A Natural Language Understanding Approach for Digitizing Aircraft Ground Taxi Instructions

Hillel Steinmetz *, Jacob Tao †, Stephen Clarke ‡ and Krishna M. Kalyanam §
NASA Ames Research Center, Moffett Field, California, 94035

Advancements in natural language processing (NLP) technologies offer a unique opportunity to furnish aircraft crews, primarily pilots, with digital instructions for taxiing operations. Digital taxi instructions, delivered either as text or graphics, can streamline taxiing procedures, thereby reducing radio congestion, minimizing communication errors, and enhancing aircraft monitoring. Techniques used for natural language understanding (NLU), a subset of NLP focused on machine comprehension of natural language, can extract taxi instructions directly from verbal radio communications. This capability paves the way for implementing a digital taxi communication framework with minimal adjustments to the existing air traffic controller operations. This paper delves into a novel application of NLU: the automated generation of digital taxi instructions from air traffic controller speech. We detail the development of an annotation scheme to represent aircraft ground traffic communications within the US National Airspace System (NAS), employing intent classification (IC) and slot filling (SF) to extract taxi instructions using NLU models. Several neural network models were trained on a dataset annotated with our scheme, achieving notable accuracy and F_1 scores. Our research demonstrates the feasibility of using NLU to automatically generate digital taxi instructions, showcasing its potential to streamline the implementation of digital taxi communications.

Nomenclature

ATC	=	Air Traffic Control
$ATCo$	=	Air Traffic Controller
ATM	=	Air Traffic Management
FFN	=	Feed-Forward Network
IC	=	Intent Classification
$LSTM$	=	Long Short-Term Memory
NAS	=	(U.S.) National Airspace System
NLP	=	Natural Language Processing
NLU	=	Natural Language Understanding
SF	=	Slot Filling

I. Introduction

In January 2023, Delta Flight 1943 was forced to abort its takeoff at John F. Kennedy International Airport (JFK) when another aircraft began crossing its assigned runway. The taxiing aircraft, flight American Airlines Flight 106, was initially instructed to cross runway 31L at taxiway J. Later, a controller instructed it to cross runway 4L at taxiway K. The pilot, however, followed the controller's initial instruction and crossed runway 31L at taxiway. An initial report by the Federal Aviation Administration (FAA) found the American Airlines crew to be distracted and recommended developing software to detect navigation errors [1]. Since then, several other notable taxiway and runway incursions have occurred, including one in April 2024 at Ronald Reagan National Airport (DCA) after a ground controller cleared a flight to cross the runway where another flight was cleared for takeoff [2]. These incidents underscore the importance of developing new technologies that enhance the safety and surveillance of taxiing operations.

*NASA OSTEM Intern, NASA Ames Research Center
†Senior Data Scientist, NASA Ames Research Center, Universities Space Research Association
‡Senior Aerospace Research Engineer, NASA Ames Research Center, Flight Research Aerospace
§Senior Aerospace Research Engineer, NASA Ames Research Center, AIAA Associate Fellow

Automatically generating digital taxi instructions from air traffic controller (ATCo) speech would enable controllers to send instructions as digital communications with minimal changes to ATCo responsibilities and workload. The use of digital taxi instructions can improve air traffic safety. Currently, ground controllers coordinate with multiple aircraft crews, primarily pilots, over a single radio frequency to issue taxiing instructions. However, exclusive use of voice communication for air traffic control operations is vulnerable to error. The human auditory processing channel is not efficient at processing and memorizing information, especially over noisy channels with unfamiliar interlocutors [3, 4]. Digital taxi instructions can represent taxiing instructions as text or can be shown on a graphical display (similar to a map application for road traffic) and enable an aircraft crew to refer back to instructions. Displaying a map in the flight deck addresses issues of pilot auditory processing and memory. Digitizing taxi instructions also improves the surveillance of aircraft movements and can even be used by downstream systems to automatically detect deviations from ATC instructions [5]. Digital instructions also reduce radio congestion, which decreases the likelihood of aircraft crews mishearing communications or failing to hear their callsign [6]. Additional benefits of digital communications are reductions in delays and fuel emissions [7]. For these reasons, digital taxi instructions are likely to improve air traffic management (ATM) systems.

However, digital taxi instructions can be difficult to implement. Data communications cannot entirely replace voice communications. ATCos and aircraft crews can communicate faster using verbal instructions. Verbal instructions also enable aircraft crews and ATCos to convey non-linguistic information (for example, urgency or emotion) that can be critical for successful communication [8]. Earlier research on replacing voice communication with digital taxi instructions has led to mixed results, with some ATCos indicating that voice communications may continue to be necessary for routine communications [9, 10]. For these reasons, many proposed implementations of digital taxi instructions recommend mixed-mode communication, such as a 2020 proposal by FAA [5]. The continued importance of voice communications provides compelling opportunities to use NLU to incorporate digital taxi communications into ATM.

II. Related Work

Several human factors studies have explored the impact of digital taxi instructions on ATCo and aircraft crew workload. In one simulation, ground controllers issued digital instructions to aircraft using a touchscreen. The touchscreen displayed a map of airport taxiways that ground controllers could interact with by pressing or dragging elements of the map to issue or alter taxi instructions. The study found that pilots preferred digital taxi communications to voice communications because it lessened communication responsibilities and reduced readback and hearback errors [9]. Another study examined the feasibility of pilots using navigation maps in simulations with a prescribed area to remain inside. The study found that pilots in the large tolerance area condition perceived the map to be safer than those who did not use a map, with no effect on workload [11]. A recent FAA study outlined the procedural changes, benefits, and scope of implementing digital taxi instructions [5]. While there are clear benefits to digital taxi communications, it remains difficult to replace verbal communications; the FAA study still proposed requiring that ATCos and pilots establish radio contact.

