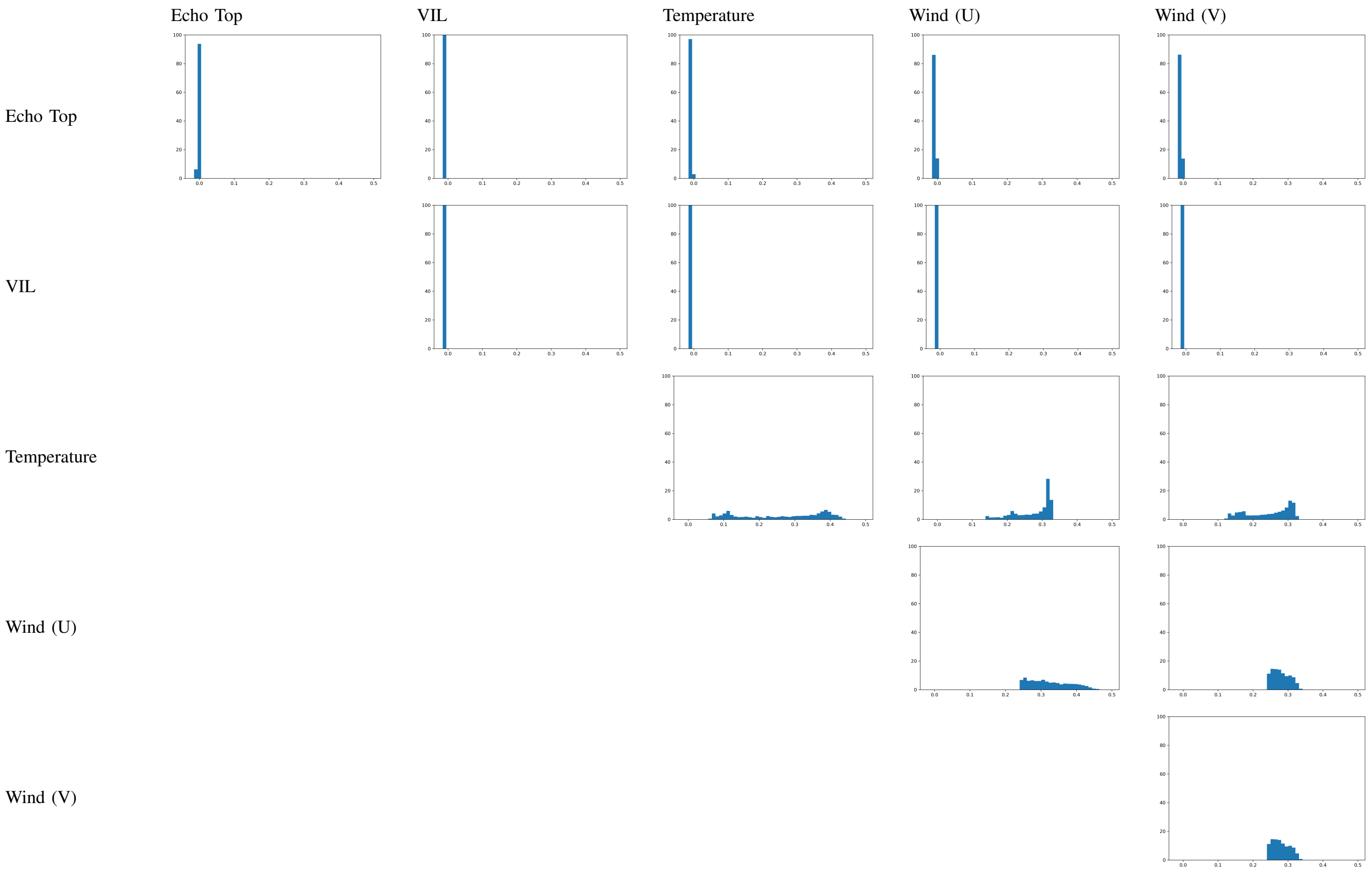


Fig. 5. Histograms of Cross-Correlation Coefficients Ranging (0, .5)

TABLE II
SUMMARY OF WEATHER PRODUCT PERFORMANCES

Product(s)	Horizontal Error (μ/σ in nmi)	Vertical Error (μ/σ in ft)	Improvement over Echo Top ($\mu_{Horiz}/\sigma_{Horiz}$ as percent)	Improvement over Echo Top (μ_{Vert}/σ_{Vert} as percent)
Echo Top	**50.017** 48.854	**1160.07** 1420.26	**0** 0	**0** 0
VIL	**55.171** 67.276	**1230.23** 1514.95	**-10.304** -37.708	**-6.048** -6.667
TMP	**52.983** 60.901	**1130.72** 1399.41	**-5.931** -24.659	**2.530** 1.468
U Wind (E/W)	**50.560** 54.588	**1128.17** 1420.57	**-1.085** -11.738	**2.749** -0.022
V Wind (N/S)	**50.167** 51.376	**1097.16** 1390.80	**-0.29** -5.164	**5.422** 2.074
ET + VIL	**50.670** 57.596	**1118.72** 1365.45	**-1.305** -17.895	**3.564** 3.859
ET + TMP	**50.194** 51.937	**1156.50** 1424.41	**-0.354** -6.312	**0.307** -0.292
VIL + TMP	**52.520** 65.513	**1248.81** 1558.70	**-5.005** -34.101	**-7.650** -9.748
TMP + V Wind	**49.578** 51.764	**1128.25** 1430.29	**0.877** -5.957	**2.743** -0.707

B. Deep Learning Mechanisms

The framework also proposes broader terms for the components of the deep learning architecture, allowing the consideration of model components excluded from prior research and discussed in Section II. This section considers a brief assessment of these deep learning mechanisms by training models varied components with comparable hyperparameters over identical data. This section only considers a direct implementation of related deep-learning mechanisms with assumed hyperparameters; a more in-depth exploration of hyperparameters and overfitting is presented in the main contributions of this work (Section V-A).

Extending the general form of Figure 3, models were

TABLE III
DEFAULT MODEL PARAMETERS

Parameter Description	Parameter Value
Convolution Kernel Sizes	[6x6, 3x3, 1x1]
Convolution Stride Lenths	[2, 2, 1]
Convolution Filter Sizes	[1, 2, 4]
Attention Output Dimensions	[128, 36, 36]
Dense Layer Sizes	LSTM, GRU: [16, 3] IndRNN: [16, 97]
Recurrent Input Size	6
Recurrent Depth	GRU, LSTM: 1 or 2 Layers IndRNN: 2 or 3 Layers
Optimizer Learning Rate	$2 * 10^{-4}$

defined by the selection of weather feature extraction mechanism, recurrence mechanism, and number of recurrence layers. These parameters were assumed and modified from the network layout in [5], and are summarized in Table III. Extraction mechanisms are defined in a depth of three layers, and include a purely convolutional design, a purely self-attention design, and a convolutional design with self-attention serving as a final layer. Recurrence techniques included LSTM, GRU, and IndRNN layers, always with a hidden state size of 100. Finally, the depth of recurrence was limited to 1 or 2 layers. For IndRNN cells, an additional layer was included to allow for information sharing between neurons. All models were trained on Echo Top feature cubes and flight data. All models were initially trained with 379 flights from Los Angeles to New York, collected over the two week period of January 10th to 24th, 2019.

