Hybrid Detection Method for Multi-Intent Recognition in Air–Ground Communication Text

Weijun Pan 1, Zixuan Wang 2,*, Zhuang Wang 2, Yidi Wang 2 and Yuanjing Huang 3

1 Flight Technology and Flight Safety Research Base of the Civil Aviation Administration of China, Civil Aviation Flight University of China, Guanghan 618307, China; wjpan@cafuc.edu.cn
2 College of Air Traffic Management, Civil Aviation Flight University of China, Guanghan 618307, China; wangzhuang@cafuc.edu.cn (Z.W.); wyd1192254593@163.com (Y.W.)
3 Air Traffic Management Center, Civil Aviation Flight University of China, Guanghan 618307, China; hyj@cafuc.edu.cn
* Correspondence: fannyw821@163.com

Abstract: In recent years, the civil aviation industry has actively promoted the automation and intelligence of control processes with the increasing use of various artificial intelligence technologies. Air–ground communication, as the primary means of interaction between controllers and pilots, typically involves one or more intents. Recognizing multiple intents within air–ground communication texts is a critical step in automating and advancing the control process intelligently. Therefore, this study proposes a hybrid detection method for multi-intent recognition in air–ground communication text. This method improves recognition accuracy by using different models for single-intent texts and multi-intent texts. First, the air–ground communication text is divided into two categories using multi-intent detection technology: single-intent text and multi-intent text. Next, for single-intent text, the Enhanced Representation through Knowledge Integration (ERNIE) 3.0 model is used for recognition; while the A Lite Bidirectional Encoder Representations from Transformers (ALBERT)_Sequence-to-Sequence_Attention (ASA) model is proposed for identifying multi-intent texts. Finally, combining the recognition results from the two models yields the final result. Experimental results demonstrate that using the ASA model for multi-intent text recognition achieved an accuracy rate of 97.84%, which is 0.34% higher than the baseline ALBERT model and 0.15% to 0.87% higher than other improved models based on ALBERT and ERNIE 3.0. The single-intent recognition model achieved an accuracy of 96.23% when recognizing single-intent texts, which is at least 2.18% higher than the multi-intent recognition model. The results indicate that employing different models for various types of texts can substantially enhance recognition accuracy.

Keywords: deep learning; multi-intent recognition; multi-intent detection; dependency parsing; air traffic control

1. Introduction

With the rapid development of the information age, textual data has found widespread applications in various fields, including the aviation field. In modern aviation, air–ground communication is the primary mode of communication between controllers and pilots, with a direct impact on aviation safety and efficiency. However, the frequent occurrence of flight accidents and incidents in aviation history caused by non-standard air–ground communication phrases serves as a profound warning, emphasizing the importance of accuracy and standardization in communication language. Therefore, conducting an in-depth analysis of air–ground communication texts is particularly necessary. Air–ground communication encompasses not only instructions from controllers to pilots but also pilots’ readbacks and requests. In texts of air–ground communication, the task of intent recognition typically involves deducing the requirements and purposes of control instructions to assist relevant parties in understanding their content. Moreover, intent recognition can play...
a crucial role in applications such as verifying control instruction repetitions and training flight simulator captains. Unlike typical text intent recognition tasks, air–ground communication texts often involve one or more intents. However, due to special circumstances such as communication interruptions, there may also be texts without a discernible intent. Therefore, this study filters out texts without intent and concentrates solely on those with clear intents. Additionally, given the stringent safety standards in the civil aviation field, the accuracy of intent recognition is of paramount importance. However, single-intent recognition models cannot meet the requirements for identifying multiple intents in air–ground communication texts. Existing multi-intent recognition models, while capable of identifying one or more intents, often do not perform well. To address the challenge of recognizing multiple intents in air–ground communication texts, this paper proposes a hybrid detection method tailored for this context. This method first determines whether multiple intents exist in the text and then uses different models to recognize single-intent and multi-intent texts, thereby enhancing recognition accuracy.

This paper aims to propose a multi-intent recognition method tailored specifically for air–ground communication texts, capable of accurately identifying multiple intents within such communications, as well as novel insights and technological support for future research and applications in related fields. Moreover, this study provides useful references and insights into the application of multi-intent recognition in other domains. The remaining sections of this paper are as follows: Section 2 introduces the current research status of intent recognition both domestically and internationally, as well as relevant studies in civil aviation, providing theoretical foundations and background knowledge for future research. Section 3 discusses multi-intent recognition technology, including the multi-intent detection strategy and related technological research used in this paper, as well as relevant models for intent recognition, such as single-intent and multi-intent recognition models. Section 4 investigates the relevant features of air–ground communication texts and datasets used for intent recognition, compares the performance of various models and strategies in this field, and draws experimental conclusions. Finally, Section 5 summarizes the main findings of this paper, provides prospects for future work, and identifies possible research directions and issues for further investigation.

2. Related Work

Intent recognition is a text classification task primarily involving the categorization of different intentions within the text for intent recognition. Conventional intent recognition methods can be divided into two categories: rule-based semantic recognition methods and classification algorithms that use statistical features. The rule-based template method requires manual construction of rule templates and category information to classify user intent texts [1]. For example, Ramanand et al. proposed a rule-based and graph-based method for consumer intent recognition [2], which achieved satisfactory classification results in a single domain. Li et al. discovered that different expressions within the same domain can result in an increase in the number of rule templates, increasing costs in terms of manpower and time [3]. Therefore, although rule-based template matching methods do not require a large amount of training data, the cost of reconstructing templates becomes significantly higher when there are changes in the categories of intent texts. In contrast, statistical feature classification methods necessitate the extraction of key features from textual corpora, such as character, word features, N-Gram, etc., followed by intent classification through classifier training. Commonly used methods include Naive Bayes [4], Adaboost [5], support vector machine (SVM) [6], logistic regression [7], etc. However, these methods typically rely on empirically determined features, which pose some issues such as heavy reliance on dataset size, sparse feature vector extraction, and the inability of extracted feature vectors to effectively represent the semantic information of short texts. With continuous breakthroughs in technology, classification methods based on deep learning are emerging as more effective approaches to address these issues. Word embeddings, convolutional neural networks (CNN), recurrent neural networks (RNN), and variants
such as long short-term memory (LSTM) networks are examples of these methods. Kim et al. used word embeddings as lexical features for intent classification [8]. In contrast to conventional bag-of-words models, intent classification methods based on word embeddings have higher representational power and domain scalability. Additionally, Firdaus et al. proposed a joint model that combines CNN and LSTM [9], introducing advanced deep learning techniques for intent recognition. With further study, existing pre-trained word embedding methods such as Glove [10] and Word2Vec [11] have been specifically trained on large corpora to generate unlabeled word vectors that can be applied to a variety of models. Kim et al. used rich word embeddings as input to bidirectional long short-term memory (BiLSTM) for intent recognition [12]. However, as the parameters of deep learning networks increase, each category necessitates a large amount of data to fully exploit the potential of neural networks in known samples. To address the small sample problem, Xia et al. proposed a self-attention-based intent recognition capsule neural network [13], but this method is only suitable for systems with a single intent. Subsequently, Srivastava et al. proposed a hierarchical bidirectional encoder representations from transformers (BERT) architecture for intent detection in utterances [14], which produced promising results.

