Bidirectional LSTM for ATC Slot Filling & Transformer-based Readback Verification
Final Report
Ryan Kelley

Abstract

This project developed a 2-stage NLP pipeline pertaining to air traffic control communications. In the first part, a bidirectional LSTM slot-filling model was trained to extract semantic components (e.g., CALLSIGN, INSTRUCTION, ALTITUDE, etc.) from ATC and pilot utterances. The LSTM model was trained on BIO-labeled ATC dialogue data; it achieved an accuracy of 0.81 and a weighted F1-score of 0.77. Then, as an extension of the LSTM labeling, a transformer model was fine-tuned to classify pairs of utterances for readback relevancy and correctness. While this model attained 64.3% accuracy, it struggled to identify the more nuanced–and critical for readback verification–classes: matches_instruction, incorrect_readback, and partial_readback, as shown by their low recall. This project demonstrates the feasibility of a lightweight LSTM for ATC slot-filling, and the need for large quantities of data to get effective results for transformer-based readback verification.

Introduction

Accurate interpretation of air traffic control utterances is crucial for ensuring aviation safety. Controllers issue instructions containing critical information—callsigns, altitudes, headings, runways—that pilots must understand and execute precisely. Misunderstanding can cascade into serious safety risks, particularly in high-traffic or complex airspace and under strenuous workloads.

There are numerous uses for a well-performing model that accurately performs slot-filling specific to ATC communications: it could serve as the language layer of an advanced ATC simulator or it could serve as the basis for a system that helps controllers track issued commands and issue warnings for unsafe instructions–to name just two examples.

Additionally, readback verification is a natural extension of ATC slot-filling. When a controller issues an instruction, in most cases, the receiving pilot must readback the specifics of the instruction for verification. It is the controller's responsibility to verify that the readback was said correctly. In the real world, however, incorrect readbacks are often missed, or misunderstood. A system that leverages modern NLP methods and does real time verification of pilot readbacks would undoubtedly increase airspace safety.

In this project, there are two goals: to label individual ATC utterances, and to classify two utterances by their readback relevance and correctness. As a constraint to these goals, we'll also see if we can accomplish this using relatively lightweight models: LSTM and miniLM. Although miniLM is a transformer model, it is designed as a particularly lightweight transformer with only 33 million parameters.

Related Work

Related work in ATC communications has mostly centered on the adaption of pre-trained models to process ATC text and perform speech recognition. Zuluaga-Gomez et al. (2022) introduced the ATCO2 Corpus, a multi-hour, annotated dataset designed to support both automatic speech recognition and natural language understanding of ATC dialogues–it is the dataset used in this project. Although my data was manually annotated, many NLP projects pertaining to ATC use the annotations provided by this dataset. Zuluaga-Gomez et al. (2022) also proposed BERTraffic, a BERT-based model for jointly detecting speaker roles and speaker turns in ATC speech. Deng et al. (2023) addressed the challenges of parsing ATC speech by introducing BERT-IRSF, an end-to-end BERT-based joint intent recognition and slot-filling model. Other efforts include an LSTM-RNN readback consistency checker for Chinese ATC readbacks (Jia et al., 2017), a real-time CNN-RNN speech recognition and controller instruction understanding pipeline (Lin et al., 2019), and a Bayesian-network ASR/text-analysis framework that correlates pilot readback errors with loss-of-separation risk (Sun & Tang, 2021). This project has similar elements to these works, although uses the novel constraint of opting for lighter-weight neural network models, like miniLM rather than BERT.

Architecture

This project had two distinct, but related, goals, and two respective approaches:

Goal 1: Tag every token in an ATC utterance with a semantic label (e.g. B-ALTITUDE, I-CALLSIGN, O).