From the above setup, few conclusive results were drawn. Results are considered inconclusive due to prior tests yielding large variances in error. Table IV is presented to indicate preliminary results and a general format for comparing effectiveness of architecture design, where horizontal and vertical error are standard metrics for general flight trajectory prediction error, and percent improvements are prescribed in comparison to the flight plan and predictions of a CNN-LSTM design of 1 layer (a model inspired from [5] and used as a baseline for this task). The table does not detail all possible model combinations, with the intent that those presented would sufficiently describe the trends in usefulness of different recurrent and extraction components.

Despite the aforementioned limitations, one trend still tends to appear, which is the usefulness of self-attention in this task. The use of self-attention, while potentially incrementing performance as a supplement to convolutional layers, notably improved performance of models when behaving as an outright replacement for convolutional layers. The improvements via fully attentional extraction may be caused by the globality of self-attention: since self-attention layers consider all features in relation to each other (as opposed to a fixed number of neighbors in one sequence element), attention mechanisms may be able to define complex filters that consider the interactions of data between each feature cube in the provided sequence. Consequently, the use of self-attention as a supplement to convolutional networks may be inhibited by convolutional extraction techniques: since convolutional layers can only extract features within each cube individually, the patterns extracted may be counterproductive from a more global perspective.

An additional point should be made, that IndRNN performed notably poorly. When comparing results of IndRNN-based models, error was notably worse than that of the flight plan, and moreso in comparison to the baseline model. This may be a reflection of implementation challenges; the selected hyperparameters may have been limiting for the model. However, due to implementation challenges, this mechanism will be abandoned in remaining sections; despite being the least computationally-complex of the recurrent mechanisms tests, they required significant training time as a GPU-accelerated variant of the mechanism was not supported for the selected version of PyTorch at the time of the experiments.

IV. GENERALIZING THE ROUTES OF TRAJECTORY PREDICTION

A notable gap from research, particularly highlighted when comparing weather products, is the potential bias of model training toward a specific route. To-date, trajectory prediction research has focused on individual routes, split into a training and validation dataset. This has the potential for concern over several reasons

- Certain weather information may correlate strongly with predicting particular routes or regions of the airspace, causing the model to fit toward patterns that only partially predict flight behaviors.
- As a corollary, certain airspace regions and seasons may not significantly affect flight routes, preventing models from learning reroute behaviors at larger scales. This may particularly be the case for flights avoiding coastal regions of the United States or flights of shorter duration (less than 2 hours).
- Model architectures may rely on learning features of a route, rather than general features and changes in flight behavior.
- Current architectures may therefore run into memory constraints resulting in large errors as a result of dealing with unknown routes.

To address this limitation, a generalized set of training and validation data was created. Five routes were selected for training and validation, as summarized in Figure 6 and Table V. Routes were selected to vary the general heading, duration, and coverage of flights within the continental United States. Where possible, routes were also selected to match with those used in prior research.

Directly extending the work in [2], 3 models were selected with the same sets of hyperparameters (CNN-LSTM, CNN-GRU, and SA-LSTM, each with a single recurrent layer) and trained over 500 epochs. Training was conducted with a 4-fold cross-validation to reduce the potential bias in data and variance from random initialization. After model training was completed, the evaluation of models was considered in two aspects: the performance of the models on their generalized validation sets, and the performance of the models specifically on validation flights from New York to Los Angeles (approximating comparisons to [2]).

TABLE IV
SUMMARY OF SELECT MODELS' PERFORMANCES

Model	Horizontal Error (μ/σ in nmi)	Vertical Error (μ/σ in ft)	Improvement over Flight Plan (μ_{Horiz}/μ_{Vert} as percent)	Improvement over CNN-LSTM1lay (μ_{Horiz}/μ_{Vert} as percent)
CNN-LSTM1lay	**63.5584**	**1160.27**	**39.5920**	**0**
	26.8905	1500.83	64.0127	0
CNN-LSTM2lay	**60.9995**	**1167.39**	**42.0241**	**4.0260**
	29.2265	1551.46	63.7919	-0.6135
CNN-GRU1lay	**59.8954**	**1120.04**	**43.0735**	**5.7632**
	28.0559	1399.75	65.2606	3.4676
CNN-GRU2lay	**47.2278**	**1156.16**	**55.1131**	**25.6938**
	22.9868	1332.40	64.1404	0.3548
CNN-IndRNN2lay	**119.1314**	**1219.99**	**-13.2263**	**-87.4361**
	63.1298	1682.68	62.1607	-5.1463
CNN-IndRNN3lay	**122.6245**	**1219.86**	**-16.5463**	**-92.9320**
	61.8804	1682.68	62.1645	-5.1355
CNN+SA-LSTM1lay	**59.3252**	**1178.57**	**43.6154**	**6.6603**
	29.5847	1546.38	63.4454	-1.5763
CNN+SA-LSTM2lay	**60.1143**	**1152.56**	**42.8654**	**5.4187**
	28.4656	1537.15	64.2520	0.6651
SA-LSTM1lay	**40.9453**	**804.73**	**61.0843**	**35.5785**
	23.7972	1054.89	75.0405	30.6436
SA-LSTM2lay	**56.2102**	**990.82**	**46.5760**	**11.5613**
	25.1934	1294.62	69.2687	14.6051

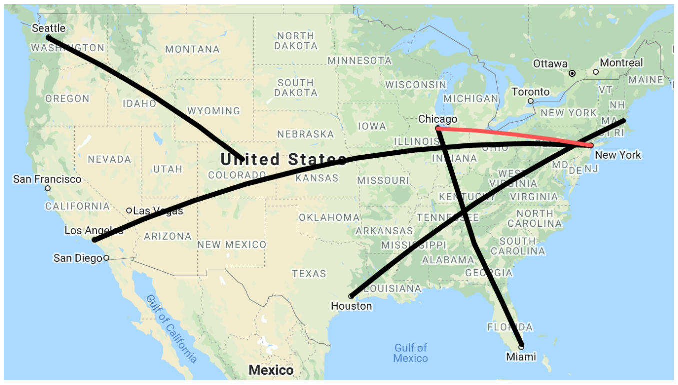


Fig. 6. Collection of all training/validation (black) and test (red) flight routes

TABLE V
DESCRIPTION OF TRAINING/VALIDATION (BLACK) AND TEST (RED) FLIGHT ROUTES

Route	Heading	Nonstop Time	Used in	Flights (100 / 14 days)
KJFK-KLAX	WSW	5.5 Hrs	[7], [5]	2,963 / 494
KIAH-KBOS	NE	3.75 Hrs	[4]	283 / 39
KATL-KORD	NNW	1.75 Hrs		1,892 / 310
KSEA-KDEN	SE	2.5 Hrs		1,463 / 237
KATL-KMCO	SSE	1.75 Hrs	[3]	2,258 / 388
KORD-KLGA	E	2.0 Hrs		3,247 / 577
Total (Train)				**8,859 / 1,468**
Grand Total				**12,106 / 2,045**

A. Results

The results in Table VI summarize the trajectorywise horizontal and vertical error of three models discussed in Section III-B, as well as the error of evaluating their generalized equivalents over the total validation set and subset of validation flights following the same original route.

In general, training identical models over a generalized set of flight routes improved the average accuracy while increasing the variance in error - this can especially be seen with the horizontal errors, and becomes more pronounced when considering the performance of generalized models on the original flight route. This is likely to indicate the current limitations in pattern learning and prediction related to the networks, and may be an indicator of necessary hyperparameter tuning.