Recent advancements in natural language processing (NLP) have sparked interest in its potential applications in aviation. Language models specifically tailored to the aerospace vocabulary have been trained and deployed to classify narrative reports for safety investigations [12, 13]. Other studies over the past decade have shown that automatic speech recognition (ASR) and NLU can be used to detect readback errors [14, 15], automatically file pilot weather reports [16], pre-fill radar label entries [17], and alert ground controllers that they instructed pilots to use a closed runway [18]. A recent Single European Sky ATM Research (SESAR) study used ASR to pre-fill radar labels, and found that ASR had reduced ATCo workload ratings and improved ATCo performance on situational awareness measures [17].

At the time of publication, there are no publicly available datasets that label taxiing instructions in ATC communications. However, previous work produced ontologies or taxonomies that can be used to label ATC communications. The German Aerospace Center (DLR) and the MITRE Corporations developed ontologies for classifying ATCo communications [19, 20]. Both ontologies define command types for desired actions to be taken by pilots or ATCos, such as requesting a pilot to descend or provide clearance. Each command type has several associated qualifiers; for instance, a *descend* command is associated with altitude measurement in feet. Only the DLR ontology contains representations for taxiing operations [19]. The DLR ontology was later used to develop a rules-based algorithm for extracting taxiing instructions, as detailed in Helmke et al. [21]. However, a rules-based approach may not successfully adapt to non-standard phraseology and is sensitive to errors in speech transcriptions. It also needs to be revised for different airspaces that use other terminology. NLU models, which are probabilistic, are more likely to successfully

extract taxi instructions from text that employs non-standard phraseology (or contains transcription errors).

III. Annotating Ground Control Communications

To train NLU models to classify ATCo/pilot speech utterances, we developed an annotation scheme and annotated 3.7 hours of transcribed speech (audio) files. The annotation scheme was developed in collaboration with subject-matter experts (SMEs) at the National Aeronautics and Space Administration (NASA) and the FAA. To create the dataset, we obtained speech (audio) files from LiveATC.net* containing radio communication between ground controllers and pilots at Dallas-Fort Worth Airport (DFW) that occurred on August 28th, 2022. The audio files were segmented using a voice activity detector (VAD), dividing long audio segments into shorter ones. The resulting segments were transcribed using the Microsoft Azure *Speech-to-Text* service [22] before being validated and corrected by trained annotators. Subsequently, the utterances in the transcriptions were manually labeled with speaker IDs that also mark whether the speaker is a pilot or ATCo. Two trained annotators used the annotation scheme described below to label pilot and ATCo utterances with intent and slot labels. Prodigy [23], a Python-based annotation tool, was used to create the interface the annotators used to label the data.

A. Digital Taxi Annotation Scheme

Using the terminology adopted by MITRE and DSR to describe their ontology [19], our annotation scheme categorizes utterances as containing *command types* and extracts *qualifiers* that modify command types. Command types describe the instructions or requests, such as "go to", "hold", or "provide information". Qualifiers represent the details critical to an instruction or request, such as "taxiway Bravo" in the instruction "hold at taxiway Bravo".

The annotation scheme utilizes intent classification (IC) and slot filling (SF) to annotate command types and qualifiers in an utterance. The IC-SF scheme is effective for representing task-oriented dialog, where the speaker (or user) aims to have another agent perform a specific action. *Intents* describe what the speaker requires from the agent, while *slots* provide information in the utterance related to that intent [24]. Typically, intent labels are assigned to the entire utterance. An utterance can also be divided into a sequence of tokens: a token represents a word, punctuation, or meaningful sub-words (such as "-n't" in the words "doesn't" or "wouldn't"). SF labels a sequence of tokens, also known as a *span*, with zero, one, or multiple slot labels. An illustration of an annotated utterance is shown in Figure 1.

Speaker	Utterance and slots	Intents
Controller	[American 2621]$_{\texttt{to\_whom}}$ [DFW Ground]$_{\texttt{from\_whom}}$. Good morning. Taxi via [Echo-Sierra-Kilo to the ramp]$_{\texttt{taxi\_route}}$	Go to

Fig. 1 An example of an annotated utterance

In our annotation scheme, IC was used to label each utterance with zero or more command types, and SF was used to label the spans that qualify the command types. A single utterance was labeled with one or more intents (command types). Pilot readbacks were labeled with an "acknowledge" intent alongside any intents contained in the instructions repeated by the pilot. The slot labels (qualifiers) consist of identifiers—such as "DFW Ground" or an aircraft callsign—and other information relevant to the intents, such as a taxi route or apron entry point. Utterances were split into lists of tokens using spaCy's whitespace tokenizer for English [25]. Each token is labeled with zero, one, or more slots. A complete list of intents and slots used by our annotation scheme can be found in tables 2-3 in the Appendix. The resulting annotations are not taxiing instructions, but are intended to be utilized by a downstream dialog manager to generate taxi instructions.

Our annotation scheme differs in a few ways from the DLR ontology described in Helmke et al. [20]. First, our annotation scheme was developed using phraseology commonly employed in the NAS, with particular attention to the phraseology discussed in the FAA's Order JO 7110.65 [26]. As a result, our command types (intents) include instructions such as *give way* and *pass*. We also included questions as intents because ATCos frequently asked pilots where they were going or for the callsign of an aircraft at a particular location. Finally, we annotate command qualifiers as slots; phrases that qualify a command are annotated at the word level rather than at the sentence level.

*https://www.liveatc.net

An advantage of using an IC and SF annotation scheme is that it captures taxiing instructions at both the word level and sentence level. This multi-level representation allows for training an NLU model to focus on specific linguistic contexts. Moreover, it supports multitask learning, where models are trained on both IC and SF tasks simultaneously, enhancing their ability to understand the relationship between span-level slots and sentence-level intents [24]. However, an IC-SF annotation scheme does not explicitly define the relationships between particular slots and intents. Therefore, extracting taxi instructions from model predictions requires a downstream dialog manager to connect intents (command types) to particular slots (qualifiers). We mitigated this issue by implicitly defining relationships between intents and slots. For example, the slot label *at <location> give way* associates a location to the command *give way*. These relationships should help models implicitly learn the association between certain commands and qualifiers.