There are two approaches to addressing the problem of multi-intent recognition. The first is problem transformation, which involves increasing categories by merging multiple intents into a new category and then solving it with existing classification algorithms. This method is data-driven, but it inevitably increases the number of labels, necessitating larger datasets and increasing algorithm complexity. The second approach involves improving algorithms to adapt to multi-intent recognition tasks, such as combining weakly supervised learning and CNN to address multi-label classification problems in images [15], or combining CNN and RNN to solve multi-label classification problems in text [16]. Kim et al. proposed a multi-intent recognition system using training data labeled with a single intent [17]. The study divides sentences into three types: single-intent statements, multi-intent statements with conjunctions, and multi-intent statements without conjunctions, and then uses a two-stage approach to recognize multiple intents. Yang et al. analyzed user intent texts using dependency parsing (DP) to determine if they contain multiple intents. They utilized term frequency-inverse document frequency (TF-IDF) and pre-trained word embeddings to calculate matrix distances to determine the number of intents in a sentence. Then, by combining syntactic features with CNN for intent classification, they were able to discern multiple user intents [18].

In the field of civil aviation, several early scholars employed natural language processing (NLP) methods to classify relevant texts with the goal of improving safety and efficiency. Rose et al. utilized a bag-of-words model, TF-IDF, and k-means clustering algorithms to cluster and analyze texts from the Aviation Safety Reporting System (ASRS) [19]. Their objective was to uncover trends that existing anomaly labels failed to reveal, thereby providing a more automated and refined framework for analyzing aviation safety text data. Madeira et al. implemented TF-IDF, Word2Vec, and the label spreading (LS) algorithm to predict human factors in aviation accident reports [20]. Their aim was to develop an intelligent prediction system capable of identifying and classifying human factors to enhance aviation safety management. Miyamoto et al. applied TF-IDF and k-means algorithms to classify and cluster text data from the ASRS, identifying inefficient operational patterns in aviation [21], thereby improving flight safety and efficiency. These scholars selected traditional classification methods for text classification due to their speed and low resource requirements. However, these methods are limited in capturing the contextual information of words and have drawbacks, such as sparse features, requiring extensive manual feature selection and parameter tuning. With the advancement of technology, deep learning-based classification methods have addressed these issues. Ray et al. proposed and evaluated the aeroBERT-Classifier [22], a system utilizing the BERT model to classify aviation demands, aiming to develop a model capable of accurately classifying aviation demands. This research, based on deep learning classification, achieved superior results but was limited by its inability to perform multi-label classification, meaning it could not recognize
multiple intents simultaneously. Nowadays, scholars have begun to study air traffic control (ATC) instructions. Kleinert et al. proposed an assistant-based speech recognition (ABSR) system [23], which significantly reduces the workload of controllers, improves safety and overall performance, and incorporates technologies related to instruction understanding. Lin et al. provided a comprehensive review of speech instruction understanding (SIU) in the ATC domain, addressing the challenges, technologies, and applications [24]. They emphasized that intent recognition is a key component of the language understanding module. At the same time, an increasing number of scholars are focusing on the research of automatic speech recognition and understanding (ASRU). Ahrenhold et al. applied ASRU technology for the validation of pre-filled radar tags [25], aiming to enhance safety and reduce the workload of air traffic controllers. Chen et al. explored the development of ASRU technology in the field of air traffic management (ATM) and its role in transatlantic research collaboration [26]. Zuluaga-Gomez et al. summarized the experiences of the ATCO2 project regarding ASRU in ATC communications [27]. Within the ASRU framework, intent recognition is a crucial component. It involves not only converting speech signals into text but, more importantly, understanding the meaning and intent behind the text to respond correctly. Although intent recognition in air–ground communication is just a subset of broader research, it remains a significant component. For instance, in tasks involving intent recognition and slot filling, intent recognition forms the foundation. Errors in intent recognition can lead to inaccuracies in slot filling. Therefore, the accuracy of intent recognition is critical to the overall research. Conducting dedicated studies on intent recognition is essential for improving its accuracy. Pan et al. constructed a multi-intent recognition model named ERNIE-Gram_BiGRU_Attention (EBA) for ATC [28]. The study adopted the approach of transforming the problem by merging multiple categories into a new category to simplify the classification task. This method addresses the issue of multiple intent recognition in air–ground communication but leads to an increase in the number of labels, which in turn increases computational complexity. An excessive number of labels can also affect the model’s generalization capability. Merging categories may cause the model to confuse different intents, thereby reducing the accuracy of intent recognition. Additionally, if additional intent categories are needed later, maintaining the system becomes challenging, resulting in reduced scalability of the dataset. Therefore, in multi-intent recognition tasks, maintaining the independence of categories not only helps improve model performance and accuracy but also enhances the system’s flexibility and scalability.

In conclusion, due to the issue of multiple intents in air–ground communication, traditional classification methods and deep learning-based classification methods fall short of meeting the requirements. The approach of merging multiple categories also has several drawbacks. Although research in other fields has proposed improved algorithms to address the problem of multiple intents, these methods do not perform well in recognizing single intents. Therefore, this study proposes a hybrid intent recognition method based on multi-intent detection to address the aforementioned issues.

3. Methodology

This study proposes a hybrid detection method for multi-intent recognition in air–ground communication text. First, multi-intent detection technology is used to determine if the air–ground communication text contains multiple intents. If the text includes multiple intents, the multi-intent recognition model ASA is used; otherwise, the single-intent recognition model ERNIE 3.0 is used. The ERNIE 3.0 model is trained using the single-intent text dataset, while the ASA model is trained using the multi-intent text dataset. Due to the relative scarcity of multi-intent text data, the single-intent text dataset is used as a supplement. The specific process is illustrated in Figure 1.
3.1. Multi-Intent Detection

This study employs DP for multi-intent detection. DP extracts syntactic features from air–ground communication text by analyzing the dependency relationships between different sentence constituents. It uses coordination relationships to determine whether or not a sentence contains multiple intents. In essence, DP identifies grammatical elements, such as “subject-verb-object” and “adverbial complement” and analyzes their relationships. This method considers the verb to be the core element in a sentence, with other components having direct or indirect connections to it.

In the field of ATC, there are strict requirements for the communication structure of air–ground communication. In the initial contact between the controller and the pilot, i.e., the first dialogue turn, which includes statements from both parties, the communication should follow this structure: “recipient’s call sign + sender’s call sign + communication content”. After the initial contact, in each subsequent communication, the pilot must still adhere to the structure established during the first contact. The controller, however, may omit their own call sign and use the structure “recipient’s call sign + communication content”. This study, in accordance with the structure requirements of air–ground communication, focuses on the communication content in air–ground communication texts.