All models trained over the generalized dataset predicted notably worse altitudes than initial models - however, altitude predictions follow the same prior divergence in error mean and variance when considering the subset of initial flights. The increase in error over the generalized set is most likely related to the varied durations of flights and subsequent challenges in predicting altitude: recall that the original flight (KJFK-KLAX) is also the longest of all flights in the dataset, and therefore likely to spend the largest percent of its duration at a constant cruising altitude. Consequently, the models evaluated over the generalized dataset are more heavily penalized for inaccurate predictions of takeoff and landing altitude behaviors. These limitations might be absolved with hyperparameter tuning (improving accuracy for less-frequent altitude changes during the en-route portion of a flight), but require closer investigation to understand.

Despite being the most apparently-promising model discussed in [2], the final model (combining self-attention and long short-term memory) encounters the most difficulty with model generalization. While it is possible that attention mechanisms are not well-suited to the task of trajectory prediction, it is believed that the assumed layer sizes and dimensionality assigned to attention mechanisms in this task may be at-fault. As a result, it is both possible and likely that the SA-LSTM model lacked the necessary complexity to generalize over a larger set of data, much moreso than prior models based on a small body of prior research.

While suffering from a larger percentage increase in error variance, CNN-GRU remained the most consistent predictive

TABLE VI
SUMMARY OF DATA GENERALIZATION MODEL PERFORMANCE

Case	Horizontal Error in [2]	Vertical Error in [2]	Case Horizontal Error	Case Vertical Error	Horiz. Error Improvement	Vertical Error Improvement
	(μ/σ in nmi)	(μ/σ in ft)	(μ/σ in nmi)	(μ/σ in ft)	(μ/σ as percent)	(μ/σ as percent)
CNN-LSTM Total Data	63.5584 26.8905	1160.27 1500.83	48.5837 63.1849	1525.08 2121.23	23.561 -134.971	-31.442 -41.337
CNN-LSTM KJFK-KLAX	63.5584 26.8905	1160.27 1500.83	29.5634 67.8611	711.24 1673.58	53.486 -152.361	38.700 -11.510
CNN-GRU Total Data	59.8954 28.0559	1120.04 1399.75	44.5988 40.8868	1486.57 2032.41	25.539 -45.733	-32.725 -45.198
CNN-GRU KJFK-KLAX	59.8954 28.0559	1120.04 1399.75	26.3954 46.1633	686.36 1597.22	55.931 -64.541	38.720 -14.107
SA-LSTM Total Data	40.9453 23.7972	804.73 1054.89	53.1642 67.7009	1565.15 2183.16	-29.842 -184.49	-94.493 -106.956
SA-LSTM KJFK-KLAX	40.9453 23.7972	804.73 1054.89	33.4991 72.7729	732.47 1708.79	18.186 -205.804	8.980 -61.987

model. This may be related to fewer internal states associated with each cell - forcing the model to learn meaningful internal representations. This earliest indication will be revisited throughout model evaluations.

V. MITIGATING GENERALIZATION VARIANCES

Noting the trends in increased error variances relating to the larger set of flight routes, this section presents two actions taken to improve model performance over a more generalized dataset. First, model tuning is conducted to address potential overfitting and expand the set of features retained by the model. Noting the fundamental the irreducible limits of vertical error over the course of tuning, adjustments to the flight plan are attempted. Finally, these adjusted models and data are used to evaluate the effectiveness of of deep learning approaches on an unused test route from O'Hare International Airport to LaGuardia International Airport.

A. Model Hyperparameter Tuning

Based on increases in error variance, the prior model architectures (CNN-LSTM, CNN-GRU, SA-LSTM) are more closely developed towards generalization, along with a fourth (SA-GRU) included for completeness. These hybrid-recurrent designs were progressively tuned for a number of parameters, approximately visualized in Figure 7. Hyperparameters are broken out into four groups: those specifically related to potential overfitting, those relating to the complexity of the weather feature extraction, those relating to the complexity of recurrent memory and feature learning, and those relating to control of the optimizer. As each group of parameters was appropriately tuned, model parameters were incorporated into the subsequent experiments.

Several measures were taken to insure the repeatability of each experiment (and therefore significance of related results).

- To minimize the observation of improvements related to randomness in model training, each model is initialized with constant weights and biases of 0.5.
- All random seeds are initialized to a global constant.
- Algorithms within PyTorch are forced to use deterministic implementations. This is particularly important for convolutions, where nondeterministic implementations are used by default to improve computing speeds.

Additionally, some parameter selections remained constant for all tuning experiments. For all experiments leading up to optimizer selection and tuning, RMSProp was used with a learning rate of $1 * 10^{-4}$. To prevent potential training errors related to zero-padding, a batch size of 1 was used for all experiments as well.

1) Overfitting Parameters: Initially, three hyperparameters related to overfitting were adjusted, including the dropout rate p, weight regularization L2 penalty (referred to as weight decay in PyTorch libraries), and use of batch normalization. Each combination of parameters was tested in isolation with a CNN-LSTM architecture following parameters in Table III, training over 200 epochs. While the use of batch normalization between each layer could be discretely tested, regularization penalties and dropout rates required a specified set of rates to test: specifically, dropout rates included 0%, 0.01%, 0.1%, 1%, 5%, 10%, and 20%; regularization penalties included 0, $1 * 10^{-8}, 1 * 10^{-6}, 1 * 10^{-5}, 1 * 10^{-4}, 1 * 10^{-3}, 1 * 10^{-2}$, and $1 * 10^{-1}$. It should be noted that dropout layers were selectively incorporated into the model, specifically before the final extraction layer, first recurrent layer, and after each subsequent recurrent layer, as illustrated in Figure 7. Results from this portion of testing indicated the importance of the minimal incorporation of dropout and regularization (with selections of 0.01% and $1 * 10^{-8}$), while batch normalization wholly impaired model training.

2) Extraction Sizes: Weather extraction hyperparameters strictly considered the size of feature retention in the hidden (k_0) and output (k_1) extraction layers, which were tested on both CNN-LSTM and SA-LSTM models with recurrent input sizes of both 6 and 10. It is assumed these parameters meaningfully transfer to GRU recurrent layers. While this size parameter can be directly treated as the number of filters in convolutional layers, attention layers treat this parameter analogously. Instead, the size represents a multiple of the total output features of the layer - an attempt to retain an equivalent number of features in each model. For each model, 200 sample models were constructed with layer sizes randomly selected between 1 and 32; each sample was trained over 50 epochs