B. Dataset Description

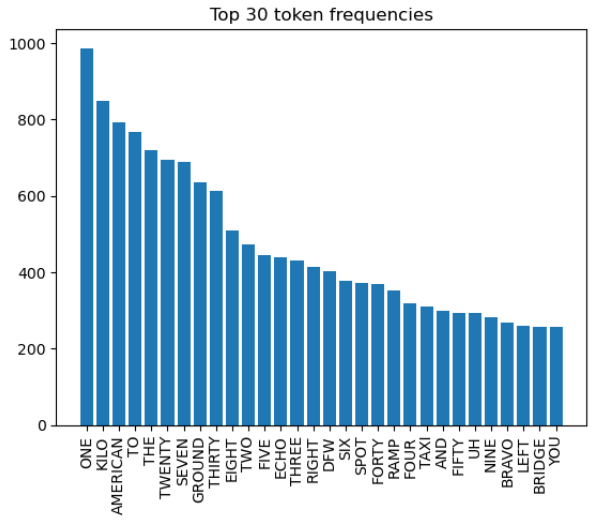

Fig. 2 Counts of the 30 most-frequent tokens.

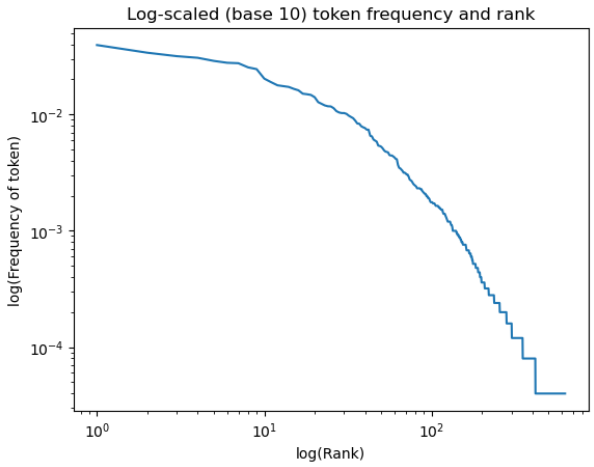

Fig. 3 A $\log_{10}$-scaled plot of token frequencies (counts) versus token rank

Our dataset contained 1978 utterances and 32,243 tokens. Of the 32,243 tokens in the data, there are 651 unique tokens. The token frequencies formed an interesting distribution. While the distribution is near-Zipfian like most distributions in natural language [27], the token frequency first decreases gently along with rank before decreasing steeply. This pattern implies that tokens tend to fall into two categories: common or uncommon, indicating that ATCo-pilot communications are more restricted than other forms of speech. This pattern is illustrated in Figure 2 and 3, which shows the log-scaled plot of token frequencies vs. rank.[†]

The distribution of intent and slot labels in the annotated dataset was imbalanced (see tables 2-3 in the Appendix for label frequencies), reflecting the fact that most ATC communications are routine instructions. While the annotation scheme was designed to capture most facets of ATC communications and taxiing instructions, we removed some low-frequency labels from the dataset, which we determined to be labels with fewer than 30 annotations. However, we did not filter the *pass* intent given its importance as a taxiing instruction, despite having fewer than 30 annotations. This filtering procedure avoids overfitting the data to a small set of occurrences at training time.

Using canonical correlation analysis [28], we found a number of our intents and slots to be correlated. Some correlations reflected the annotation scheme design: the *frequency* slot and *change frequency* intent had a correlation coefficient of 0.9. Others, such as *spot*, were strongly correlated with several intents, such as pilots informing ATCos of their destination or ATCos asking for the callsign of an aircraft at a particular location.

IV. Methods

To evaluate the quality of the dataset and annotation scheme, we trained and evaluated baseline feedforward neural networks (FFNs) and multitask long short-term memory (LSTM) networks, a type of recurrent neural network

[†]Token rank assigns the most frequent token is assigned a rank of 1, the second most frequent token is assigned a rank of 2 and so on.

(RNN) model. The baseline models were trained separately on IC and SF tasks. These baseline FFNs were used to investigate whether our small dataset accurately captures intents and slots, as well as identify several areas for potential improvements. The LSTM models were trained simultaneously on IC and SF tasks and were used to determine whether a model can leverage the systematic relationships between slots and intents.

We trained and evaluated models on both ATCo and pilot utterances. Pilot utterances can augment ATCo data since they contain similar phraseology. Readbacks, in particular, can be helpful since they were annotated with nearly the same intents and slots as the preceding instruction. However, pilot speech was likely to deviate from prescribed phraseology, which could diminish overall IC performance. Therefore, we also trained models on data exclusively consisting of ATCo utterances to serve as a point of comparison. In total, we trained and evaluated six models. We trained two LSTMs to predict both IC and SF labels: one was trained on ATCo and pilot data and the other was trained on ATCo-only data. Separate FFNs were trained on IC or SF tasks. Therefore, we trained a total of four FNNs: two were trained on ATCo and pilot data and two were trained on ATCo-only data.

The dataset was split into training and evaluation sets using a ratio of 4:1. Intents were stratified proportionally across the training and evaluation sets. Because intents were correlated with slots, we used intent labels as a proxy for both label types when stratifying the data. All tokens that appeared fewer than two times in the training set were transformed into an "unknown" token, $[UNK]$. We performed this transformation to train models to generalize to unseen tokens. Similarly, tokens not seen in the training set are transformed into the $[UNK]$ token.

A. Baseline models

For our baseline models, we trained separate FFN classifiers to predict intents and slots. Each classifier consisted of 3 hidden layers with 100 neurons, with the rectified linear unit (ReLU) functions as activation functions. Both FFN classifiers were trained using the Adam optimizer, a variant of stochastic gradient descent (SGD) [29]. Cross-entropy (CE) loss was used as the loss function for both IC and SF tasks.