In multi-intent detection tasks, this study should pay special attention to whether air–ground communication texts contain coordinate relationships (COO). When the sentence structure contains COO, it indicates that the sentence contains multiple entities or actions, which implies multiple intents. For example, consider the following air–ground communication text: “CCA8891, Approach radar contact, climb to standard flight level 36”. Here, the recipient’s call sign is “CCA8891”, and the communication content is “approach radar contact, climb to standard flight level 36”. In this context, “approach radar contact” indicates that the controller informs the pilot that they have been identified by the radar, forming a subject–verb relationship, where “approach radar” is the subject and “contact” is the verb. “climb to standard flight level 36” represents a verb–object relationship, with “climb to” being another parallel verb, “standard flight level” as the object, and “36” as a numeric modifying “standard flight level”. Analyzing the communication content of this text reveals the presence of coordinate relationships in the sentence, indicating the concurrent occurrence of multiple actions and, thus, multiple intents. Figure 2 depicts the DP structure for this example, and Table 1 details the relationships within it.
After determining the dependency relationships of air–ground communication texts, $S_{DP} = \{dp_i\} (i = 1, 2, \ldots)$ is used to represent the dependency relationship set of air–ground communication text $S$, and $m_S$ is used to indicate whether the air–ground communication text contains multiple intents. The calculation formula is as follows:

$$m_S = \begin{cases} 
1, & \text{COO} \in S_{DP} \\
0, & \text{COO} \notin S_{DP} 
\end{cases}$$

(1)

### 3.2. Single-Intent Recognition Model

Single intent recognition requires selecting the correct category from multiple possible intent categories, making it a typical multi-class classification problem. In this study, the ERNIE 3.0 model is used to classify single intents in air–ground communication texts. ERNIE 3.0 [29] is a large pre-trained language model in the ERNIE series proposed by Baidu. These models are based on the transformer architecture and have been pre-trained on large-scale corpora, resulting in a strong semantic understanding and representation learning capabilities. Conventional large-scale pre-trained language models have demonstrated relatively poor performance in downstream language understanding tasks. To address this issue, the ERNIE 3.0 model integrates the advantages of autoregressive networks and autoencoding networks. This allows trained models to quickly adapt to zero-shot, few-shot, or fine-tuning scenarios in natural language understanding and text generation tasks. Additionally, the ERNIE 3.0 model incorporates knowledge graph data during the pre-training phase. The architecture of the model is illustrated in Figure 3 [29].

![ERNIE 3.0 model architecture diagram.](image)

The ERNIE 3.0 model progresses from general to specific. It first establishes a general language model with large-scale text data and knowledge graphs, then continuously learns and fine-tunes to adapt to different language understanding tasks. Because of the
integration of autoregressive and autoencoding networks, ERNIE 3.0 performs well in both text generation and language understanding tasks. This study primarily focuses on language understanding tasks. The following sections will provide in-depth introductions to autoregressive and autoencoding networks.

3.2.1. Autoregressive Networks

Although the initial ERNIE model did not emphasize autoregressive properties, ERNIE 3.0 uses an autoregressive language model training task similar to the generative pre-trained transformer (GPT). This approach allows the model to predict the next word based on the preceding words, optimizing its text generation capability. Autoregressive training enables the model to use previous words when generating text, which is useful for tasks such as text generation. Autoregressive networks model a text sequence by estimating its probability distribution. In general, autoregressive networks can calculate the probability of a text sequence from left to right or from right to left. However, regardless of the direction, the modeling is unidirectional. This indicates that when predicting a word, the model cannot consider information from both sides of the word’s position. Given a text sequence of $X = \{x_1, x_2, \ldots, x_n\}$, its probability of sequence generation from left to right can be represented as follows:

$$p(x) = \prod_{t=1}^{n} p(x_t|x_{<t}) \quad (2)$$

3.2.2. Autoencoding Networks

ERNIE 3.0 adopts a pre-training mechanism similar to BERT called masked language modeling (MLM). In this mechanism, the model learns to fill in randomly masked words from the input text, forcing it to rely on context to predict missing information. This autoencoding training method allows the model to capture the bidirectional semantics of words, phrases, and entire sentences, resulting in rich language representations. Autoencoding networks work by reconstructing the original data from disrupted input text sequences. For example, the BERT model reconstructs the original sequence by predicting the masked-out words. This pre-training approach enables the model to understand and infer the masked vocabulary based on context, resulting in deep semantic representations of words, phrases, and sentences. Assuming the masked words in the sequence are denoted as $w \in W_m$ and the unmasked words are denoted as $w \in W_n$, their respective calculation probabilities are as follows:

$$p(x) = \prod_{w \in W_m} p(w|W_n) \quad (3)$$

3.3. Multi-Intent Recognition Model

Multi-intent recognition requires simultaneously identifying multiple possible intent categories, meaning a single input text may correspond to multiple intent labels, making it a multi-label classification problem. In this study, we propose an ASA model to address the multi-intent recognition problem. This model starts with the original air–ground communication text input, tokenizes it to split it into a series of token units, and then uses the ALBERT model to convert these token units into embedding vectors. Through multiple layers of transformer layers, deep semantic features of the text are extracted. Subsequently, the text features are fed into the encoder, where a series of BiLSTM layers further encode the ALBERT-output features, capturing the sequence’s contextual dependencies. The decoder’s LSTM layer processes information from the encoder, and a local attention mechanism allows the model to focus on the most relevant parts of the input sequence during output generation. The output of the decoder passes through a fully connected layer, which serves as the classifier. Finally, the output of the fully connected layer is processed using the SoftMax function to determine the probability distribution of each possible output, resulting in the label sequence. The structural diagram of the model is shown in Figure 4.
3.3.1. ALBERT

ALBERT [30] is a lightweight version of BERT, which is a pre-trained language representation model based on the Transformer architecture. ALBERT’s design provides similar language representation capabilities as BERT while significantly reducing resource consumption and improving training speed. Despite having fewer parameters, ALBERT’s performance on multiple NLP tasks is comparable to, if not superior to, that of BERT. Lan et al. [30] found that the performance of the ALBERT-xxlarge model can significantly outperform that of BERT-large, despite having only 70% of BERT-large’s parameters. The parameter comparison between ALBERT and BERT is shown in Table 2 [30].

Table 2. Comparison of parameter counts between ALBERT and BERT.

<table>
<thead>
<tr>
<th>Model</th>
<th>Size</th>
<th>Parameters</th>
<th>Layers</th>
<th>Hidden</th>
<th>Embedding</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>base</td>
<td>108 M</td>
<td>12</td>
<td>768</td>
<td>768</td>
</tr>
<tr>
<td>BERT</td>
<td>large</td>
<td>334 M</td>
<td>24</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>BERT</td>
<td>xlarge</td>
<td>1270 M</td>
<td>24</td>
<td>2048</td>
<td>2048</td>
</tr>
<tr>
<td>ALBERT</td>
<td>base</td>
<td>12 M</td>
<td>12</td>
<td>768</td>
<td>128</td>
</tr>
<tr>
<td>ALBERT</td>
<td>large</td>
<td>18 M</td>
<td>24</td>
<td>1024</td>
<td>128</td>
</tr>
<tr>
<td>ALBERT</td>
<td>xlarge</td>
<td>59 M</td>
<td>24</td>
<td>2048</td>
<td>128</td>
</tr>
<tr>
<td>ALBERT</td>
<td>xxlarge</td>
<td>233 M</td>
<td>12</td>
<td>4096</td>
<td>128</td>
</tr>
</tbody>
</table>

ALBERT reduces BERT’s parameter count using two techniques: parameter factorization of word embeddings and cross-layer parameter sharing.

Word Embedding Parameter Factorization: In the conventional BERT model, the word embedding layer maps vocabulary to a high-dimensional space (usually the size of the model’s hidden layers), requiring a large matrix with dimension of “vocab_size*hidden_size”. ALBERT employs factorization techniques to change this direct mapping approach. It first maps vocabulary to a smaller dimension (referred to as the embedding dimension), and then maps this smaller-dimensional embedding vector to the model’s hidden layer dimension. This approach decomposes the original large matrix into two smaller matrices, with dimensions “vocab_size*embedding_size” and “hidden_size*embedding_size”, respectively. Due to “embedding_size=hidden_size”, this method significantly reduces the number of parameters.