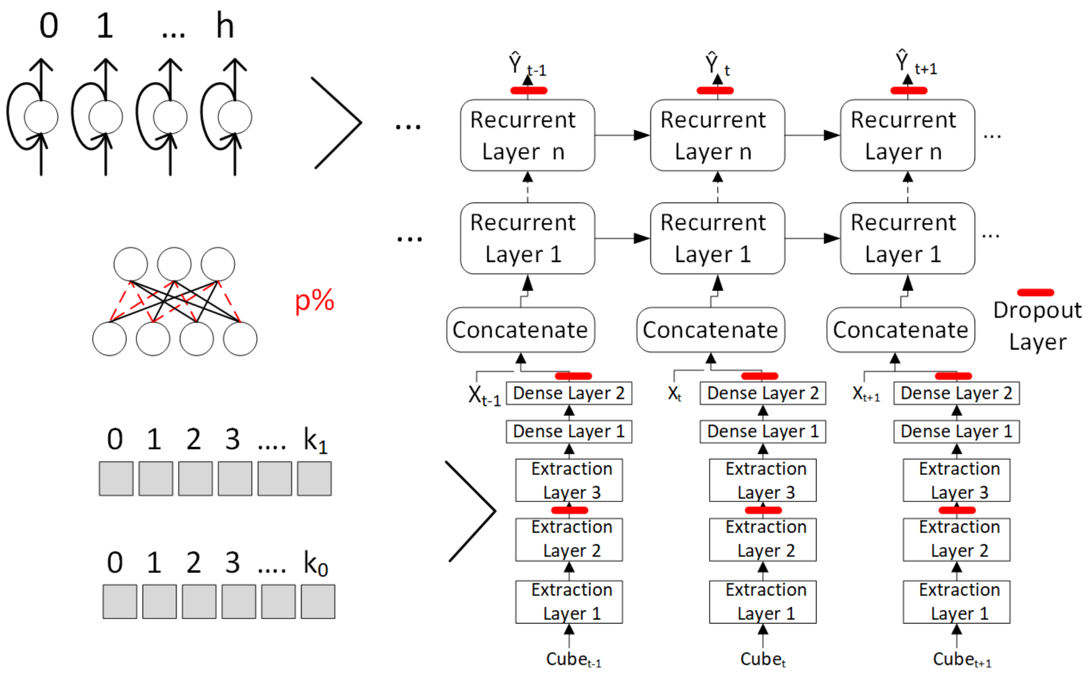


Fig. 7. Approximate visualization of architecture hyperparameters

before all samples were compared for common trends. Based on this test, convolutional models performed best with larger hidden and output sizes, while attention models benefitted from large hidden sizes but significantly-reduced output sizes; all models performed better with the smaller recurrent input size of 6. Convolution sizes of (28, 22) and attention sizes of (31, 8) were selected.

3) Recurrent Depth and Size: Expansion of the recurrent portion of the model came under consideration to increase the number of features and potential of compression / retention of flight behaviors and corresponding changes in trajectory. Recurrent hyperparameters considered both a total recurrent depth (n) and hidden size of each layer (h). For this experiment, each set of hyperparameters must be tested individually for CNN-LSTM, CNN-GRU, SA-LSTM, and SA-GRU models, as each model is likely to have learned different classes of weather features and each recurrent type require different parameters to retain said features. For completeness, a grid search was conducted: recurrent depths of up to 4 layers were considered, while hidden sizes varied up to 1000 cells in increments of 50. Because recurrent networks require the largest amount of hardware resources (and therefore the longest training time), the experiment was limited to 20 epochs. Based on the results of this experiment, both attention models were found to require an additional recurrent depth (2) for optimal performance, with a notably larger hidden size (600). Neither convolution model significantly benefited from the added recurrent depth (and consequent computational cost), though still notably improved from increased hidden size (LSTM: 1000, GRU: 650).

4) Optimizer Selection and Decay: Finally, selection of an appropriate optimization approach was considered. Optimizer choice played a particularly important role due to concerns of generalization; research so far has relied heavily on adaptive optimizers, which have been found to poorly generalize models for some tasks [16]. While this concern had not appeared in validation tests up to this point, there was still interest in considering a wider array of optimization algorithms prior to finalizing model tuning. This subsection is broken into two experiments: the first of which considers selection of an optimization algorithm, the second of which considers the scheduling of a limited set of algorithms.

For the first experiment, a total of 8 optimizer options were initially considered for CNN-LSTM models, assuming transferrability. These optimizers included Stochastic Gradient Descent (SGD), SGD with momentum, SGD with Nesterov momentum updating, Adam, Adam with default parameters (no weight regularization), Adadelta, Adagrad, and RMSProp. With exception to each SGD variant, each optimizer utilized its default learning rate in the PyTorch library. for each SGD variant, the optimal learning rate and momentum (if applicable) were determined by grid search in a similar process as the prior subsection. These parameters are summarized in Table VII. For each optimizer, a CNN-LSTM model was trained over 250 epochs with 5 random initializations.

The results of this initial attempt are illustrated in Figure 10, with the average of each model's training sessions plotted. While it is clear that SGD variants will not achieve the same degree of accuracy with enough brevity to be sufficient for this problem, several challenges came with final training of models, particularly insuring the convergence of GRU models. This lead to experiments with decaying the learning rates of select optimizers.

For the second experiment, optimizer scheduling considered

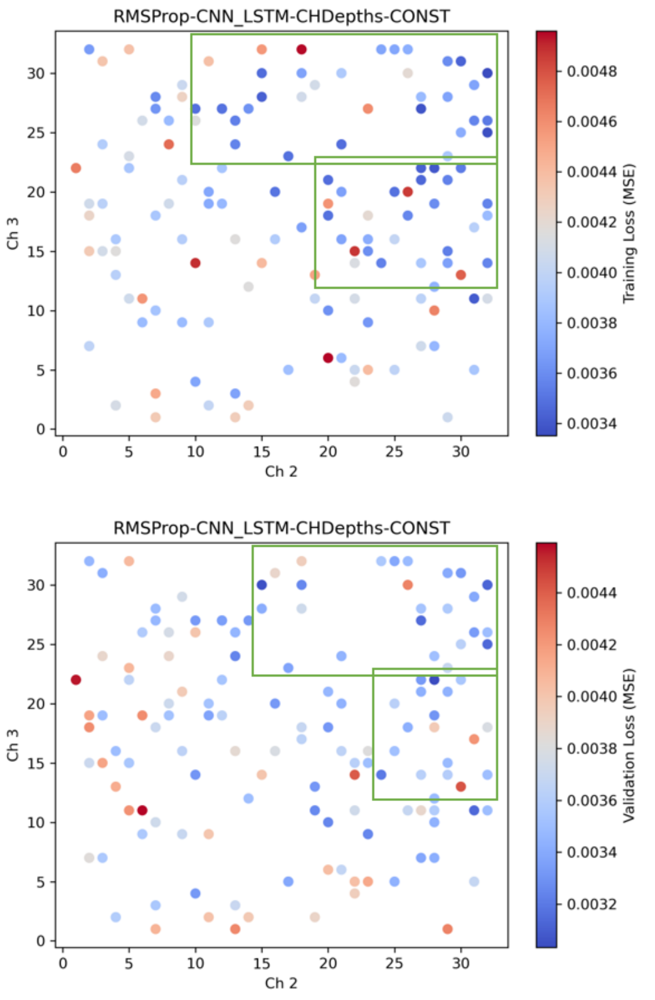


Fig. 8. Scatterplot of extraction channel depths for training (top) and validation (bottom) datasets