The baseline IC model used vectors of token counts in an utterance as inputs. Each index of the vector corresponds to a token in the training set's vocabulary. The values assigned to each index are the counts of the times the token appeared in the utterance. More formally, for the set of utterances in the data, $U = \{u_1, ..., u_n\}$, the input was a vector $\mathbf{x}$, of length $|V|$ where $V = \{t_0, ..., t_n\}$ is the set of unique tokens in the training data and $\mathbf{x} = [\texttt{count}(t_0, u_j), ..., \texttt{count}(t_n, u_j)]$ and $\texttt{count}(t_i, u_j)$ is the number of times t_i occurred in the utterance u_j.

For the baseline SF model, our inputs were concatenated one-hot vectors of the current token and the previous n tokens. More formally, the inputs were vectors of size $n \cdot |V|$, where n is the number of previous words encoded as features and $|V|$ is the size of the training set vocabulary. For our experiments, we set n to 5. The resulting input vector looked like: $[0, 0, 1, ..., 0, 1, 0, ..., 0, 0, 1] = [OH(\texttt{prev}_n), ..., OH(\texttt{prev}_1), ..., OH(\texttt{token})]$ where OH is a function that generates a one-hot vector from the token text. Any $[UNK]$ tokens in the training data were ignored (and were mapped to a vector of zeros). If the token index i is less than the number of preceding tokens n (i.e., if the current token is one of the first tokens in the sentence), then all preceding tokens greater than i were set to a beginning of the sentence token, $[BOS]$.

B. Multitask model

The LSTM models were trained to simultaneously predict IC and SF labels, a training scheme known as multitask models [30–32]. When tasks are sufficiently similar, multitask models perform better than independently trained models and have fewer parameters [33]. In several IC-SF datasets, slots and intents tend to be correlated, but their performance might relies on different linguistic information (for example, SF may be more dependent on syntactic information). A model trained on both tasks can utilize more varied linguistic information and perform better than models independently trained on IC and SF tasks [24]. The restricted syntax and vocabulary of ATCo/pilot communications are information-dense, suggesting that tokens carry information relevant to both slots and intents. Therefore, we hypothesized that training a multitask model on IC and SF was the optimal approach for classifying digital taxi data.

The multitask model architecture consisted of a bidirectional LSTM (BiLSTM) and that branches into two FFN networks. One branch serves as an intent classifier and the other a slot classifier. We represented tokens as one-hot vectors. The LSTM generated a sequence of hidden states from the sequence of one-hot vectors. The outputs of the hidden layers were fed to the FFN slot classifier as a batched input, outputting slot probabilities for each token. The FFN intent classifier took the max-pooled values of hidden states. The max-pool operation obtained the maximum value at each index of the hidden states. So, for hidden state vectors of length m, often represented as a matrix $H = [\mathbf{h}_1, ..., \mathbf{h}_n]$, then, $\text{maxpool}(H) = [\max(h_{1,1}, ...h_{1,m}), ..., \max(h_{n,1}, ...h_{n,m})]$, where $h_{i,j}$ is the value of hidden state i at index j.

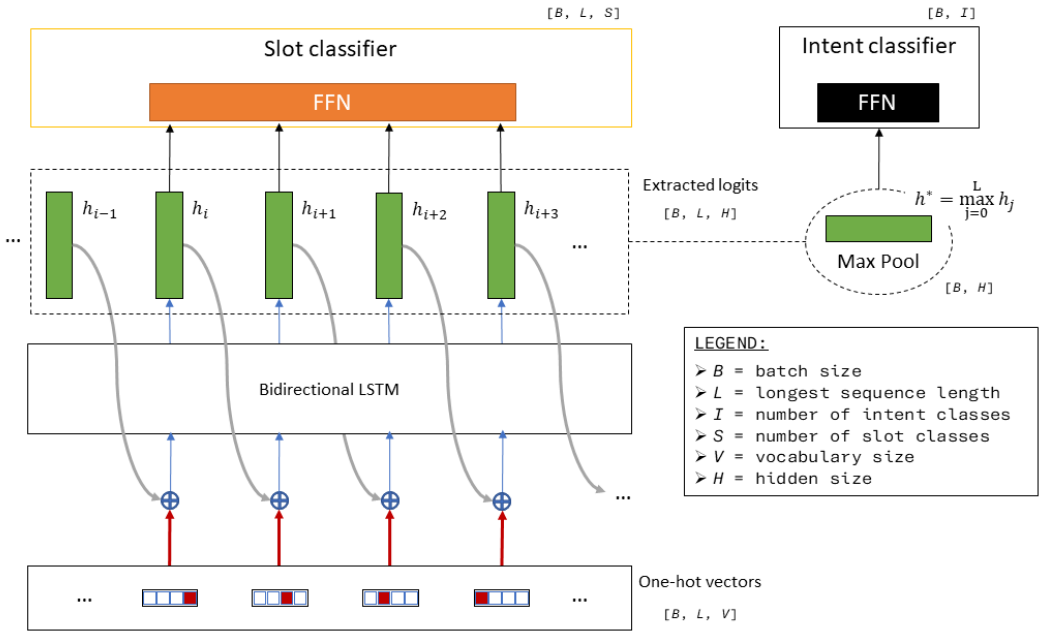

Fig. 4 An overview of the proposed multitask learning LSTM model

The intent classifier used this pooled value as its input to produce an output of intent probabilities. Our resulting multitask model was lightweight, with only approximately half a million parameters. In comparison, BERT, a typical transformer-based model, boasts around 110 million parameters [34]. The model architecture is shown in Figure 4.