Cross-Layer Parameter Sharing: In the BERT model, each transformer layer has its own set of parameters. For example, a BERT model with 12 layers would have 12 different parameter sets. While this design increases the model’s capability, it also significantly
increases the number of parameters and computational costs. ALBERT adopts a strategy of cross-layer parameter sharing, which means that all transformer layers in the model share the same set of parameters. This not only reduces the number of model parameters but also reduces the risk of overfitting because the model encodes information with the same parameters across all layers. Additionally, this improves model training efficiency by reducing the number of parameters that need to be updated.

Through the factorization of word embedding parameters and cross-layer parameter sharing, ALBERT has successfully reduced the size of the model and the computational resources required while maintaining performance comparable to BERT. Therefore, in this study’s multi-intent recognition model, the ALBERT model is used to extract features from the text.

3.3.2. LSTM

LSTM [31] is a specialized type of RNN designed to address the limitations of standard RNNs in handling long-term dependencies. LSTM controls the flow of information through its unique structural units, which include three key “gates”: the forget gate, input gate, and output gate. These gates allow the LSTM unit to retain or forget information as needed, enabling the network to flexibly remember or forget information. Figure 5 illustrates the specific workflow of LSTM.

![Figure 5. LSTM workflow diagram.](image)

The specific workflow of LSTM is as follows:

The memory cell along with the hidden state, memorizes the historical information of the sequence data. The forget gate, $f_t$, determines which information to be deleted from the memory cell based on $h_{t-1}$ and $x_t$, as shown in the following formula:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

(4)

where $\sigma$ is the sigmoid activation function, $W_f$ represents the weight of the forget gate, $b_f$ is the bias of the forget gate, $h_{t-1}$ represents the hidden state from the previous time step, and $x_t$ is the input vector at the current time step.

The input gate, $i_t$, determines which new information will be stored in the cell state and decides which values to be updated based on $h_{t-1}$ and $x_t$. The specific formula is as follows:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

(5)

$$g_t = \tanh(W_g \cdot [h_{t-1}, x_t] + b_g)$$

(6)

where $g_t$ is the candidate memory cell used to update the memory cell, $W_i$ and $W_g$ are the weights of the input gate, and $b_i$ and $b_g$ are the biases of the input gate.

After computing the forget gate, $f_t$, and the input gate, $i_t$, the old cell state, $c_{t-1}$, is updated to a new memory cell state, $c_t$, according to the following formula:

$$c_t = f_t \circ c_{t-1} + i_t \circ g_t$$

(7)
where $\odot$ is the Hadamard product, which performs element-wise multiplication of the corresponding elements in the matrices.

The output gate $o_t$ determines which part of the cell state will be computed as the output based on $c_t$, $h_{t-1}$, and $x_t$. Then, it is passed through a tanh activation function, as expressed by the following formula:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \odot \tanh(c_t)$$

where $W_o$ is the weight of the output gate, and $b_o$ is the bias of the output gate.

3.3.3. BiLSTM

BiLSTM is a model that combines bidirectional RNNs and LSTM, with two LSTM units in each forward and backward direction. At each time step $t$, the input is provided simultaneously to the forward and backward neural networks, and the output is determined jointly by these two directions of networks. Specifically, BiLSTM can capture information from both the forward and backward directions of the text sequence at the same time, allowing the model to better understand contextual relationships and contexts.

The final output of BiLSTM is obtained by concatenating the computations of both the forward and backward LSTMs. The forward computation is performed from index 1 to $T$, while the backward computation is similar to the forward computation but with the index ranging from $T$ to 1. The specific computation formulas are as follows:

$$H = [\rightarrow h_1, \rightarrow h_2, \ldots, \rightarrow h_t, \leftarrow h_t, \ldots, \leftarrow h_1]$$

where $\rightarrow h_t$ represents the result of the forward computation, $\leftarrow h_t$ represents the result of the backward computation, and $H$ represents the final output of the BiLSTM model, which is the concatenation of the forward and backward computations.

3.3.4. LSTM with Local Attention Mechanism

The LSTM model, with a local attention mechanism, decodes only a small portion of the input sequence rather than the entire sequence. This approach reduces the computational burden and increases processing efficiency by requiring the model to handle only the most relevant information to the current output. The method dynamically determines a point of focus and creates a window around it to compute attention weights only within this window. As a result, the decoding process focuses more on crucial information, which improves performance and accuracy. The specific computational process of the model is as follows:

- Define the alignment position;

  The alignment position $p_t$ at each time step $t$ is computed based on the decoder’s current or previous time step’s hidden state $h_{t-1}$ to predict. This position indicates the center of the input sequence part where the decoder should focus its attention at the current time step. The specific calculation formula is as follows:

$$p_t = \text{Sigmoid}(W_p \cdot h_{t-1} + b_p) \times S$$

where $W_p$ is the weight parameters of the feedforward network, $b_p$ represents the bias of the feedforward network, $h_{t-1}$ is the hidden state of the LSTM decoder at the previous time step, Sigmoid is the activation function that transforms the output into values between 0 and 1, and $S$ is the total length of the input sequence, or the maximum sequence length processed by the model. This length is used to scale the calculation results of the alignment position $p_t$ to the actual range of the input sequence.

- Generating the attention window;
By generating an attention window with \( p \) as the center, a fixed-sized window, \( L \), is created to determine the local region. This window defines which parts of the input sequence will be used to compute the attention weights and context vector for the current time step.

- Calculating local attention weights:
  We compute an attention weight, \( \alpha_{t,i} \), for each encoder hidden state \( \tilde{h_i} \) relative to the current state of the decoder within the window range as follows:
  \[
  \alpha_{t,i} = \frac{\exp(e_{t,i})}{\sum_{j=p_t-L/2}^{p_t+L/2} \exp(e_{t,j})}
  \]  
  (12)
  where \( e_{t,j} = f(h_{t-1}, \tilde{h_i}) \) is a function computing the compatibility between the decoder state \( h_{t-1} \) and the encoder state \( \tilde{h_i} \), and \( \exp \) refers to the exponential function.

- Constructing the context vector:
  The context vector, \( c_t \), is the weighted average of the encoder outputs, \( \tilde{h_i} \), within the local window, with weights provided by \( \alpha_{t,i} \). The context vector contains the input information that the decoder must focus on at the current time step. The calculation formula is as follows:
  \[
  c_t = \frac{p_t+L/2}{i=p_t-L/2} \sum \alpha_{t,i} \tilde{h_i}
  \]  
  (13)

- Update the decoder state:
  We update the decoder state, \( h_t \), by combining the context vector, \( c_t \), with the previous output of the decoder.
  \[
  (h_t, c_t) = \text{LSTM}(h_{t-1}, c_{t-1}, x_t, c_t)
  \]  
  (14)
  where \( x_t \) is the output from the previous time step, and \( c_t \) is used as an additional input to guide the decoder’s attention to specific parts of the input.