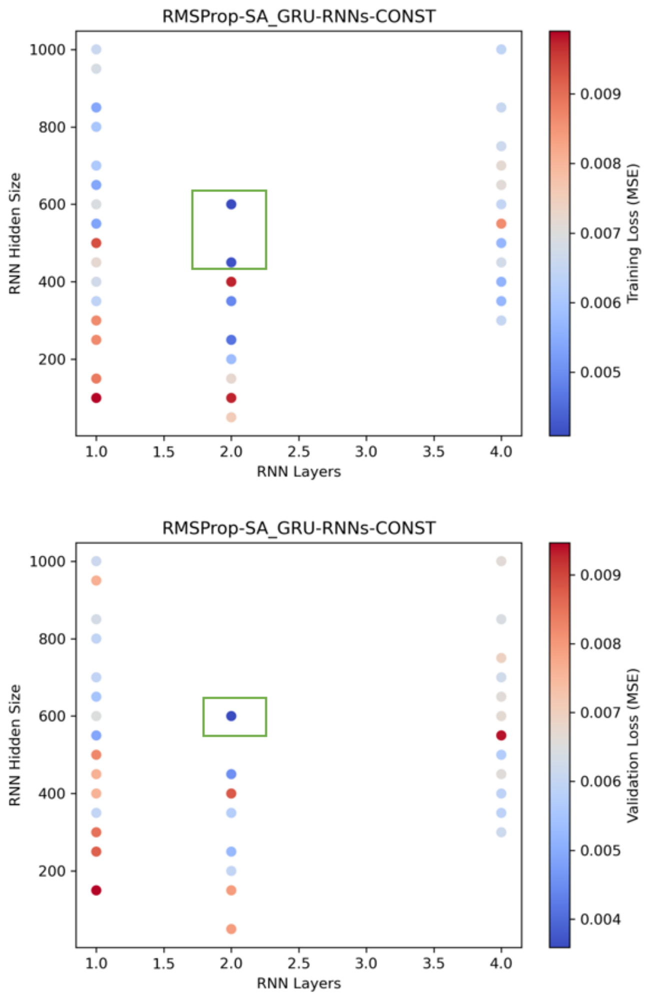


Fig. 9. Scatterplot of GRU depth and size for training (top) and validation (bottom) datasets

TABLE VII
DEFAULT PARAMETERS OF TESTED OPTIMIZERS

Optimizer	Learning Rate	Weight Regularization	Momentum
SGD	0.01	$1 * 10^{-6}$	0.0
SGD+Momentum	0.01	$1 * 10^{-6}$	0.5
SGD+Nesterov	0.01	$1 * 10^{-6}$	0.5
Adam	0.001	$1 * 10^{-6}$	N/A
Adam with Initial Parameters	0.001	0.0	N/A
Adadelta	1.0	$1 * 10^{-6}$	N/A
Adagrad	0.01	$1 * 10^{-6}$	N/A
RMSProp	0.0001	$1 * 10^{-6}$	N/A

a simple learning rate decay. Three decay rates (0.1, 0.5, 0.9) and step sizes (10, 30, 50 epochs) are considered, while each optimizer (SGD, Adam, RMSProp) is initialized with a learning rate of 0.01. Each model is initialized randomly 3 times, and the average final training and validation losses were compared.

5) Final Model Tuning and Results: Based on the set of above experiments the complete hyperparameters for each model are listed in Table VIII. With these tuned models, each was again trained over the 5-flight dataset over 300 epochs and a 4-fold cross-validation. The average of each model's validation scores are listed in Table IX. Additionally, the number of failed folds for each tuned model (how many of the 4 cross-validation folds failed to converge and therefore were excluded from the results) is provided.

Overall, the results illustrate an improvement in consistency - across all models, horizontal errors improved and achieved similar final results; albeit weakly, this holds for vertical error as well. As a consequence, the previously-noted advantage of CNN-GRU seems to be minimized. For horizontal error the improvements are most pronounced, with largest gains in horizontal variance - a particular concern point from data generalization. While vertical error did improve on accuracy and consistency, its achievements are still below what might be desired for useful for its application in estimating path loss and sector localization.

It should be noted that learning rate decay improved some, but not all, challenges with convergence. While CNN-GRU models consistently converged to a meaningful value (an

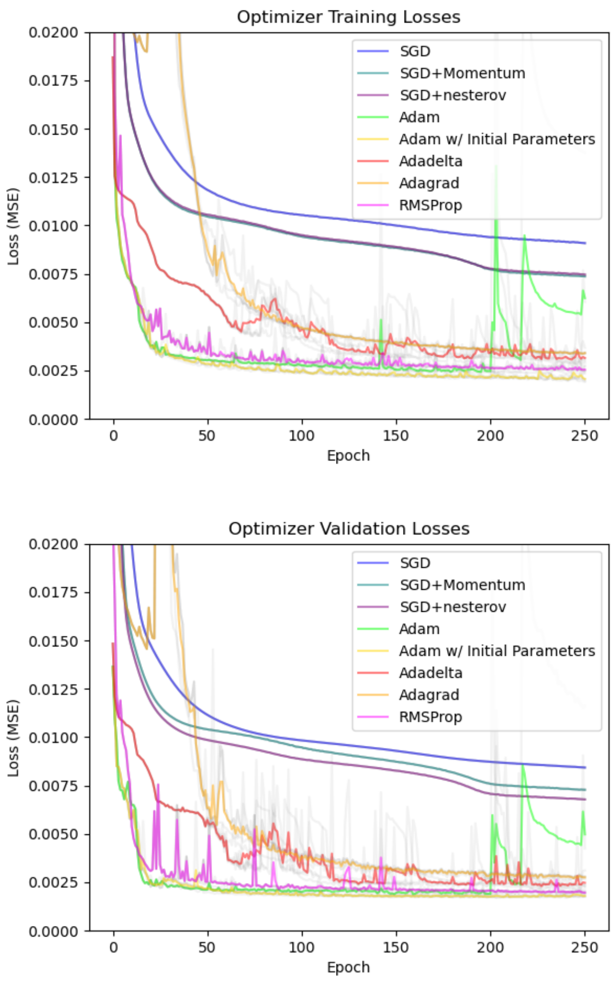


Fig. 10. Training (top) and Validation (bottom) Losses of Optimizers with Default Parameters for Tuned CNN-LSTM Model

improvement over prior observation), SA-LSTM and SA-GRU did not. The single failed fold of SA-LSTM may be acceptable, however SA-GRU obviously struggles to converge; as a result, SA-GRU will not be further discussed or included in future evaluation.

While the use of self-attention seemed initially promising, once all models are tuned there appears to be no clear benefit of self-attention over state-of-the-art convolution. In reality, comparisons between tuned SA- and CNN- models illustrate SA- models lagging in vertical accuracy. An obvious outlier to accuracy is the error variance of SA-GRU, which is intentionally neglected due to the limited number of useful cross-validation folds. That being said, no model outperformed the SA-LSTM initially discussed in Section III-B; however, this may be a reflection of the potential of models when considering only a single flight route.

Comparing these results to those presented in other state-of-the-art research illustrates some possible improvements. While comparisons may be drawn analogously using the results in Section III-B, no direct comparisons are drawn between the Pang *et al.*'s results in [7] or [5]. Additionally, no comparisons are drawn to [6]. For all papers, this is a reflection of differences in performance metrics, where [7] and [5] discuss results as a relative improvement over flight plan Euclidean norms, and [6] discusses results in terms of final error of several deep learning metrics (Mean-Squared Error, Mean Absolute Percentage Error, etc).

TABLE VIII
SELECTED HYPERPARAMETERS FOR EACH MODEL

Parameter	Value (CNN-LSTM)	Value (SA-LSTM)	Value (CNN-GRU)	Value (SA-GRU)
Extraction Hidden Depth	28	31	28	31
Extraction Output Depth	22	8	22	8
RNN Input	6	6	6	6
RNN Hidden	1000	600	650	600
RNN Depth	1	2	1	2
Optimizer	RMSProp	Adam	RMSProp	RMSProp
Learning Rate	.01	.01	.01	.01
Decay Rate	0.5	0.5	0.5	0.5
Decay Step Size	10	30	10	30
Dropout	0.01%	0.01%	0.01%	0.01%
Weight Decay	$1 * 10^{-6}$	$1 * 10^{-6}$	$1 * 10^{-6}$	$1 * 10^{-6}$
Batch Norm	None	None	None	None

No model is able to achieve close to the horizontal accuracy presented in [3]; however, the vertical error, particularly standard deviation in altitude, is improved over significantly. Accounting for data selection, both accuracy trends may be expected: flight data was collected for a specific aircraft (DAL2173) over a one flight route (ATL to MIA), one of the shortest routes considered in the generalized dataset. This short duration, combined with the likelihood of the airline having preferred internal flight plans for each route (in terms of cost savings, trip time, etc), may contribute to a consistent intended route for all flights, providing the greatest degree of predictability in model training and evaluation. Inversely, the large vertical error deviation is expected for shorter flight routes due to the lack of predictability in takeoff and landing phases of flight.