The LSTMs were trained using the AdamW algorithm [35], a variant of Adam that uses a different method for applying weight decay. The loss function for both IC and SF tasks is binary cross entropy (BCE) loss. BCE loss was calculated for each label type (or class), with the total loss corresponding to the mean class loss. The loss function as applied to a single batched element is shown in (1), where i is a label, x_i is the logit predicted for the label i, y_i is the true value of the label i, σ is the sigmoid function, N is the total number of labels, and w_i is a weight applied to positive instances of label i to improve recall.

$$L_{BCE} = \frac{1}{N}\sum_{i=0}^{N} w_i \cdot y_i \cdot \log \sigma(x_i) + (1 - y_i) \cdot \log(1 - \sigma(x_i)) \tag{1}$$

Because SF losses were calculated for each token, the model's parameters were updated using the mean loss of the slot predictions of all tokens in an utterance. The total loss for an utterance, $\mathcal{L}$, was calculated using (2), where α is a hyperparameter, $\mathcal{L}_{\mathrm{SF},t_j}$ is the BCE loss calculated for token t_j the sequence of tokens in the utterance, $T = (t_1, ...t_n)$. The total loss for all utterances in a batch was calculated as the mean loss of the utterances in the batch.

$$\mathcal{L}_i = \alpha \mathcal{L}_{\mathrm{IC}} + (1 - \alpha)\frac{1}{n}\sum_{j=0}^{n} \mathcal{L}_{\mathrm{SF},t_j} \tag{2}$$

C. Evaluation

Our NLU models were evaluated using three measures: macro F_1 score, micro F_1 score, and accuracy. F_1 and accuracy scores were used to evaluate performance on both IC and SF tasks. Accuracy was defined as the number of correctly labeled utterances or tokens as a fraction of the total number of utterances or tokens. F_1 score was calculated according to (3):

$$F_1 = \frac{tp}{tp + \frac{1}{2}(fp + fn)} \tag{3}$$

where tp, is the number of true positives, fp is the number of false positives, and fn is the number of false negatives. A micro F_1 score is the weighted average of the F_1 score for all the labels, with the weights corresponding to the number of

times the label occurs in the evaluation set. A macro F_1 score is the uniformly-weighted average F_1 score of the labels.

V. Results

Table 1 contains the F_1 scores of our independently trained FFN classifiers and multitask LSTM model. The tables show the scores of each model architecture when trained on datasets containing solely ATCo utterances or both ATCo and pilot utterances. The highest performance score for each metric is in bold text.

			Macro F_1	Micro F_1	Accuracy
Intent Classification	FFN	ATCO+Pilot	74.6	89.3	77.8
		ATCO-only	77.2	90.1	80.4
	Multitask LSTM	ATCO+Pilot	81.4	90.4	80.3
		ATCO-only	**84.1**	**94.3**	**86.6**
Slot Filling	FFN	ATCO+Pilot	77.7	93.0	93.4
		ATCO-only	75.1	93.6	94.3
	Multitask LSTM	ATCO+Pilot	**85.5**	**94.3**	**94.5**
		ATCO-only	79.2	93.5	93.4

Table 1 F_1 and accuracy scores for baseline FFN classifiers separately trained on IC and SF tasks and multitask LSTM trained simulatenously on IC and SF tasks. The Multitask LSTM models are the same model across SF and IC tasks, while the FFNs are distinct model across tasks.

The baseline FFN classifiers achieved commendable performance on the tasks, with micro-F_1 scores approaching 90% on both IC and SF tasks. As hypothesized, the multitask LSTMs generally outperforms the baseline models. However, SF performance degraded slightly when the multitask model was trained exclusively on ATCo utterances. The largest performance improvements were observed in the IC task, whose accuracy scores were 2.5% and 6.2% higher than baseline models when trained on the ATCo+Pilot and ATCo-only data respectively. Overall, the results demonstrate that the multitask model surpasses the baseline models.

The LSTM's performance varied across specific intent and slot labels. In the LSTM trained on ATCo and pilot data, the intent with the highest F_1 score was *hold* (100); the intent with the lowest F_1 score was *inform* (48.0); the slot with the highest F_1 score was *frequency* (97.5), and the slot with the lowest F_1 score was <*Vehicle*> - *pass* (55.6). We hypothesize that the labels with poor performance lack a restricted set of linguistic cues (for instance, <*Vehicle*> - *pass* can take on spans such as "the babybus", "A318", "the airbus", or "United", among many others).

VI. Discussion

Our annotation scheme and experiments demonstrate the feasibility of using NLU to generate digital taxi instruction directly from controller speech. The performance of the baseline FFN models was quite good, at nearly 90% micro-F_1 scores. The fact that baseline FFN models can achieve this performance indicates that the annotation scheme and dataset are well-constructed.

Our multitask model performed better on both IC and SF tasks than the separately trained FFN models, achieving higher F_1 and accuracy scores. Our experiments also show that—for both model types—IC performance degraded while SF performance improved when pilot utterances were included in the training and evaluation data sets. Performance on IC task may have been lower when trained on ATCo+Pilot data since pilot speech tended to be more variable than ATCo speech, making it more difficult to classify intents. This assumption is supported by the fact that F_1 and accuracy scores of ATCo-only utterances within ATCO+Pilot data were higher than F_1 and accuracy scores of the whole dataset. Concerning the improvements in SF performance across models trained on the ATCo+Pilot data compared to ATCo-only data, we hypothesize that the inclusion of pilot data, particularly readbacks of instructions, provided more examples for the model to learn to extract slots. We believe that pilot readbacks were particularly helpful for teaching models low-frequency slots.

The multitask training method seemed to enhance performance on the IC task. It is unclear how much multitask

learning improves model performance on the SF task: the accuracy scores of the multitask model are only 1.1% higher than baseline models when trained on the ATCo+Pilot data, and 0.9% lower than baseline models when trained on ATCo-only data. Perhaps this discrepancy implies that the benefits of multitask learning are asymmetric for our dataset: the IC task benefits more from learning representations applicable to SF than the converse.