- Generate the output:
  Finally, the output of the decoder \( y_t \) is generated based on \( h_t \) and \( c_t \).
  \[
  y_t = \text{softmax}(g(h_t, c_t))
  \]  
  (15)
  where \( g \) is a learnable function used to generate output from the current hidden state and the context vector.

4. Experiments
4.1. Experimental Data
4.1.1. Text Intent Description
To ensure accurate communication between air traffic controllers and pilots around the world, the International Civil Aviation Organization (ICAO) has established a set of standard regulations for air–ground communication phraseology. China has developed its own air–ground communication regulations based on ICAO requirements and the country’s specific circumstances. The scope of air–ground communication includes the taxiing phase, the takeoff and landing phase, the approach phase, and the cruising phase. In this study, the intent of air–ground communication text is classified according to different flight phases, resulting in a total of 18 text intents. The classification of these intents is based on relevant documents and guidelines from the Civil Aviation Administration of China (CAAC), combined with practical flight operation requirements, and references to a substantial amount of domestic and international research literature, as well as opinions from numerous experienced flight and air traffic control experts. It should be noted that while further refinement of the intents is possible, overly detailed classification might
lead to overfitting during model training, thus affecting the model’s generalization ability and practical application effectiveness. Therefore, this study encompasses all relevant intents within these 18 categories. This classification approach not only aids in a more comprehensive understanding and analysis of air–ground communication texts across different flight phases, enhancing the accuracy and practicality of intent recognition, but also prevents excessive model complexity, ensuring stable performance across different datasets. The detailed text intent classifications can be found in Table 3.

Table 3. Text intent classifications.

<table>
<thead>
<tr>
<th>Text Intent</th>
<th>Flight Phases</th>
<th>Text Instance</th>
<th>Text Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>departure clearance intent</td>
<td>taxiing phase</td>
<td>CCA4307, clear to MianYang via flight planed route, cruising level 9800 m, expected runway 05 R, Shigezhuang 16 departure, initial altitude 3000 m, QNH1006, departure frequency 119.1, squawk 3212.</td>
<td>Granting departure clearance to aircraft.</td>
</tr>
<tr>
<td>push back and start up intent</td>
<td>taxiing phase</td>
<td>CSN6879, cleared for push back and start up, facing south.</td>
<td>Allowing pushback to change the aircraft’s static state.</td>
</tr>
<tr>
<td>taxi intent</td>
<td>taxiing phase</td>
<td>CCA1270, taxi to stand 183 via S4, Z3, Z6</td>
<td>Providing taxi instructions and relevant routes.</td>
</tr>
<tr>
<td>enter the runway intent</td>
<td>taxiing phase</td>
<td>CSZ9850, cleared for line up</td>
<td>Aircraft can enter the runway and prepare for takeoff.</td>
</tr>
<tr>
<td>takeoff intent</td>
<td>takeoff and landing phase</td>
<td>CXA8521, surface wind 280, 1 m/s, runway 39 L, cleared for takeoff</td>
<td>Aircraft are allowed to take off and provided with necessary meteorological information.</td>
</tr>
<tr>
<td>landing clearance intent</td>
<td>takeoff and landing phase</td>
<td>CSN8317, wind 030, 5 m/s, runway 34 R, clear to land.</td>
<td>Aircraft are allowed to land and provided with necessary meteorological information.</td>
</tr>
<tr>
<td>vacate the runway intent</td>
<td>taxiing phase, takeoff and landing phase</td>
<td>CSC8117, report runway vacated</td>
<td>Command aircraft to vacate the runway.</td>
</tr>
<tr>
<td>procedural flight intent</td>
<td>takeoff and landing phase, approach phase</td>
<td>CCA1267, direct to OBLIK, resume to planned route, resume own navigation</td>
<td>Flight plans and actions based on established procedures and standard operations.</td>
</tr>
<tr>
<td>bias intent</td>
<td>approach phase, cruising phases</td>
<td>CDG8841, offset 5 miles right of the track.</td>
<td>Guide the aircraft to deviate from the original route.</td>
</tr>
<tr>
<td>altitude intent</td>
<td>takeoff and landing phase, approach phase, cruising phases</td>
<td>CXA8509, descend and maintain 2100 m</td>
<td>Adjust the altitude of the aircraft.</td>
</tr>
<tr>
<td>heading intent</td>
<td>takeoff and landing phase, approach phase, cruising phases</td>
<td>DKH1688, turn left heading 090</td>
<td>Adjust the heading of the aircraft.</td>
</tr>
<tr>
<td>wait intent</td>
<td>taxiing phase, takeoff and landing phase, approach phase, cruising phases</td>
<td>CCA1331, hold at H</td>
<td>Maintain current status and await further instructions.</td>
</tr>
<tr>
<td>handover intent</td>
<td>taxiing phase, takeoff and landing phase, approach phase, cruising phases</td>
<td>CES6235, contact Shanghai approach on 125.4, goodbye</td>
<td>Transfer control of the aircraft to the next control unit.</td>
</tr>
<tr>
<td>speed intent</td>
<td>taxiing phase, takeoff and landing phase, approach phase, cruising phases</td>
<td>CCA1550, speed 250 due separation</td>
<td>Adjust the speed of the aircraft.</td>
</tr>
<tr>
<td>notification intent</td>
<td>taxiing phase, takeoff and landing phase, approach phase, cruising phases</td>
<td>DKH6856, no obstruction ahead</td>
<td>Report or inform various flight-related information.</td>
</tr>
<tr>
<td>radar identification intent</td>
<td>taxiing phase, takeoff and landing phase, approach phase, cruising phases</td>
<td>CSH9305, approach radar contact</td>
<td>Inform the aircraft pilot that they have been identified by the radar controller.</td>
</tr>
<tr>
<td>reminder intent</td>
<td>taxiing phase, takeoff and landing phase, approach phase, cruising phases</td>
<td>CSH3528, caution similar callsign in my frequency</td>
<td>Remind the aircraft pilot.</td>
</tr>
<tr>
<td>coordination intent</td>
<td>taxiing phase, takeoff and landing phase, approach phase, cruising phases</td>
<td>CC1271, is the cruising altitude acceptable at 110?</td>
<td>Coordination between the controller and the pilot due to the specificity of the instructions.</td>
</tr>
</tbody>
</table>
In the table above, this study categorizes air–ground communication text into 18 specific intents based on the flight phase. Categorizing text intents for air–ground communication based on flight phases makes it easier to determine the flight phase of the communication. The table demonstrates that there are one-to-one correlations, such as departure clearance intent, taxi intent, etc. In such cases, the flight phase of air–ground communication can be inferred directly from the control intent. However, there are also one-to-many situations that require inferring the flight phase based on contextual clues. For example, if the previous air–ground communication directs the aircraft to climb, it cannot be in the taxiing phase.

4.1.2. Dataset Description

The study collected real air–ground communication audio from multiple airports and control units and converted it to text format using automatic speech recognition (ASR) technology. After obtaining the air–ground communication text, the researchers annotated it using the previously mentioned intents. A single air–ground communication text can include one or more intents. Finally, the study collected 9800 instances of single-intent air–ground communication text data and 3208 instances of multi-intent air–ground communication text data, which comprised the final dataset. For experimentation, the dataset was divided into training, validation, and test sets in a 7:2:1 ratio. The label distribution of the dataset is shown in Figure 6.