Comparing to the results in [4], all models appear to have significantly improved both horizontal and vertical accuracy - on average, by 17.54% and 54.52%, respectively. While significant, these improvements should still be contextualized with the selected route. Collected flight data considered the second-longest and most infrequent route in the generalized datset, flying from Houston to Boston and accounting for 6.76% of all flights in the 2 weeks of data. Collection of 1679 flights would likely require the complete year of flight data discussed by the authors, incorporating the seasonal changes in weather (and consequent changes in flight patterns and reroutes) within a limited dataset. It is therefore expected that improvements over horizontal error are accomplished by the

TABLE IX
COMPARISON OF INITIAL AND TUNED MODELS OVER GENERALIZED TRAINING/VALIDATION DATA

Model	Initial Failed Folds	Initial Horizontal Error (μ/σ in nmi.)	Initial Vertical Error (μ/σ in ft.)	Tuned Failed Folds	Tuned Horizontal Error (μ/σ in nmi.)	Tuned Vertical Error (μ/σ in ft.)	Horizontal Error Improvement (μ/σ in %)	Vertical Error Improvement (μ/σ in %)
CNN-LSTM	0	48.5837 63.1849	1525.08 2121.23	0	40.3659 33.7023	1208.13 1637.33	16.915 46.661	20.783 22.812
SA-LSTM	0	53.1642 67.7009	1565.15 2183.16	1	40.2251 33.5524	1258.36 1676.77	24.338 50.440	19.601 23.195
CNN-GRU	0	44.5988 40.8868	1486.57 2032.41	0	40.8215 34.8290	1163.07 1554.76	8.470 14.816	21.762 23.502
SA-GRU	0	46.6859 44.1792	1646.01 2232.96	3	42.3441 29.9462	1575.54 1782.26	9.300 32.217	4.281 20.184

results of data generalization and model tuning; improvements in vertical error, however, are noteworthy, as longer routes tend to minimize this error as a result of a larger portion of the flight occurring at the filed cruising altitude.

B. Data Input Sufficiency (adjusting flight plans)

One of the primary concerns noted in trajectory prediction efforts so-far has been the increase in vertical error as a result of data generalization. While model tuning improved altitude predictions, there seems to be a certain irreducible increase in error when considering flights of shorter duration, tied to the larger emphasis on ascent and descent portions of the flight. An initial attempt to address this is discussed below, where the altitudes of the flight plan are more closely represented by assigning climb rates along observed trends in data.

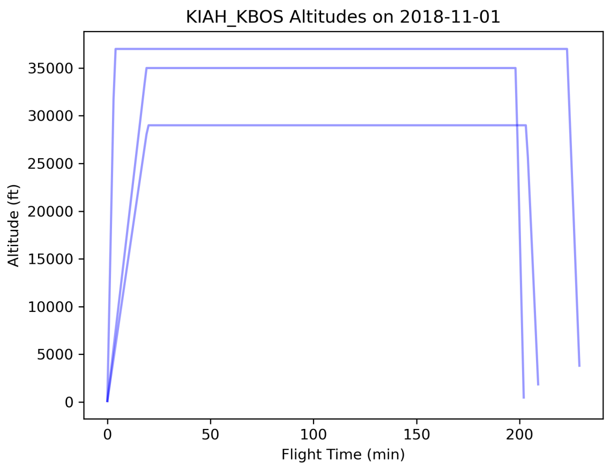
Initial flight plans were developed solely based on a linear interpolation using available cruising altitude, navigation aids, and waypoints from Sherlock Data Warehouse. While this provided a sufficient approximation of horizontal position (to collect an appropriate set of feature cubes), the altitude generation remained a concern - climb rates would often be too rapid and unrealistic for aircraft behavior, as seen in Figure 11. By comparison, most flights would approach a cruising altitude gradually over time (20-30 minutes) - to illustrate this point, a collection of flight trajectory altitudes and their approximate climb rates are provided in Figure 12.

Based on these observations, a simple model was built to assign appropriate climb and descent rates; Figure 13 provides an overview. In this approach, general behaviors are assigned for ascent and descent phases each: in ascent, a climb rate of 3000 ft / min is initially achieved, before decaying to a computed climb rate C and finally to 0 (reaching the cruising altitude); in descent, a constant descent rate of 2000 ft / min is reached and held, before decaying to a rate of 0 ft / min over 10 minutes (with constant rate m of 200 ft/min/min). The climb and descent phases are simplified to Equations 6 through 9, then sampled to fit the 1 min interval of the flight plan data.

$$X = -25t_0^2 + 3000t_0 + 7750 \tag{6}$$

$$C = 3000 - 50t_0 \tag{7}$$

$$t_f = C/500 \tag{8}$$



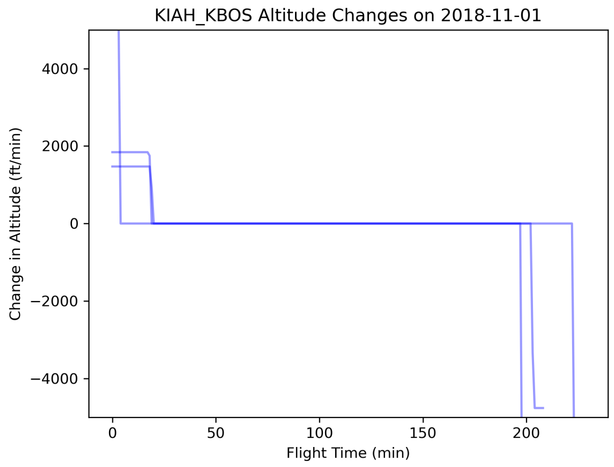
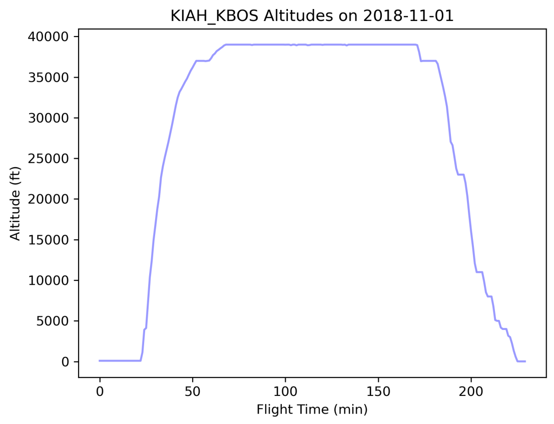


Fig. 11. Sample altitudes (top) and climb rate (bottom) of initial flight plans

$$t_1 = X - z_f - 15000/2000 \tag{9}$$

This process was performed for all flight plans generated in both datasets, and models using the tuned hyperparameters in Table VIII were trained using the adjusted flight plans and original feature cubes. The adjustment process was not guaranteed to work for all flights, due to some flights being too brief to yield real solutions, and as a result the total set of flights were reduced - with 60 fewer flights in the 2 weeks