While there is room for improvement, our multitask BiLSTM model shows promise. The model achieved performance scores similar to other NLU models that extract information from air traffic communications. For instance, Chen et al. [16] trained an LSTM model that extracted pilot weather reports with a F_1 score of 78.9. As a part of a larger digital taxi system that incorporates contextual data, such as flight plans or surveillance data, we believe that overall performance is likely to improve; similar performance boosts can be observed in Ahrenhold et al. [17], whose speech-to-text model's callsign recognition accuracy improved from 71.6% to 97.8% after providing their model with a set of possible callsigns using surveillance data of aircraft movements.

Several challenges were identified pertaining to the use of NLU to generate digital taxi instructions. Misspoken words, self-corrections, disfluencies, and noise in ATC speech are all potential roadblocks for dataset annotation and model training. On several occasions in the recordings, controllers use disfluencies in the middle of an instruction (for example, "via Kilo, uh, Kilo, Kilo six"), or were interrupted by crosstalk. While we successfully removed disfluencies and noise from the data, the success of an NLU model depends on upstream capabilities to filter noise and disfluencies from the data. Self-corrections, on the other hand, cannot as easily be identified and removed from the data. Additionally, self-corrections may require a slot to be annotated twice in a single utterance; for example, in "gate Bravo eight, erm, Bravo nine", annotating only "Bravo nine" without sufficient data may result in a model failing to categorize Bravo eight as a gate in other utterances. The presence of multiple annotated slots means that a downstream dialog manager is needed to extract the best candidate from multiple slots.

Another potential challenge is speech that deviates from the phraseology detailed in JO 7110.65 and the idiosyncratic phraseology of certain airports. For instance, at DFW, pilots taxiing to their gates are typically told to travel to a numeric Apron Entry Point (AEP) called "spots" rather than to a particular gate. This idiosyncrasy suggests that our annotation scheme may need to be modified to reflect the differences across airports. Future research may be interested in developing a large language model of ATC speech and finetuning on airport-specific annotations.

VII. Conclusion

The annotation scheme and models presented in this paper demonstrate the feasibility of using NLU to synthesize digital taxi instruction directly from ATCo speech. Our research developed an annotation scheme for NAS that labels whole utterances and spans of text to extract details necessary to generate taxi instructions. Our lightweight, multitask LSTM demonstrates significant potential for using NLU to generate digital taxi instructions, which could ultimately enhance the safety of ground traffic movements. Future work will investigate other NLU tasks such as dialog state tracking (DST), and the use of transformer-based large language models (LLMs) like Bidirectional Encoder Representations from Transformers (BERT) or Generative Pre-trained Transformer (GPT) to fine-tune the IC and SF tasks, aiming to transform model predictions into taxi instructions.

Appendix

Intent	Count (utterances)	Description
Answer	236	Provide an answer to a question.
Request	29	A request issued by a pilot or controller.
Get back to you	22	Will reply soon.
New frequency	101	Change to a different radio frequency.
Turn	124	Turn the aircraft.
Going where	196	Ask a pilot about the location they are scheduled to taxi toward.
Inform	92	Let the controller know that an aircraft is ready to receive instructions.
Hold	66	Do not move beyond a certain point.
Who at location	236	Provide the callsign of an aircraft at a location.
Who needs help	8	Ask if any aircraft/pilot is in need of instructions.
Give way	235	Do not proceed until another aircraft passes.
Question	27	A question raised by a pilot or controller.
Say again	26	Repeat your previous transmission.
Go to	1032	Taxi to a location.
Follow	15	Follow another aircraft.
Behind	54	Pass behind another aircraft.
Correction	22	Correct a previous miscommunication or error.
Ready?	54	Ask if an aircraft is ready to receive taxiing instructions.
Acknowledge	606	Taxi instructions were received.
Going to destination	129	Inform the controller of the destination the aircraft is scheduled to head towards.
Pass	16	Proceed with a taxi route in front of another aircraft.
Set squawk	6	Make sure the transponder is correctly set.
Check ATIS	88	Ask pilots to ensure they have the latest ATIS information.

Table 2 Descriptions of intents used by annotation scheme. (Models are trained on labels in bold text.)

Slot	Count (spans)	Description
Taxi route	1038	The route an aircraft should take to arrive at its destination.
From who	1336	The speaker's identification (callsign or ground).
To whom	862	The identification (callsign or ground) of the intended listener.
Runway	471	A runway identifier.
Request	29	A request issued by a pilot or controller.
Question	27	A question raised by a pilot or controller.
Vehicle	301	A reference to a vehicle.
<Vehicle> - give way	208	An aircraft that should be given way.
<Vehicle> - follow	15	An aircraft that should be followed.
<Vehicle> - pass	13	An aircraft that should be passed.
<Vehicle> - behind	53	An aircraft the listener should be pass behind.
At <location> - hold	62	A location an aircraft should hold.
At <location> - give way	139	A location an aircraft should wait to give way to another aircraft.
Spot (AEP)	497	A surface marking that notes the entrance and exit to the apron.
Gate	14	An airport gate.
ATIS	246	Automatic terminal information service: a broadcast of weather conditions, open runways, and other information.
Direction (L/R)	179	The direction an aircraft should turn.
Squawk code	6	The signal a transponder should emit.
Frequency	96	A radio frequency that aircraft should switch to.
Aircraft location	179	The aircraft's current location at the airport.

Table 3 Descriptions of slots used by annotation scheme. (Models are trained on labels in bold text.)

Acknowledgments

The material is based upon work supported by the National Aeronautics and Space Administration under Contract Number NNA16BD14C, managed by the Universities Space Research Association (USRA). We are grateful for the support and guidance provided by subject-matter experts and other stakeholders at the FAA Office of NextGen.

References

[1] "Runway Incursion and Rejected Takeoff American Airlines Flight 106, Boeing 777-200, N754AN, and Delta Air Lines Flight 1943, Boeing 737-900, N914DU, Queens, New York, January 13, 2023," Tech. Rep. DCA23LA125, National Transportation Safety Board, ???? URL https://www.ntsb.gov/investigations/Pages/DCA23LA125.aspx.