![Distribution of the dataset labels.](image)

In this experiment, the dataset faces a number of potential risks, including but not limited to the following:

1. **Errors in converting air–ground communications into text:** Errors in converting air–ground communications to text can occur due to a variety of factors, including technical limitations of the ASR system, environmental noise, speaker accents, diversity in vocabulary expression, etc. These factors may cause partial errors in the recognized air–ground communication text.

2. **Annotation errors:** When converting air–ground communication text into intent labels, there may be annotation errors or inconsistencies. For example, for complex speech content, different annotators may assign different intent labels, resulting in annotation errors.

3. **Imbalanced data:** In actual datasets, the number of samples for different intents may be significantly imbalanced, with some categories having far more or far fewer samples than others. This may result in insufficient learning for minority classes by the model, reducing the model’s generalization capability.

To address the risks present in the dataset, this study proposes the following measures:
(1) To reduce errors in converting air–ground communications to text, we choose an ASR system that demonstrates high accuracy and stability. In addition, we perform manual review and correction of recognition results, using human inspection and proofreading to identify and correct incorrectly recognized text segments.

(2) To mitigate annotation errors, we create clear annotation standards and guidelines that precisely define each air–ground communication intent and provide detailed annotation instructions. Each air–ground communication text is independently annotated by two annotators, and the results are then checked for accuracy and consistency. For discrepant annotations, a third party conducts further examination to resolve inconsistencies and ultimately determine the correct annotation.

(3) To address the issue of data imbalance, this experiment uses stratified sampling, dividing each intent text into training, validation, and test sets in a 7:2:1 ratio to ensure that the sample sizes of each category are relatively balanced across these sets. Considering the pronounced imbalance in the multi-intent dataset, we supplement it with single-label data to increase the sample size.

4.1.3. Experimental Results of ASR Systems

In previous research, a performance evaluation method for ATC speech recognition systems was proposed. First, ATC speech was collected and annotated according to specific ATC scenario proportions to establish a test corpus for the ATC speech recognition system. Next, an evaluation index system for the ATC speech recognition system was designed, and the weights of each index were calculated using the analytic hierarchy process (AHP). Finally, three ATC speech recognition systems were proposed and trained for evaluation and analysis. Through the training and evaluation of deep speech recognition 2 (DeepSpeech2), convolution-augmented transformer (Conformer), and Whisper, Conformer was ultimately selected as the final ATC speech recognition system. The performance of the three ASR systems is shown in Table 4.

<table>
<thead>
<tr>
<th>ASR System</th>
<th>Word Accuracy</th>
<th>Sentence Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepSpeech2</td>
<td>43.98%</td>
<td>8.40%</td>
</tr>
<tr>
<td>Conformer</td>
<td>94.67%</td>
<td>86.54%</td>
</tr>
<tr>
<td>Whisper</td>
<td>91.60%</td>
<td>77.17%</td>
</tr>
</tbody>
</table>

4.2. Experimental Configuration

The software and hardware platforms used in this study are shown in Table 5, and the experimental parameters for the ERNIE 3.0 model and the ASA model are listed in Table 6.

<table>
<thead>
<tr>
<th>Category</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>Windows 13th Gen Intel(R) Core(TM) i7-13700K</td>
</tr>
<tr>
<td>CPU</td>
<td>3.40 GHz (Intel, Santa Clara, CA, USA)</td>
</tr>
<tr>
<td>Memory</td>
<td>32G</td>
</tr>
<tr>
<td>GPU</td>
<td>Nvidia GeForce RTX490 (Nvidia, Santa Clara, CA, USA)</td>
</tr>
<tr>
<td>Python</td>
<td>3.9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>Batch Size</th>
<th>Epochs</th>
<th>Learning Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERNIE 3.0</td>
<td>32</td>
<td>50</td>
<td>$5 \times 10^{-5}$</td>
</tr>
<tr>
<td>ASA</td>
<td>32</td>
<td>100</td>
<td>$3 \times 10^{-5}$</td>
</tr>
</tbody>
</table>
4.3. Indicator Calculation

In multi-intent recognition, a text can correspond to one or more labels from the label set. When evaluating the prediction results of multi-intent recognition, metrics similar to binary classification problems, such as accuracy, recall, and F1 score, are commonly utilized. Although the number of labels in multi-intent recognition is greater than or equal to one, the label categories can still be divided into positive and negative samples for their respective calculations. The following is the calculation process:

- Assuming the given label set is \( L = \{l_1, l_2, \ldots, l_n, h_1, h_2, \ldots, h_m\} \).
  
  \( L_1 = \{l_1, l_2, \ldots, l_n\} \) represents the set of target labels, which can also be understood as the set of positive sample labels, and \( L_2 = \{h_1, h_2, \ldots, h_m\} \) represents the set of non-target labels, which can also be understood as the set of negative sample labels.

- Label example

  The dataset labels for multi-intent recognition are shown in Table 7.

### Table 7. Table of label examples.

<table>
<thead>
<tr>
<th>Text Number</th>
<th>Real Label</th>
<th>One-Hot (Real)</th>
<th>Predicted Label</th>
<th>One-Hot (Predicted)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( l_1 )</td>
<td>[1,0,0,0,...]</td>
<td>( l_1 )</td>
<td>[1,0,0,0,...]</td>
</tr>
<tr>
<td>2</td>
<td>( l_1, l_3 )</td>
<td>[1,0,1,0,...]</td>
<td>( l_2, l_3 )</td>
<td>[0,1,1,0,...]</td>
</tr>
<tr>
<td>3</td>
<td>( l_1, l_2, l_3 )</td>
<td>[1,1,1,0,...]</td>
<td>( l_2, l_3 )</td>
<td>[0,1,1,0,...]</td>
</tr>
</tbody>
</table>

- Label example

  We calculate the following four evaluation metrics for a single text:
  
  \( TP_{sub} \): predicting labels that should be present as present and correct;
  
  \( FN_{sub} \): predicting labels that should be present as absent or predicting labels that should be present but are incorrect;
  
  \( FP_{sub} \): predicting labels that should be absent as present;
  
  \( TN_{sub} \): predicting labels that should be absent as absent.

  The calculation methods for the four evaluation metrics are as follows: \[ TP_{sub} + FN_{sub} + FP_{sub} + TN_{sub} = 1 \]

  In this context, \( TP_{sub} + FN_{sub} + FP_{sub} + TN_{sub} = 1 \). \( i \) represents the number of occurrences of the situations described by the metrics, while \( k \) denotes the total count of non-zero values in both the true labels and the predicted labels. If both the true and predicted labels are one, it is counted only once.

- We calculate the four evaluation metrics below for multiple texts.

  We calculate the overall values of \( TP_{sub}, FN_{sub}, FP_{sub}, \) and \( TN_{sub} \) as \( TP_{total}, FN_{total}, FP_{total}, \) and \( TN_{total} \), respectively:

  \[ TP_{total} = TP_{sub(1)} + TP_{sub(2)} + \ldots + TP_{sub(N)} \] 

  (16)

  \[ FN_{total} = FN_{sub(1)} + FN_{sub(2)} + \ldots + FN_{sub(N)} \] 

  (17)

  \[ FP_{total} = FP_{sub(1)} + FP_{sub(2)} + \ldots + FP_{sub(N)} \] 

  (18)

  \[ TN_{total} = TN_{sub(1)} + TN_{sub(2)} + \ldots + TN_{sub(N)} \] 

  (19)

  where \( N \) is the total number of texts to be evaluated; \( TP_{sub(i)}, FN_{sub(i)}, FP_{sub(i)}, \) and \( TN_{sub(i)} \) represent the values of \( TP_{sub}, FN_{sub}, FP_{sub}, \) and \( TN_{sub} \) in text \( i \), respectively.