Approach: I trained a bidirectional LSTM slot-filling model that tags each token in an utterance with labels. Inputs—mapped from token IDs to embeddings—are fed through a packed, bidirectional LSTM so that padding is ignored. The final model used a vocabulary size of 526 tokens and 42 labels, listed below. Each token was mapped to a 100-dimension embedding, then run through a two-layer (forward and backward) LSTM with 128 hidden units per direction. The backward and forward hidden layers were concatenated and passed through a linear classifier to produce BIO tag logits.

```
['uwb-atcc_twr-7sxor3_000114_000686_at:', 'donavia', 'three', 'zero', 'five', 'runway', 'three', 'one', 'clear', 'to', 'land', 'wind', 'three', 'three',
'zero', 'degrees', 'and', 'three', 'knots']

['O', 'B-CALLSIGN', 'I-CALLSIGN', 'I-CALLSIGN', 'I-CALLSIGN', 'B-RUNWAY', 'I-RUNWAY', 'I-RUNWAY', 'B-CLEARANCE', 'I-CLEARANCE', 'I-CLEARANCE', 'B-WIND',
'I-WIND', 'I-WIND', 'I-WIND', 'I-WIND', 'I-WIND', 'I-WIND', 'I-WIND']
```

Figure 1: Sample of one example labeled

For post-training prediction and to serve the demonstration CLI, the logits are then passed through a softmax classifier to normalize the class probabilities.

```
Labels: ['B-ALTITUDE', 'B-APPROACH', 'B-CALLSIGN', 'B-CLEARANCE', 'B-CONDITION', 'B-FACILITY', 'B-FREQUENCY', 'B-HEADING', 'B-INSTRUCTION', 'B-REQUEST',
'B-ROUTE', 'B-RUNWAY', 'B-SPEED', 'B-SQUAWK', 'B-STATUS', 'B-TAXIWAY', 'B-TEMPORAL_MODIFIER', 'B-TRAFFIC', 'B-TRIGGER', 'B-WAYPOINT', 'B-WIND', 'I-ALTITU
DE', 'I-APPROACH', 'I-CALLSIGN', 'I-CLEARANCE', 'I-CONDITION', 'I-FACILITY', 'I-FREQUENCY', 'I-HEADING', 'I-INSTRUCTION', 'I-REQUEST', 'I-ROUTE', 'I-RUNW
AY', 'I-SPEED', 'I-SQUAWK', 'I-STATUS', 'I-TAXIWAY', 'I-TRAFFIC', 'I-TRIGGER', 'I-WAYPOINT', 'I-WIND', 'O']
```

Figure 2: 42 labels for slot-filling

Over 12 epochs, the model achieved an average training loss of 0.0243.

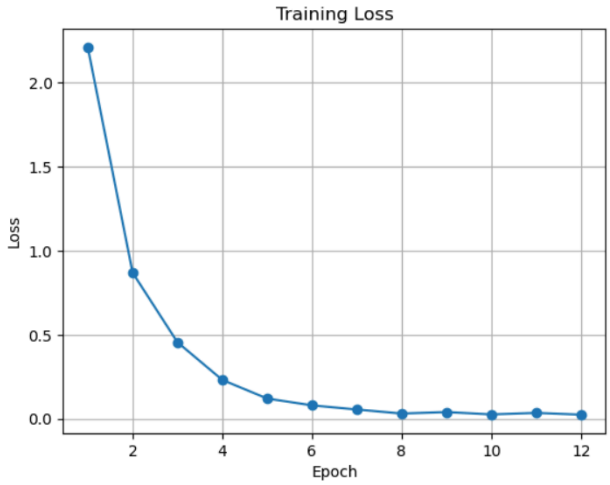


Figure 3: Average training loss over epochs

Goal 2: Given an ATC instruction s and a pilot readback r, classify their relation as one of four labels: matches_instruction, partial_readback, incorrect_readback, no_relation.

Approach: Microsoft's MiniLM (microsoft/MiniLM-L12-H384-uncased) was fine-tuned to verify pilot readbacks against controller instructions. Instruction and readback texts were concatenated with special [CLS]/[SEP] tokens, tokenized into input IDs and attention masks, and fed through the pretrained model.