[2] Aratani, L., "FAA investigating near miss at Reagan National Airport," *Washington Post*, 2024. URL https://www.washingtonpost.com/transportation/2024/04/19/faa-investigating-near-miss-reagan-national/.

[3] Nygaard, L. C., Sommers, M. S., and Pisoni, D. B., "Speech Perception as a Talker-Contingent Process," *Psychological Science*, Vol. 5, No. 1, 1994, pp. 42–46. https://doi.org/10.1111/j.1467-9280.1994.tb00612.x, URL http://journals.sagepub.com/doi/10.1111/j.1467-9280.1994.tb00612.x.

[4] Smith, E., "Impact of RNAV terminal procedures on controller workload," *24th Digital Avionics Systems Conference*, Vol. 1, 2005, pp. 5.C.3–51. https://doi.org/10.1109/DASC.2005.1563382, iSSN: 2155-7209.

[5] Ghazavi, N., Masarky, S., Monahan, J., Copp, M., Sanchez, S., David, D., and Supamusdisukul, T., "Operational Evaluation of Digital Taxi Instruction," *2020 Integrated Communications Navigation and Surveillance Conference (ICNS)*, IEEE, Herndon, VA, USA, 2020, pp. 3E1–1–3E1–8. https://doi.org/10.1109/ICNS50378.2020.9222996, URL https://ieeexplore.ieee.org/document/9222996/.

[6] Skaltsas, G., Rakas, J., and Karlaftis, M. G., "An analysis of air traffic controller-pilot miscommunication in the NextGen environment," *Journal of Air Transport Management*, Vol. 27, 2013, pp. 46–51. https://doi.org/10.1016/j.jairtraman.2012.11.010, URL https://linkinghub.elsevier.com/retrieve/pii/S0969699712001536.

[7] Okuniek, J., Gerdes, I., Jakobi, J., Ludwig, T., Hooey, B., Foyle, D., Jung, Y., and Zhu, Z., "A Concept of Operations for Trajectory-based Taxi Operations," *16th AIAA Aviation Technology, Integration, and Operations Conferenc*, AAIA, 2016. https://doi.org/10.2514/6.2016-3753.

[8] Johnson, D. R., Nenov, V. I., and Espinoza, G., "Automatic Speech Semantic Recognition and verification in Air Traffic Control," *2013 IEEE/AIAA 32nd Digital Avionics Systems Conference (DASC)*, 2013, pp. 5B5–1–5B5–14. https://doi.org/10.1109/DASC.2013.6712602, iSSN: 2155-7209.

[9] Truitt, T. R., "An Empirical Study of Digital Taxi Clearances for Departure Aircraft," *Air Traffic Control Quarterly*, Vol. 21, No. 2, 2013, pp. 125–151. https://doi.org/10.2514/atcq.21.2.125, URL https://arc.aiaa.org/doi/10.2514/atcq.21.2.125.

[10] Prinzel, L. L. J., Shelton, K. J., Jones, D. R., Allamandola, A. S., Arthur, J. T. J., and Bailey, R. E., "Evaluation of Mixed-Mode Data-Link Communications for NextGen 4DT and Equivalent Visual Surface Operations," *Air Traffic Control Quarterly*, Vol. 18, No. 2, 2010, pp. 177–212. https://doi.org/10.2514/atcq.18.2.177, URL https://arc.aiaa.org/doi/10.2514/atcq.18.2.177.

[11] Bakowski, D. L., Hooey, B. L., and Foyle, D. C., "Flight deck surface trajectory-based operations (STBO): A four-dimensional trajectory (4DT) simulation," *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*, IEEE, St. Petersburg, FL, 2017, pp. 1–10. https://doi.org/10.1109/DASC.2017.8102008, URL http://ieeexplore.ieee.org/document/8102008/.

[12] Nanyonga, A., Wasswa, H., Turhan, U., Molloy, O., and Wild, G., "Sequential Classification of Aviation Safety Occurrences with Natural Language Processing," *AIAA AVIATION 2023 Forum*, American Institute of Aeronautics and Astronautics, San Diego, CA and Online, 2023. https://doi.org/10.2514/6.2023-4325, URL https://arc.aiaa.org/doi/10.2514/6.2023-4325.

[13] Jing, X., Chennakesavan, A., Chandra, C., Bendarkar, M. V., Kirby, M., and Mavris, D. N., "BERT for Aviation Text Classification," *AIAA AVIATION 2023 Forum*, American Institute of Aeronautics and Astronautics, San Diego, CA and Online, 2023. https://doi.org/10.2514/6.2023-3438, URL https://arc.aiaa.org/doi/10.2514/6.2023-3438.

[14] Chen, S., Kopald, H., Chong, R. S., Wei, Y.-J., and Levonian, Z., "Read Back Error Detection using Automatic Speech Recognition," *Proceedings of the Twelfth USA/Europe Air Traffic Management Research and Development Seminar*, Seattle, WA, 2017.

[15] Helmke, H., Kleinert, M., Shetty, S., Ohneiser, O., Ehr, H., Prasad, A., Motlícek, P., and Ondrej, K., "Readback Error Detection by Automatic Speech Recognition to Increase ATM Safety," *Fourteenth USA/Europe Air Traffic Management Research and Development Seminar (ATM2021)*, Online, 2021. URL https://publications.idiap.ch/attachments/papers/2021/Helmke_ATMSEMINAR-2_2021.pdf.

[16] Chen, S., Kopald, H., Avjian, B., and Fronzak, M., "Automatic Pilot Report Extraction from Radio Communications," *2022 IEEE/AIAA 41st Digital Avionics Systems Conference (DASC)*, IEEE, Portsmouth, VA, USA, 2022, pp. 1–8. https://doi.org/10.1109/DASC55683.2022.9925803, URL https://ieeexplore.ieee.org/document/9925803/.