- Final evaluation result.

  \[ \text{Precision} = \frac{TP_{total}}{TP_{total} + FP_{total}} \] 

  (20)

  \[ \text{Recall} = \frac{TP_{total}}{TP_{total} + FN_{total}} \] 

  (21)
\[ F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \] (22)

4.4. Ablation Experiment

This study carried out ablation experiments to ensure the effectiveness of each module proposed in the ASA model, and the results are shown in Table 8.

Table 8. Results of the ablation experiments for the ASA model.

<table>
<thead>
<tr>
<th>Model</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALBERT</td>
<td>97.72</td>
<td>96.23</td>
<td>96.97</td>
</tr>
<tr>
<td>AS</td>
<td>96.74</td>
<td>91.78</td>
<td>94.30</td>
</tr>
<tr>
<td>ASA (GRU)</td>
<td>97.53</td>
<td>97.10</td>
<td>97.31</td>
</tr>
<tr>
<td>ASA (LSTM)</td>
<td>97.84</td>
<td>97.17</td>
<td>97.50</td>
</tr>
</tbody>
</table>

In the table above, AS represents the ALBERT_Sequence-to-Sequence model, ASA represents the ALBERT_Sequence-to-Sequence_Attention model, and ASA models employ GRU and LSTM in both the encoder and decoder. The table demonstrates that the baseline ALBERT model already has high precision, recall, and F1 scores. However, the addition of the sequence-to-sequence architecture results in a slight decrease in performance, especially in recall. The introduction of the sequence-to-sequence architecture can cause issues such as information loss, decoder limitations, and generation biases, resulting in some loss or ambiguity in information and affecting the model’s recall performance. Therefore, in the improved experiments, a local attention mechanism was added to the decoder to help the model handle the correlation between inputs and outputs more effectively, thereby improving performance. The local attention mechanism enables the model to focus attention on input sequence parts near the current position of the decoder, reducing the impact of long sequence inputs on model performance and improving the model’s efficiency and accuracy in the task. While using the gated recurrent unit (GRU) as the RNN unit resulted in a slight improvement in performance, precision and recall were slightly lower than when using LSTM. This is because LSTM has superior modeling capabilities to GRU, allowing it to capture long-term dependencies in sequences. Overall, introducing LSTM as the RNN unit and utilizing a local attention mechanism in the decoder can effectively enhance the performance of the ALBERT model in sequence tasks. Particularly noteworthy is the improvement in recall while maintaining high precision, resulting in a more balanced performance.

4.5. Experiment Analysis

The experiment in the research of multi-intent recognition in air–ground communication texts uses the ERNIE 3.0 large model and the ALBERT pre-trained speech model as underlying models. Initially, a multi-label classification model, equivalent to a multi-intent recognition model, was used to examine all air–ground communication texts (including both single-intent and multi-intent texts). The results are shown in Table 9.

Table 9. Recognizing all texts using a multi-label classification model.

<table>
<thead>
<tr>
<th>Model</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>Latency (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERNIE 3.0</td>
<td>97.46</td>
<td>96.18</td>
<td>96.75</td>
<td>1.35</td>
</tr>
<tr>
<td>ALBERT</td>
<td>97.72</td>
<td>96.23</td>
<td>96.97</td>
<td>0.675</td>
</tr>
<tr>
<td>ALBERT_Denses</td>
<td>97.73</td>
<td>96.37</td>
<td>97.05</td>
<td>0.81</td>
</tr>
<tr>
<td>ALBERT_TextCNN</td>
<td>98.11</td>
<td>96.12</td>
<td>97.11</td>
<td>0.878</td>
</tr>
<tr>
<td>ASA</td>
<td>97.84</td>
<td>97.17</td>
<td>97.50</td>
<td>1.08</td>
</tr>
</tbody>
</table>

The experimental results in the table above demonstrate that in the multi-label classification task, the ALBERT pre-trained language model outperforms the ERNIE 3.0 large
model. Therefore, this study selected the ALBERT pre-trained language model as the updated model. The results show that the ALBERT_TextCNN model has the highest precision value but performs poorly in terms of recall. In contrast, the ASA model outperforms the ALBERT_TextCNN model in terms of recall, though its precision value is lower. Considering the comprehensive performance of precision and recall, the model’s performance can be evaluated by the F1 value. The ASA model outperforms other models in terms of F1 value. Meanwhile, by calculating the inference time of each model, it is evident that all inference times are at the millisecond level, resulting in relatively small differences.

In subsequent research, this study classified single-intent and multi-intent texts separately, using a multi-label classification model for recognition, yielding the results shown in Table 10.

**Table 10.** Using a multi-label classification model to recognize single-intent and multi-intent texts separately.

<table>
<thead>
<tr>
<th>Model</th>
<th>Single-Label Precision</th>
<th>Multi-Label Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERNIE 3.0</td>
<td>94.05</td>
<td>98.58</td>
</tr>
<tr>
<td>ALBERT</td>
<td>93.59</td>
<td>99.05</td>
</tr>
<tr>
<td>ALBERT_Denses</td>
<td>93.64</td>
<td>98.74</td>
</tr>
<tr>
<td>ALBERT_TextCNN</td>
<td>93.69</td>
<td>99.24</td>
</tr>
<tr>
<td>ASA</td>
<td>94.00</td>
<td>99.39</td>
</tr>
</tbody>
</table>

According to the results in the table above, the multi-label classification model performs slightly worse in the single-intent text category but performs well in the multi-intent text category, with the ASA model performing best. Therefore, this study separately classified single-intent texts and used a multi-class classification model, which is equivalent to a single-intent recognition model, for single-intent recognition. The specific results are shown in Table 11.

**Table 11.** Using a multi-class classification model to recognize single-intent texts.

<table>
<thead>
<tr>
<th>Model</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>Latency (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERNIE 3.0</td>
<td>96.23</td>
<td>95.85</td>
<td>96.04</td>
<td>1.035</td>
</tr>
<tr>
<td>BERT</td>
<td>95.44</td>
<td>95.49</td>
<td>95.42</td>
<td>0.828</td>
</tr>
<tr>
<td>BERT + CNN</td>
<td>95.62</td>
<td>95.09</td>
<td>95.30</td>
<td>0.911</td>
</tr>
<tr>
<td>BERT + RNN</td>
<td>96.06</td>
<td>95.59</td>
<td>95.73</td>
<td>1.242</td>
</tr>
<tr>
<td>BERT + RCNN</td>
<td>95.29</td>
<td>95.41</td>
<td>95.29</td>
<td>1.076</td>
</tr>
<tr>
<td>BERT + DPCNN</td>
<td>95.40</td>
<td>94.94</td>
<td>95.08</td>
<td>0.952</td>
</tr>
</tbody>
</table>

The results in the table demonstrate that the ERNIE 3.0 model outperforms the other models in the single-intent recognition task. It outperforms BERT and other BERT-based models in terms of precision, recall, and F1 value. Figures 7 and 8 compare the effects of using multi-label classification models versus multi-class classification models for recognizing single-intent texts.

