Method And Apparatus for Extracting Commands Between A Controller And An Operator

Field

The present invention relates to a method and apparatus for extracting commands between a controller (e.g. air traffic controller, marine vessel traffic controller or operator, land traffic controller, etc.) and an operator (e.g. a pilot, vessel operator, vehicle driver, etc.). In particular, the present invention is capable of, for air traffic control application, checking whether a pilot's repeat-back response to a flight command or instruction is consistent with the flight command or instruction issued by the controller.

Background

In the aviation sector, the extraction of instructions between pilot and Air Traffic Controller (ATC) conversations is of paramount importance. Effective communication between pilots and Air Traffic Controllers (ATCs) is a cornerstone of flight safety. Any ambiguity or misinterpretation can have dire consequences. One critical aspect of this communication is the read-back process, where pilots repeat back instructions to ensure clarity and mutual understanding. Accurately extracting instructions is vital for detecting read-back errors discrepancies between the original instruction and the pilot's repeat. Such errors, if unnoticed, can lead to operational inconsistencies and pose significant safety risks. Therefore, a reliable instruction extraction system is not just a technological advancement but a crucial tool for enhancing aviation safety. Similar issues may arise in other areas where repeat-back may also be involved. For example, communication between a vessel traffic controller and a ship/boat operator, between a land traffic controller and land vehicle operators, between a commanding officer and soldiers, etc.

Summary

According to an example of the present disclosure, there are provided a method and apparatus for extracting commands between a controller and an operator as claimed in the independent claims. Some optional features are defined in the dependent claims.

Brief Description of the Drawings

Examples in the present disclosure will be better understood and readily apparent to one skilled in the art from the following written description, by way of example only and in conjunction with the drawings, in which:

Fig. 1 shows a diagram illustrating communication between a pilot and an air traffic controller.

Fig. 2 shows key components of an apparatus or system according to an example of the present disclosure.

Fig. 3 illustrates a command or instruction extraction method or process according to an example of the present disclosure.

Fig. 4 illustrates steps and components of a fuzzy matching method according to an example of the present disclosure.

Fig. 5 shows a table showcasing the performance of an apparatus and method according to an example of the present disclosure.

Fig. 6 shows a system architecture of an example of an apparatus or system of the present disclosure.

Detailed Description

Examples of apparatuses (or systems) and/or methods of the present disclosure may be configured to extract instructions in conversations between a controller and an operator. For instance, in the aviation sector, pilot (which is essentially an aircraft operator) and Air Traffic Controller (ATC) conversations. A dual approach for instruction extraction is proposed. It incorporates a Bi-LSTM-CRF (Bidirectional Long Short-Term Memory-Conditional Random Field) model that is adept at identifying complete instructions and a fuzzy algorithm for matching. One objective of an example of the present disclosure is to check that a read-back reply in response to a command contains expected instruction.

It should be noted that although extraction of flight commands or instructions between ATC and pilot are discussed extensively in the present disclosure, the same apparatus (or system) and method described in the present disclosure can also be applied to communications between a vessel traffic controller and a ship/boat operator, between a land traffic controller and land vehicle operators, between a commanding officer and

soldiers, etc. In such communications, a read-back reply in response to a command may also be involved.

An example relating to the aviation sector will now be described. Due to the special nature of dialogue in aviation, pilots will read back the instructions given by ATC. there are specific examples in table 1 below. For conversation 01, the instruction information is complete. For conversation 02, when the pilot reads back, the instruction information reported by the pilot is only in numbers. For this kind of instruction, which are only numbers or incomplete, it is very difficult to extract such information. To solve this problem, in an example of the present disclosure, in addition to the Bi-LSTM-CRF model, a fuzzy matching part is proposed.

Table 1 - Two pilots-ATCs conversations (with instructions underlined):

CONVERSATION	SPEAKER	UTTERANCE
01	ATC	FLIGHT 123 YOU SHOULD STOP DESCEND FLIGHT LEVEL ONE FOUR ZERO
	PILOT	STOP DESCEND FLIGHT LEVEL ONE FOUR ZERO
02	ATC	FLIGHT 123 MAINTAIN SPEED TWO EIGHT ZERO KNOTS OR GREATER
	PILOT	ROGER TWO EIGHT ZERO

Fig. 1 illustrates a conversation between an aircraft pilot and an air traffic controller.