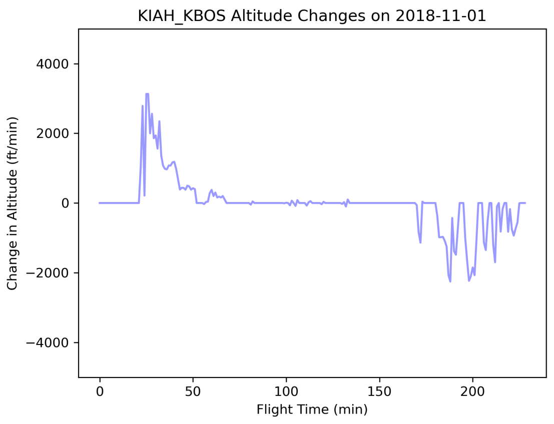


Fig. 12. Sample altitudes (top) and climb rate (bottom) of observed flights

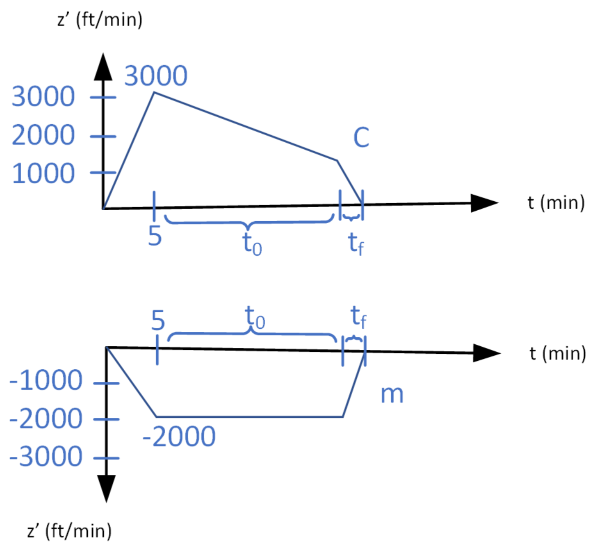


Fig. 13. General approach to calculating ascent (top) and descent (bottom) rates

generalized dataset.

1) Results: Table X summarizes the training of identical models using the different flight plans, based on the final tuned models. In addition to the horizontal and vertical error, the number of cross-validation folds which failed to converge is included; this value generally increased, which may reflect that the complexity of the modified ascent and descent phases inhibited the retention of useful features from the flight plan. Additionally, no notable improvements were made on either error; in fact, vertical error appears to marginally increase with the adjusted flight plans. Several reasons for this increase are suspected, the most likely of which being the complexity of the ascent and descent phases of flight. From observation, it is not uncommon for aircraft to be held above or below their filed cruising altitude during takeoff, as a method of handling high congestion near airports. These altitude holds cannot be predicted from weather conditions alone, and would require additional NAS information to potentially predict - making it difficult to accurately estimate aircraft altitudes close to airports and other centers. Minor contributing factors to the increase in error include a sub-optimal quadratic fit to the cruising altitude, and the increased fidelity of flight plan altitudes providing unnecessary information, adversely impacting training.

C. Evaluating Improvements Over a Test Route

From the results of generalizing the training and validation datasets, it became clear that some amount of increase in error could be expected. However, with approximately 375 airports in the continental United States, collecting and training over each route would be untenable. It is therefore significant to understand the challenges associated with predicting over routes excluded from training. This section considers such an event by evaluating the prior trained models on a test route from Chicago to New York (indicated in red in Figure 6 and Table V). The trained sets of tuned models for each type of flight plan were re-used, as their model states had been saved. Because each model in the four-fold cross validation had no prior knowledge of the test route, each flight was predicted multiple times, once by each model.

1) Results: Results indicate a drastic increase in error for models evaluated over unknown routes. Because the data is within the same 2-week period, and the flight route is within the same region of the United States as other flights present in the training data, it can be decidedly ruled that this affect is not related to potentially unobserved weather behavior. This leaves the assumption that the model performed poorly due to a lack of exposure to the flight plan and trends in rerouting. Throughout this research, models have improved over the accuracy of flight plans; inspecting one of these predictions, as in Figure 14, illustrates how these improvements are often made by marginal deviations from the original flight plan. It is therefore key to understand that current approaches rely on an embedded degree of knowledge based on the flight plans provided in training, a key indicator of overfitting at a larger scale.

While all models performed unacceptably on the unknown route, not all performed equally. From prior comparisons of validation scoring of tuned models on the original and adjusted flight plans, it is known that marginal error increases are expected with the adjusted altitude; however the opposite occurred, indicating how large of a difference can occur in

TABLE X
Comparison of tuned models using assumed and adjusted altitudes

Model	Initial Failed Folds	Initial Horizontal Error	Initial Vertical Error	Adjusted Failed Folds	Adjusted Horizontal Error	Adjusted Vertical Error	Improvement over Initial Horiz. Error	Improvement over Initial Vertical Error
CNN-LSTM	0	40.4594	1175.34	1	40.5796	1256.50	-0.297	-6.905
		33.3621	1601.43		33.4905	1716.29	-0.385	-7.172
CNN-GRU	0	41.6861	1158.91	1	39.3819	1171.08	5.528	-1.050
		36.7627	1585.86		37.0114	1621.20	-0.677	-2.228
SA-LSTM	1	41.9151	1201.79	0	42.2542	1297.17	-0.809	-7.936
		38.5151	1543.64		32.8472	1671.91	14.716	-8.310

TABLE XI
Comparison of tuned models using initial flight plans over a test route

Model	Validation Horizontal Error (μ/σ in nmi.)	Validation Vertical Error (μ/σ in ft.)	Test Horizontal Error (μ/σ in nmi.)	Test Vertical Error (μ/σ in ft.)	Improvement over Validation Horiz. Error (μ/σ in %)	Improvement over Validation Vertical Error (μ/σ in %)
CNN-LSTM	40.4594	1175.34	137.1898	3206.68	-239.08	-172.83
	33.3621	1601.43	58.3072	3899.85	-74.771	-143.522
CNN-GRU	41.6861	1158.91	70.5607	3199.60	-69.267	-176.087
	36.7627	1585.86	31.6596	3861.28	13.881	-143.482
SA-LSTM	41.9151	1201.79	270.6174	6171.50	-545.633	-413.524
	38.5151	1543.64	99.3068	3869.39	-157.839	-150.668

TABLE XII
Comparison of tuned models using adjusted flight plans over a test route

Model	Validation Horizontal Error (μ/σ in nmi.)	Validation Vertical Error (μ/σ in ft.)	Test Horizontal Error (μ/σ in nmi.)	Test Vertical Error (μ/σ in ft.)	Improvement over Validation Horiz. Error (μ/σ in %)	Improvement over Validation Vertical Error (μ/σ in %)
CNN-LSTM	40.5796	1256.50	115.574	3188.76	-184.809	-153.781
	33.4905	1716.29	53.3834	3325.29	-59.399	-93.748
CNN-GRU	39.3819	1171.08	77.5868	2012.65	-97.011	-71.864
	37.0114	1621.20	31.1632	2338.64	15.801	-44.254
SA-LSTM	42.2542	1297.17	269.080	3556.86	-536.813	-174.201
	32.8472	1671.91	70.8284	3671.12	-115.630	-119.577