The way potential readbacks were matched and encoded was of particular importance. For every instructional token – meaning, the actual instruction and the corresponding value, such as the provided heading, altitude, runway, etc. – a pair was created with those tokens identified by [ins] and [rbk] labels in the full utterance. So for example, if "descend and maintain" was identified in an ATC utterance and "descending" was identified in a pilot utterance, one pair would be created containing the full utterances to be classified. If in the same two utterances, an altitude was also identified, another pair will be created with those labels. See below for example.

```
Instruction: "delta five one three descend and maintain [ins] flight level two seven zero [ins]"
Readback   : "descending [rbk] flight level two eight zero [rbk] delta five one three"

Instruction: "delta five one three [ins] descend and maintain [ins] flight level two seven zero"
Readback   : "[rbk]descending[rbk] flight level two eight zero delta five one three"
```

Figure 4: Example of two pairs from one labeled example for reachback relationship classification

The pretrained Microsoft MiniLM-L12-H384-uncased model is a 12-layer transformer with 384-dimensional hidden states, 12 self-attention heads, and an approximately 30,000 word

vocabulary. Over 25 epochs of training, the average loss fell steadily from 1.1973 to 0.1001 at epoch 25, dropping below 0.5 by Epoch 10 and reaching ~0.1 by the end.

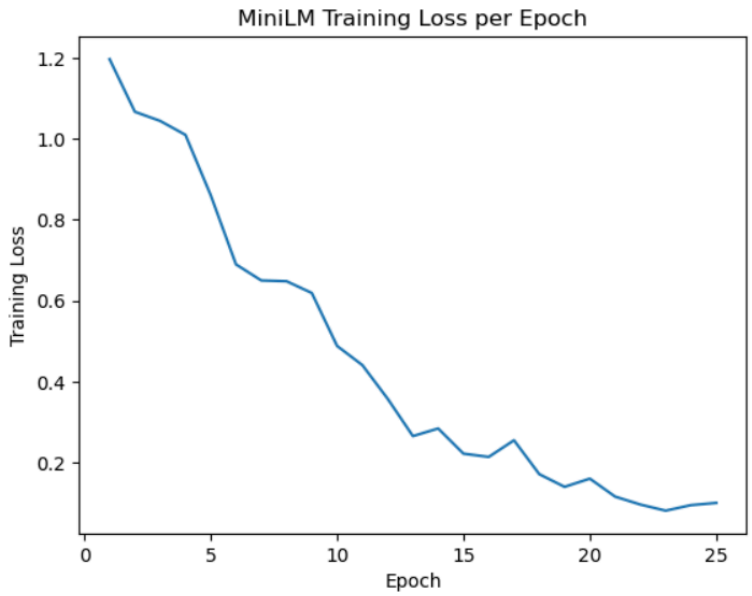


Figure 5: MiniLM training loss per epoch

The data I used was extracted from the UWB-ATCC Corpus, a manually-transcribed collection of 20 hours of ATC recordings from European airspace. From the full dataset, I extracted 324 individual utterances across all three levels of ATC facilities (tower, terminal and center) and supplemented them with 43 examples of synthetic utterances created by myself to add North American phrases and underrepresented labels. Collections of these utterances were also used to create 25 dialogues from which the readback relationship data was extracted. Potential readback pairs were extracted from these dialogues and combined with negative examples at a 1:1 ratio. These negative examples have no relation to each other and are tagged “no_relation”.

Testing & Experimentation

Both parts of the pipeline—slot-filling and readback classification—were evaluated on held-out test sets, and the LSTM for utterance labeling was compared against a simple baseline of assuming all tokens were assigned the most common label.
Overall, the LSTM achieved an accuracy of 81% with a weighted-average F1 of 0.77. The accuracy assigning the most common label (“I-CALLSIGN”) to all tokens was calculated to be 0.23 – our model accuracy greatly exceeds this indicating at least some base success. With an F1 over 0.80, it did particularly well on common slots like ALTITUDE, CALLSIGN, INSTRUCTION, HEADING, and RUNWAY. However, its macro-average F1 of 0.68 shows that it struggled on low-support labels (e.g. REQUEST, SQUAWK, STATUS, and the “O” class), where precision and recall often fall to zero. The full charted evaluation results are available at the end of this report.