[17] Ahrenhold, N., Helmke, H., Mühlhausen, T., Ohneiser, O., Kleinert, M., Ehr, H., Klamert, L., and Zuluaga-Gómez, J., "Validating Automatic Speech Recognition and Understanding for Pre-Filling Radar Labels—Increasing Safety While Reducing Air Traffic Controllers' Workload," *Aerospace*, Vol. 10, No. 6, 2023, p. 538. https://doi.org/10.3390/aerospace10060538, URL https://www.mdpi.com/2226-4310/10/6/538.

[18] Kopald, H., and Chen, S., "Design and Evaluation of the Closed Runway Operation Prevention Device," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 58, No. 1, 2014, pp. 82–86. https://doi.org/10.1177/1541931214581018, URL http://journals.sagepub.com/doi/10.1177/1541931214581018.

[19] Chen, S., Helmke, H., Tarakan, R. M., Ohneiser, O., Kopald, H., and Kleinert, M., "Effects of Language Ontology on Transatlantic Automatic Speech Understanding Research Collaboration in the Air Traffic Management Domain," *Aerospace*, Vol. 10, No. 6, 2023, p. 526. https://doi.org/10.3390/aerospace10060526, URL https://www.mdpi.com/2226-4310/10/6/526.

[20] Helmke, H., Slotty, M., Poiger, M., Herrer, D. F., Ohneiser, O., Vink, N., Cerna, A., Hartikainen, P., Josefsson, B., Langr, D., Lasheras, R. G., Marin, G., Mevatne, O. G., Moos, S., Nilsson, M. N., and Perez, M. B., "Ontology for Transcription of ATC Speech Commands of SESAR 2020 Solution PJ.16-04," *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*, IEEE, London, 2018, pp. 1–10. https://doi.org/10.1109/DASC.2018.8569238, URL https://ieeexplore.ieee.org/document/8569238/.

[21] Helmke, H., Matthias Kleinert, Nils Ahrenhold, Heiko Ehr, Thorsten Mühlhausen, Oliver Ohneiser, Lucas Klamert, Petr Motlicek, Amrutha Prasad, Juan Zuluaga-Gomez, Jelena Dokic, and Ella Pinska Chauvin, "Automatic speech recognition and understanding for radar label maintenance support increases safety and reduces air traffic controllers' workload," *Fifteenth USA/Europe Air Traffic Management Research and Development Seminar (ATM2023)*, Savannah, Georgia, USA, 2023. URL https://publications.idiap.ch/attachments/papers/2023/Helmke_ATM2023_2023.pdf.

[22] "Microsoft Azure Speech to Text," , no year. URL https://azure.microsoft.com/en-us/products/ai-services/speech-to-text.

[23] Montani, I., and Honnibal, M., "Prodigy," , no year. URL https://explosion.ai/blog/prodigy-annotation-tool-active-learning.

[24] Louvan, S., and Magnini, B., "Recent Neural Methods on Slot Filling and Intent Classification for Task-Oriented Dialogue Systems: A Survey," *Proceedings of the 28th International Conference on Computational Linguistics*, International Committee on Computational Linguistics, Barcelona, Spain (Online), 2020, pp. 480–496. https://doi.org/10.18653/v1/2020.coling-main.42, URL https://www.aclweb.org/anthology/2020.coling-main.42.

[25] Honnibal, M., Montani, I., Van Landeghem, S., and Boyd, A., "spaCy: Industrial-strength Natural Language Processing in Python," , 2020. URL https://zenodo.org/records/10009823.

[26] "Order JO 7110.65AA - Air Traffic Control," , Apr. 2023. URL https://www.faa.gov/regulations_policies/orders_notices/index.cfm/go/document.current/documentnumber/7110.65, last Modified: 2023-08-22T12:38:13-0400.

[27] Piantadosi, S. T., "Zipf's word frequency law in natural language: A critical review and future directions," *Psychonomic Bulletin & Review*, Vol. 21, No. 5, 2014, pp. 1112–1130. https://doi.org/10.3758/s13423-014-0585-6, URL https://doi.org/10.3758/s13423-014-0585-6.

[28] Hardle, W. K., and Simar, L., "Canonical Correlation Analysis," *Applied Multivariate Statistical Analysis*, Springer Berlin Heidelberg : Imprint: Springer, Berlin, Heidelberg, 2015, 4th ed., pp. 321–330.

[29] Kingma, D. P., and Ba, J., "Adam: A Method for Stochastic Optimization," 2014. https://doi.org/10.48550/ARXIV.1412.6980, URL https://arxiv.org/abs/1412.6980.

[30] Chen, Q., Zhuo, Z., and Wang, W., "BERT for Joint Intent Classification and Slot Filling," , Feb. 2019. URL http://arxiv.org/abs/1902.10909, arXiv:1902.10909 [cs].

[31] Liu, B., and Lane, I., "Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling," , Sep. 2016. URL http://arxiv.org/abs/1609.01454, arXiv:1609.01454 [cs].

[32] Wang, Y., Shen, Y., and Jin, H., "A Bi-model based RNN Semantic Frame Parsing Model for Intent Detection and Slot Filling," , Dec. 2018. URL http://arxiv.org/abs/1812.10235, arXiv:1812.10235 [cs].

[33] Zhang, Z., Yu, W., Yu, M., Guo, Z., and Jiang, M., "A Survey of Multi-task Learning in Natural Language Processing: Regarding Task Relatedness and Training Methods," *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, edited by A. Vlachos and I. Augenstein, Association for Computational Linguistics, Dubrovnik, Croatia, 2023, pp. 943–956. https://doi.org/10.18653/v1/2023.eacl-main.66, URL https://aclanthology.org/2023.eacl-main.66.

[34] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, edited by J. Burstein, C. Doran, and T. Solorio, Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. https://doi.org/10.18653/v1/N19-1423, URL https://aclanthology.org/N19-1423.

[35] Loshchilov, I., and Hutter, F., "Decoupled Weight Decay Regularization," 2017. https://doi.org/10.48550/ARXIV.1711.05101, URL https://arxiv.org/abs/1711.05101.