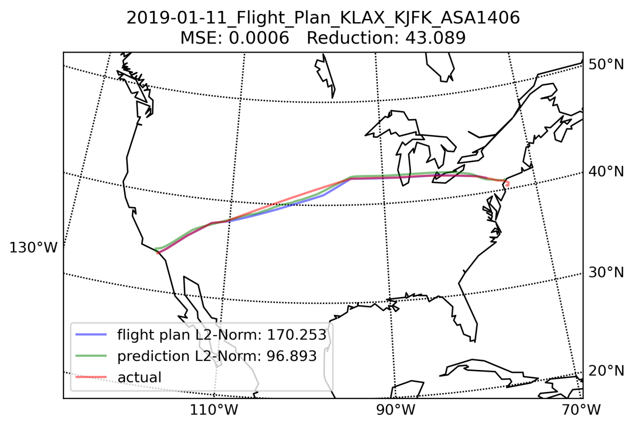

2019-01-11_Flight_Plan_KLAX_KJFK_ASA1406
MSE: 0.0006   Reduction: 43.089
50°N
40°N
30°N
20°N
130°W
110°W
90°W
70°W
flight plan L2-Norm: 170.253
prediction L2-Norm: 96.893
actual

Fig. 14. Sample flight prediction

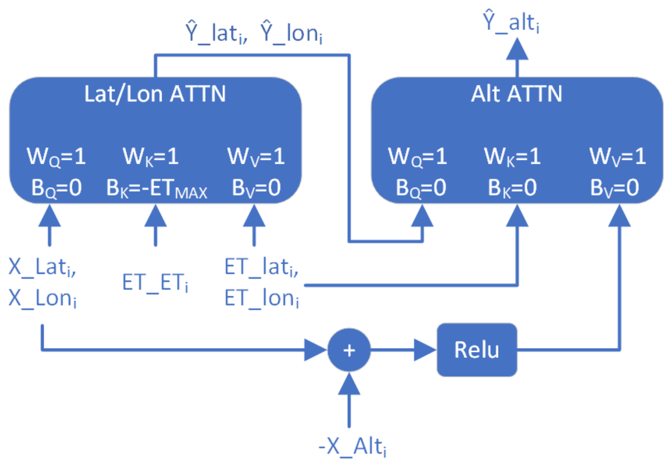


Fig. 15. Sample decision process architecture

test data error and between individual training sessions. At the same time, CNN-GRU appears to be the least-adversely affected model of those tested; the model is the only to not consistently double in horizontal mean error, while also improving on horizontal error deviation. While certainly sub-optimal, the reported horizontal error from CNN-GRU is may fall within an acceptable range. It is anticipated that this outlier in generalization may be a result of GRU's limited cell parameters.

Seeing as no model sufficiently meets the demand of generalization, considerations of how to achieve this generalization are instead provided. Future research may consider the viability of transfer learning and generalization over larger collections of flight routes. Revisiting the selection of optimizers may still provide a resolution to this issue, though achieving useful training results with SGD variants remains a challenge. Furthermore, training with a loss function that does not penalize outliers, such as L1 loss, may be beneficial; other factors - such as lack of coverage and insufficient flight and weather data during preprocessing - may be significant cause for outlying errors, and may penalize the accuracy of the total model as a consequence of their inclusion in training. The complexity of factors affecting each flight phase may require splitting flight predictions into ascent, en-route, and descent phases, which can be supplemented with additional airspace knowledge (such as airspace density or Notices to Air Missions). Finally, design of fundamentally different architectures may be appropriate, specifically to replace the recurrent mechanisms with a form of decision-making; this may be possible with cross-attention (see Figure 15) or well-planned convolution networks.

VI. Conclusion

This paper considered the impact of varied data and architectures within the task of trajectory prediction. After a preliminary establishment of data items and model design, the body of work illustrates how generalization of flight data remains a challenge for trajectory prediction due to increased variance in error. Model tuning and adjustments to existing data are conducted as mitigation techniques, prior to evaluating the improved models on a test flight route. While no model directly improves the accuracy of those presented in current literature, tuned models were able to improve on horizontal or vertical accuracy for select papers; those tuned models also presented a trade-off of error mean and variance from initial research. This potential gains, however, are lost when considering the 70-500% increase in mean error of tuned models over a test route. For these reasons, it is imperative that future research consider the complexities and variety of flight patterns in the proposal of deep learning trajectory prediction models.

Acknowledgment

N.S. thanks Kim Calden (MIT Lincoln Labs), Margit Pawlak (MIT Lincoln Labs), and Tom Webster (FAA) for their support in accessing select CIWS products.

References

[1] E. J. Knoblock, R. D. Apaza, H. Li, Z. Wang, R. Han, N. Schimpf, and N. Rose, "Investigation and evaluation of advanced spectrum management concepts for aeronautical communications," in *Proceedings of the Integrated Communications, Navigation, and Surveillance Conference, ICNS 2021*, 2021.

[2] N. Schimpf, E. J. Knoblock, Z. Wang, R. D. Apaza, and H. Li, "Flight trajectory prediction based on hybrid-recurrent networks," in *2021 IEEE Cognitive Communications for Aeronautical Applications Workshop (CCAAW)*, 2021, pp. 1–6.

[3] S. Ayhan and H. Samet, "Aircraft trajectory prediction made easy with predictive analytics," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 21–30. [Online]. Available: https://doi.org/10.1145/2939672.2939694

[4] Y. Liu and M. Hansen, "Predicting aircraft trajectories: A deep generative convolutional recurrent neural networks approach." *arXiv preprint arXiv:1812.11670*, 2018.

[5] Y. Pang, H. Yao, J. Hu, and Y. Liu, *A Recurrent Neural Network Approach for Aircraft Trajectory Prediction with Weather Features From Sherlock*, 2019. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2019-3413

[6] L. Ma and S. Tian, "A hybrid cnn-lstm model for aircraft 4d trajectory prediction," *IEEE Access*, vol. 8, pp. 134 668–134 680, 2020.

[7] Y. Pang, N. Xu, and Y. Liu, "Aircraft trajectory prediction using lstm neural network with embedded convolutional layer," in *Conference of the PHM Society 11*, ser. PHM 2019. Scottsdale, AZ, USA: Prognostics and Health Management Society (PHM), 2019, pp. 1–10. [Online]. Available: https://doi.org/10.36001/phmconf.2019.v11i1.849

[8] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.355v1*, 12 2014.

[9] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao, "Independently recurrent neural network (indrnn): Building a longer and deeper rnn," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5457–5466.

[10] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/3416a75f4cea9109507cacd8e2f2aefc-Paper.pdf

[11] Y. Lecun and A. Canziani, "Deep learning: Ds-ga 1008, spring 2020," in *https://atcold.github.io/pytorch-Deep-Learning/*, 2021, accessed: 2020-11-15.

[12] N. Schimpf, "Weather preprocessing," in *Github*. https://github.com/schimpfen/Weather-Preprocessing, 2021.

[13] ——, "Flight track prediction," in *Github*. https://github.com/schimpfen/Flight-Track-Prediction, 2021, development in-progress.

[14] M. M. Eshow, M. Lui, and S. Ranjan, "Architecture and capabilities of a data warehouse for atm research," in *2014 IEEE/AIAA 33rd Digital Avionics Systems Conference (DASC)*, 2014, pp. 1E3–1–1E3–14.

[15] J. Klingle-Wilson, D.; Evans, "Description of the corridor integrated weather system (ciws) weather products," Lincoln Laboratory, Massachusetts Institute of Technology, Tech. Rep., 2005.

[16] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht, "The marginal value of adaptive gradient methods in machine learning," 2018.