As part of a qualitative testing, I ran several varying types of utterances through the LSTM model. I found that the model often mislabels pilot readback tokens when pilots shorten or respond in less formal phrases than ATC usually uses. For example, in the utterance below, “five thousand” should be recognized as “B-ALTITUDE”. It seems the model was confused by the lack of a readback instruction preceding the altitude and assumed numbers at the start of the utterance were a callsign.

```
Input: five thousand for delta one two one
  five       → I-CALLSIGN      (62.1%)
  thousand   → I-ALTITUDE      (49.4%)
  for        → B-STATUS        (39.6%)
  delta      → B-CALLSIGN      (97.7%)
  one        → I-CALLSIGN      (93.3%)
  two        → I-CALLSIGN      (99.8%)
  one        → I-CALLSIGN      (99.1%)
```

And in cases where pilots readback very quick and colloquial language, the model has great difficulty differentiating the difference between numbers:

```
Input: one six for two five
  one        → I-CALLSIGN      (38.0%)
  six        → I-CALLSIGN      (40.2%)
  for        → B-CALLSIGN      (37.0%)
  two        → I-CALLSIGN      (92.9%)
  five       → I-CALLSIGN      (98.5%)
```

The LSTM also shows overconfidence in high-frequency tags (CALLSIGN, INSTRUCTION), sometimes at the expense of correctly tagging less frequent slots like CONDITION or SPEED. Numeric sequences present another challenge: multi-digit altitudes (e.g., “three five zero”) occasionally get broken into unpredictable subwords and mis-tagged when adjacent digits have some ambiguity. In each of the examples below, a number is mislabeled at either the end (“nine” on the right) or at the end (“five” at the beginning).

```
  takeoff    → I-CLEARANCE     (99.0%)
  winds      → B-WIND          (98.2%)
  zero       → I-WIND          (81.0%)
  five       → I-WIND          (82.9%)
  zero       → I-WIND          (75.1%)
  at         → I-WIND          (92.5%)
  nine       → I-CALLSIGN      (55.8%)
```

```
Input: five thousand for delta one two one
  five       → I-CALLSIGN      (50.0%)
  thousand   → I-ALTITUDE      (87.0%)
  for        → B-STATUS        (44.0%)
  delta      → B-CALLSIGN      (99.4%)
  one        → I-CALLSIGN      (99.8%)
  two        → I-CALLSIGN      (100.0%)
  one        → I-CALLSIGN      (99.9%)
```

In the case of the miniLM readback verification model, the results were less encouraging. At epoch 25 (avg loss 0.1001), the relation classifier achieves 57.5 % accuracy with a weighted-average F1 of 0.61, but a much lower macro-average F1 of 0.46—showing it does well on the dominant no_relation class (F1 0.73) yet struggles on partial_readback (F1 0.29) and incorrect_readback (F1 0.35). incorrect_readback and partial_readback are critical classes

in a model that should be able to differentiate the difference between a valid and incorrect readback. In qualitative testing, the model failed to tell the difference between valid and incorrect readbacks for several simple examples. For example, the first of these below should be predicted to be incorrect. The second should be classified as correct.

```
Instruction: “delta five one three descend and maintain [ins] flight level two seven zero [ins]”
Readback   : “descending [rbk] flight level two eight zero [rbk] delta five one three”

Predicted Relation: partial_readback  (88.5% confidence)

matches_instruction :   0.4%
partial_readback    :  88.5%
incorrect_readback  :   8.0%
no_relation         :   3.1%
('partial_readback', tensor([0.0037, 0.8854, 0.0802, 0.0307]))

Instruction: “southwest five two two descend and maintain [ins] flight level two seven zero [ins]”
Readback   : “descending [rbk] flight level two seven zero [rbk] southwest five two two”

Predicted Relation: partial_readback  (71.8% confidence)

matches_instruction :   1.3%
partial_readback    :  71.8%
incorrect_readback  :   6.3%
no_relation         :  20.6%
('partial_readback', tensor([0.0130, 0.7178, 0.0628, 0.2064]))
```

Based on the amount of data used for training in similar projects, these results would suggest that the model simply does not have enough training for the transformer to learn from – especially when it needs to learn the subtle differences between correct and incorrect readbacks. For example, Chen et al. (2019) used 4478 examples for training in a BERT-based transformer model trained on slot-filling as opposed to 185.