The graph depicts a line chart of precision values for both the multi-label and multi-class classification models in the single-intent recognition task. The graph clearly demonstrates that the multi-class classification model has significantly higher precision values in the single-intent recognition task than the multi-label classification model. The recall and F1 values in Figure 8 also validate the superiority of the multi-class classification model.

In summary, in the single-intent recognition task, the multi-class classification model outperforms the multi-label classification model. Therefore, in the multi-intent recognition task, recognizing single-intent and multi-intent texts separately can achieve better recognition results. Meanwhile, it is worth noting that the inference times of the ERNIE 3.0 model for single-intent recognition and the ASA model for multi-intent recognition are quite similar, both remaining at the millisecond level with minimal latency. In the aviation field, although latency is an important factor, accuracy is always the primary concern.
because accurate recognition results are crucial for ensuring aviation safety and efficiency. Therefore, slight latency is acceptable.

This study investigated the prediction results thoroughly, analyzing the predictions of both the multi-class classification model and the multi-label classification model in detail. In terms of predicting single-label texts, the multi-class classification model consistently produces a single label, whereas the multi-label classification model can produce one or more labels, increasing the possibility of incorrect predictions. Therefore, for single-label text predictions, the multiclass classification model outperforms the multi-label classification model.

Figure 7. Comparison of precision between multi-label classification model and multi-class classification model in single-intent recognition task.

Figure 8. (a) Comparison of recall between multi-label classification model and multi-class classification model in single-intent recognition task.; (b) Comparison of F1 score between multi-label classification model and multi-class classification model in single-intent recognition task.
5. Conclusions

This study proposes a hybrid detection approach for multi-intent recognition in air–ground communication texts. By utilizing multi-intent detection technology, air–ground communication texts are classified into single-intent and multi-intent texts for separate recognition. Experimental results demonstrate that using the ASA model for multi-intent text recognition achieved an accuracy rate of 97.84%, which is 0.34% higher than the baseline ALBERT model and 0.15% to 0.87% higher than other improved models based on ALBERT and ERNIE 3.0. Meanwhile, using the multi-class classification model for single-intent text recognition yields an accuracy of 96.23%, which is at least 2.18% higher than the multi-label model. The innovation of this study lies in distinguishing air–ground communication texts into single-intent and multi-intent texts using multi-intent detection technology and employing different models for intent recognition, accordingly, thereby significantly improving the accuracy of recognition. Additionally, the ASA model is proposed to further enhance the recognition effect. With the increase in air traffic flow, ATC faces an increasingly complex communication environment. Accurately identifying multiple intents in air–ground communications can effectively detect and verify the instructions and responses between pilots and controllers, ensuring the precise transmission and execution of commands, thereby reducing safety hazards caused by misunderstandings or misjudgments. Moreover, accurate multi-intent recognition can more precisely record the communication content during flights, providing strong data support for post-flight analysis and accident investigations, among other purposes.

In future research, further exploration of multi-class classification models and multi-label classification models will be conducted, incorporating multi-modal data, such as speech, radar images, and flight plans, to assist text intent recognition in order to achieve better recognition performance. Additionally, joint research on multi-intent recognition tasks and slot-filling tasks is planned, using the identified intentions to select different slots for filling. In the slot-filling task, multi-call sign recognition technology will also be studied to handle situations where controllers speak to multiple pilots in a single sentence, ensuring the effective identification and differentiation of different pilots’ call signs. This joint research will make the slot-filling task more accurate and efficient, enabling the precise extraction of key information from air–ground communications. This helps controllers and pilots quickly obtain the necessary information during the decision-making process, thereby improving the overall efficiency of ATC. Moreover, the potential applications of these technologies in other fields, such as drone control and autonomous aircraft navigation and control, will be explored. Through this research, the aim is to drive the ATC system towards greater automation and intelligence, bringing revolutionary improvements to aviation safety and efficiency.

Author Contributions: Conceptualization, W.P. and Z.W. (Zixuan Wang); methodology, W.P.; software, Z.W. (Zixuan Wang); validation, Z.W. (Zhuang Wang), W.P. and Z.W. (Zixuan Wang); formal analysis, Z.W. (Zhuang Wang); investigation, Y.H.; resources, W.P.; data curation, Y.W.; writing—original draft preparation, W.P.; writing—review and editing, Z.W. (Zixuan Wang); visualization, Z.W. (Zixuan Wang); supervision, Z.W. (Zhuang Wang); project administration, W.P.; funding acquisition, W.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (U2333209), National Key R&d Program of China (No.2021YFF0609040), the National Natural Science Foundation of China (U1733203), the Safety Capacity Building Project of Civil Aviation Administration of China (TM2019-16-1/3), the Civil Aircraft Fire Science and Safety Engineering Key Laboratory of Sichuan Province (MZ2024J801), the 2024 Annual Central University Fundamental Research Funds Support Project (24CAFU04025), and the 2024 Annual Central University Fundamental Research Funds Support Project (24CAFU10184).

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy and ethical concerns.

Conflicts of Interest: The authors declare no conflicts of interest.
Abbreviations
The following abbreviations are used in this manuscript:

ERNIE Enhanced Representation through Knowledge Integration
ALBERT A Lite Bidirectional Encoder Representations from Transformers
ASA ALBERT_Sequence-to-Sequence_Attention
SVM Support Vector Machine
CNN Convolutional Neural Networks
RNN Recurrent Neural Networks
LSTM Long Short-Term Memory
BiLSTM Bidirectional Long Short-Term Memory
BERT Bidirectional Encoder Representations from Transformers
DP Dependency Parsing
TF-IDF Term Frequency-Inverse Document Frequency
GRU Gate Recurrent Unit
BiGRU Bidirectional Gate Recurrent Unit
EBA ERNIE-Gram_BiGRU_Attention
ATC Air Traffic Control
COO Coordinate Relationships
GPT Generative Pre-Trained Transformer
MLM Masked Language Modeling
NLP Natural Language Processing
ICAO International Civil Aviation Organization
ASR Automatic Speech Recognition
ASRS Aviation Safety Reporting System
LS Label Spreading
SIU Speech Instruction Understanding
ABSR Assistant-Based Speech Recognition
ASRU Automatic Speech Recognition and Understanding
CAAC Civil Aviation Administration of China
AHP Analytic Hierarchy Process
DeepSpeech2 Deep Speech Recognition 2
Conformer Convolution-Augmented Transformer

References


19. Rose, R.L.; Puranik, T.G.; Mavris, D.N. Natural Language Processing Based Method for Clustering and Analysis of Aviation Safety Narratives. Aerospace 2020, 7, 143. [CrossRef]

20. Madeira, T.; Melicio, R.; Valério, D.; Santos, L. Machine Learning and Natural Language Processing for Prediction of Human Factors in Aviation Incident Reports. Aerospace 2021, 8, 47. [CrossRef]

21. Miyamoto, A.; Bendarkar, M.V.; Mavris, D.N. Natural Language Processing of Aviation Safety Reports to Identify Inefficient Operational Patterns. Aerospace 2022, 9, 450. [CrossRef]


24. Lin, Y. Spoken Instruction Understanding in Air Traffic Control: Challenge, Technique, and Application. Aerospace 2021, 8, 65. [CrossRef]


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.