Conclusion

The results demonstrate that, for labeling at least, we can get good results from a relatively light-weight model. Based on these results, I would like to continue to add data – I would expect that doing so would increase accuracy even more, and raise the F1 scores for the least-represented labels. Despite the relatively disappointing results from the readback verification classification, there are a number of things I would like to try in the future, beyond just adding more data (although I do think the lack of annotated data was a major problem with that part of the project). For starters, limiting the classes to only 2: matches_instruction and incorrect_readback might focus the model, and prevent too much of spread across many different classes. Also, reducing the pairs themselves to just the semantically labelled parts might also allow for the model to better identify the small details of incorrect readbacks. And finally, creating more instances of incorrect_readbacks might prevent the model from becoming overly confident on correct readbacks – an over-represented class within that part of the project.

References

Haque, I. (2020). A simple overview of RNN, LSTM and attention mechanism. The Startup. Medium. Retrieved from https://medium.com/swlh/a-simple-overview-of-rnn-lstm-and-attention-mechanism-9e844763d07b

Tang, Y. (n.d.). Build and train a PyTorch LSTM in under 100 lines of code [Video]. YouTube. https://www.youtube.com/watch?v=J_ksCv_r_rU

Maurya, M. (2021). Name entity recognition and various tagging schemes. Medium. Retrieved from https://medium.com/@muskaan.maurya06/name-entity-recognition-and-various-tagging-schemes-533f2ac99f52

Maheshkar, S. (2021). An introduction to Datasets and DataLoader in PyTorch. Weights & Biases. Retrieved from https://wandb.ai/sauravmaheshkar/Dataset-DataLoader/reports/An-Introduction-to-Datasets-and-DataLoader-in-PyTorch--VmlldzoxMDI5MTY2

Deng, Q., Yang, Y., Zhang, X., Qian, S., Zhang, M., & Cai, K. (2023). A BERT-based intent recognition and slot filling joint model for air traffic control instruction understanding. In Proceedings of the IEEE (Conference/Journal name) (pp. 1–9)

Chen, Q., Zhuo, Z., & Wang, W. (2019). BERT for joint intent classification and slot filling [Preprint]. arXiv. https://arxiv.org/abs/1902.10909

Microsoft. (n.d.). Microsoft/MiniLM-L12-H384-uncased [Hugging Face model]. Hugging Face. Retrieved May 3, 2025, from https://huggingface.co/microsoft/MiniLM-L12-H384-uncased

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers [Preprint]. CoRR, abs/2002.10957. arXiv:2002.10957.

Jia, G., Lu, Y., Lu, W., Shi, Y., & Yang, J. (2017). A novel verification method for Chinese aviation radiotelephony readbacks based on LSTM-RNN. Electronics Letters, 53(6), 401–403. https://doi.org/10.1049/el.2016.2877

Lin, Y., Tan, X., Yang, B., Yang, K., Zhang, J., & Yu, J. (2019). Real-time controlling dynamics sensing in air traffic system. Sensors, 19(3), 679. https://doi.org/10.3390/s19030679

Sun, Z., & Tang, P. (2021). Automatic communication error detection using speech recognition and linguistic analysis for proactive control of loss of separation. Transportation Research Record: Journal of the Transportation Research Board, 2675(5), 1–12

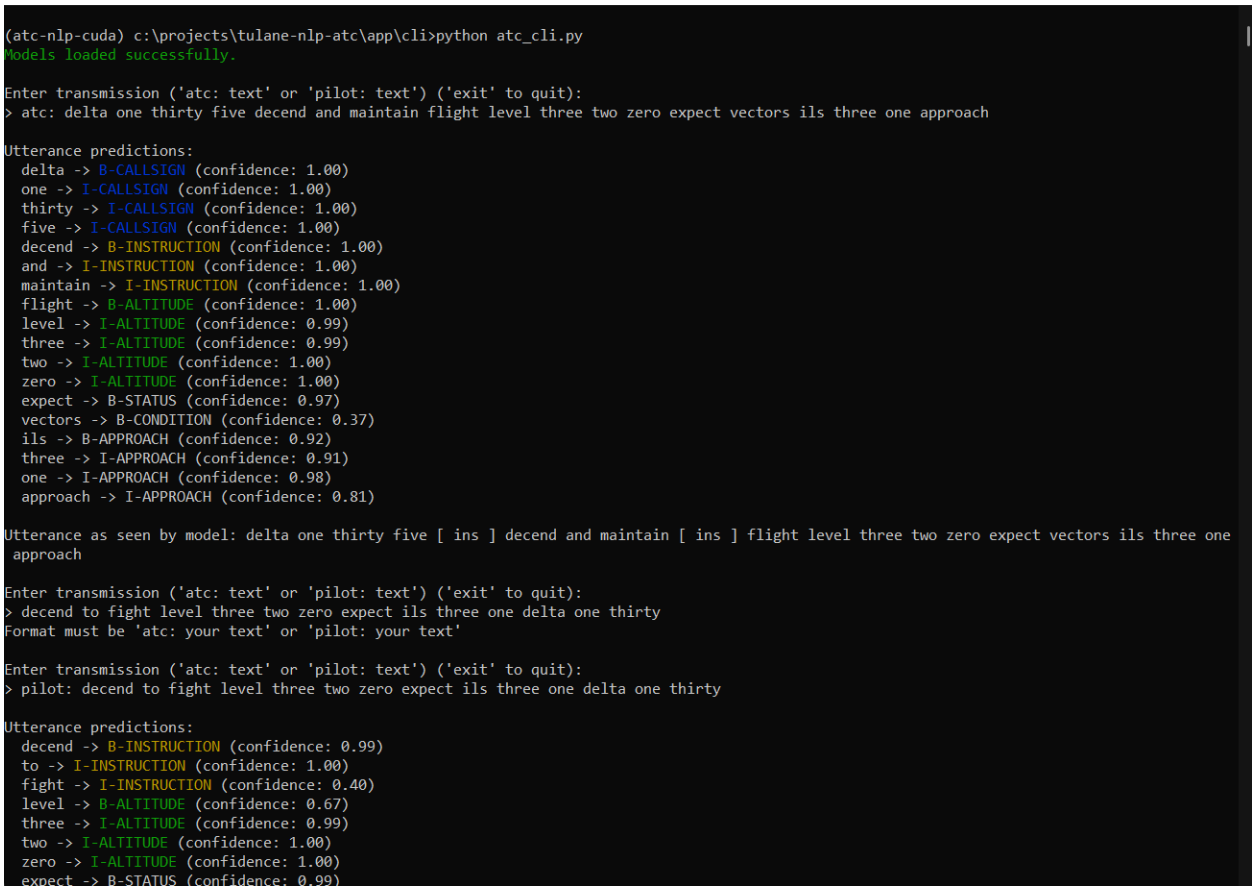
Zuluaga-Gomez, J., Veselý, K., Szőke, I., Motlicek, P., Prasad, A., Sarfjoo, S. S., Nigmatulina, I., Cevenini, C., Kolčárek, P., Tart, A., Černocký, J., & Klakow, D. (2022). ATCO2 corpus: A large-scale dataset for research on automatic speech recognition and natural language understanding of air traffic control communications. arXiv. https://arxiv.org/abs/2211.04054

Zuluaga-Gomez, J., Prasad, A., Nigmatulina, I., Sarfjoo, S. S., … & Klakow, D. (2022). How does pre-trained Wav2Vec2.0 perform on domain-shifted ASR? An extensive benchmark on air traffic control communications. In IEEE Spoken Language Technology Workshop (SLT), Doha, Qatar.

Zuluaga-Gomez, J., Sarfjoo, S. S., Prasad, A., … & Klakow, D. (2022). BERTraffic: BERT-based joint speaker role and speaker change detection for air traffic control communications. In IEEE Spoken Language Technology Workshop (SLT), Doha, Qatar.

Šmídl, L., Švec, J., Tihelka, D., Matoušek, J., Romportl, J., & Ircing, P. (2019). Air traffic control communication (ATCC) speech corpora and their use for ASR and TTS development. Language Resources and Evaluation, 53(3), 449–464. https://doi.org/10.1007/s10579-018-9425-8

Screenshot of Demo CLI:



Slot-filling LSTM full evaluation results:

Slot Report:

	precision	recall	f1-score	support
B-ALTITUDE	0.70	0.88	0.78	8
B-APPROACH	0.40	1.00	0.57	2
B-CALLSIGN	0.68	0.79	0.73	34
B-CLEARANCE	0.89	1.00	0.94	8
B-CONDITION	0.75	0.60	0.67	5
B-FACILITY	0.67	1.00	0.80	6
B-FREQUENCY	1.00	1.00	1.00	3
B-HEADING	0.71	1.00	0.83	5
B-INSTRUCTION	0.78	0.93	0.85	27
B-REQUEST	0.00	0.00	0.00	1
B-ROUTE	1.00	1.00	1.00	1
B-RUNWAY	1.00	0.78	0.88	9
B-SPEED	1.00	0.33	0.50	3
B-SQUAWK	0.00	0.00	0.00	0
B-STATUS	0.00	0.00	0.00	0
B-TAXIWAY	0.67	1.00	0.80	2
B-TEMPORAL_MODIFIER	1.00	1.00	1.00	3
B-TRAFFIC	1.00	1.00	1.00	1
B-TRIGGER	1.00	1.00	1.00	3
B-WAYPOINT	0.67	0.57	0.62	7
B-WIND	1.00	1.00	1.00	3
I-ALTITUDE	0.90	1.00	0.95	26
I-APPROACH	0.60	1.00	0.75	3
I-CALLSIGN	0.84	0.89	0.86	100
I-CLEARANCE	0.81	0.93	0.87	14
I-CONDITION	1.00	0.57	0.73	7
I-FACILITY	0.75	1.00	0.86	3
I-FREQUENCY	0.48	1.00	0.65	13
I-HEADING	0.94	1.00	0.97	15
I-INSTRUCTION	0.79	0.86	0.83	22
I-REQUEST	0.00	0.00	0.00	2
I-ROUTE	1.00	1.00	1.00	3
I-RUNWAY	0.93	0.87	0.90	15
I-SPEED	0.88	0.78	0.82	9
I-SQUAWK	0.00	0.00	0.00	0
I-STATUS	0.00	0.00	0.00	0
I-TAXIWAY	0.00	0.00	0.00	0
I-TRAFFIC	1.00	1.00	1.00	3
I-TRIGGER	1.00	1.00	1.00	3
I-WAYPOINT	0.00	0.00	0.00	0
I-WIND	0.95	1.00	0.97	18
O	0.00	0.00	0.00	43
accuracy			0.80	430
macro avg	0.66	0.71	0.67	430
weighted avg	0.74	0.80	0.76	430

Average loss per epoch and evaluation summary for readback relationship classifier:

```
Epoch 1 — Avg Training Loss: 1.1973
Epoch 2 — Avg Training Loss: 1.0672
Epoch 3 — Avg Training Loss: 1.0447
Epoch 4 — Avg Training Loss: 1.0102
Epoch 5 — Avg Training Loss: 0.8610
Epoch 6 — Avg Training Loss: 0.6899
Epoch 7 — Avg Training Loss: 0.6502
Epoch 8 — Avg Training Loss: 0.6483
Epoch 9 — Avg Training Loss: 0.6192
Epoch 10 — Avg Training Loss: 0.4883
Epoch 11 — Avg Training Loss: 0.4408
Epoch 12 — Avg Training Loss: 0.3577
Epoch 13 — Avg Training Loss: 0.2653
Epoch 14 — Avg Training Loss: 0.2841
Epoch 15 — Avg Training Loss: 0.2219
Epoch 16 — Avg Training Loss: 0.2138
Epoch 17 — Avg Training Loss: 0.2550
Epoch 18 — Avg Training Loss: 0.1712
Epoch 19 — Avg Training Loss: 0.1398
Epoch 20 — Avg Training Loss: 0.1601
Epoch 21 — Avg Training Loss: 0.1157
Epoch 22 — Avg Training Loss: 0.0960
Epoch 23 — Avg Training Loss: 0.0809
Epoch 24 — Avg Training Loss: 0.0946
Epoch 25 — Avg Training Loss: 0.1001
                    precision    recall  f1-score   support

matches_instruction     0.5556    0.4167    0.4762        12
   partial_readback     0.1667    1.0000    0.2857         1
 incorrect_readback     0.2727    0.5000    0.3529         6
        no_relation     0.8571    0.6429    0.7347        28

           accuracy                         0.5745        47
          macro avg     0.4630    0.6399    0.4624        47
       weighted avg     0.6908    0.5745    0.6104        47
```