In the field of aviation, the precise extraction of instructions from dialogues between pilots and Air Traffic Controllers (ATCs) takes on paramount importance. One of the primary applications of the present example described is in the detection of 'read-back errors' i.e. discrepancies where pilots may misinterpret or incorrectly repeat back the given instructions. Such scenarios may occur due to pilots abbreviating the read-back instructions. Read-back errors are pivotal considerations in flight safety, as they can lead to operational challenges and heighten safety risks. This is also applicable in other areas such as in communications between a vessel traffic controller and a ship/boat operator, between a land traffic controller and land vehicle operators, between a commanding officer and soldiers, etc. Moreover, beyond just detecting errors, effective instruction extraction within the Air Traffic Management (ATM) realm ensures clearer

communication and facilitates smoother flight operations. Furthermore, the instructions from the conversations can be used in post-mission analysis. Therefore, the present example aims to be a reliable instruction extraction system that is not just a technological advancement but a crucial tool for the aviation field.

Specifically, with reference to Fig. 2, the present example is an instruction extraction system 202 employing a dual strategy that balances the effective extraction of both complete and incomplete instructions. The technology foundation is based on Natural Language Processing (NLP) technology. The system is based on a Bidirectional Long Short-Term Memory-Conditional Random Field (Bi-LSTM-CRF) model 204, which performs well in the recognition of complete instructions 206. To compensate the model's 204 limitations in the extraction of incomplete instructions 210, a fuzzy matching method 208 is used. This fuzzy matching method 208 takes the complete instructions provided by the air traffic controller as a reference and successfully extracts the incomplete instructions from the pilots. This ensures the completeness and accuracy of instructions and the extraction results significantly enhance downstream tasks performance. Examples of pilot-ATC conversations are shown in table 212 in Fig. 2.

The instruction extraction process begins with labelling instruction data using the "Begin-Inside-Outside" (BIO) method, followed by employing the BI-LSTM CRF model 204 to extract complete instructions 206. For incomplete instructions 210, a fuzzy matching mechanism for performing the fuzzy matching method 208 specifically makes use of a complete instruction given by the ATC as a reference to extract information from the incomplete instruction from the pilot. The combination of the BI-LSTM CRF model 204 and the fuzzy matching method 208 ensures comprehensive coverage of instruction types, facilitating clearer ATC-pilot communication. The combination adapts to irregular/incomplete instructions and varied expressions, and has high accuracy.

The dual approach may involve the following:

1. Instruction data labelling: About 8000 sentences manually annotated using the BIO method for training and testing.
2. Use of the BI-LSTM CRF model 204 to extract complete instructions.
3. Adding fuzzy match for special cases: In the airline industry or field, pilots sometime repeat parts of ATCs' instructions. Those special repeated

instruction might only have some numbers. This is difficult to just use the BI-LSTM CRF model to extract.

An advantage of the dual approach is that it not only improves the detection of read-back errors but also ensures clearer communication, facilitating smoother flight operations.

Details of an apparatus or system and the method 300 it uses for flight instruction information extraction of the present example is described below with reference to Fig. 3.

Firstly, instruction data 306 comprising one or more transcripts of voice conversation is labelled via a "Begin-Inside-Outside" (BIO) method. The output would be labelled instruction data 308. The Bi-LSTM-CRF model 204 then extracts complete instructions from the labelled instruction data 308. The output of the Bi-LSTM-CRF model 204 is ATC and pilot conversations with extracted complete instructions 302. The instruction extraction task performed by the Bi-LSTM-CRF model 204 involves identifying instructions within the dialogue between ATCs and pilots and determining the corresponding instruction type. Note that in some cases, the pilot would be issuing commands and the ATC would response and the roles are reversed. In any case, the objective is to ensure that the response contains the repeat-back command present in the command given or contains the expected response for the command given.

In the present example, about 8000 manually annotated sentences (of pilot-ATC conversations) also annotated using the "Begin-Inside-Outside" (BIO) method are used as the training and testing dataset for the Bi-LSTM-CRF model 204.

In machine learning, particularly in the field of Natural Language Processing (NLP), the "Begin-Inside-Outside (BIO)" method is a tagging scheme used to identify the boundaries of named entities within a text by labelling each word as either the "Beginning" (B) of an entity, "Inside" (I) an entity, or "Outside" (O) of any entity, allowing the model to accurately extract multi-word entities from a sentence. The key points about the BIO method are as follows.

Tagging scheme:

Each word in a sentence is assigned one of three tags: "B" (Beginning of entity), "I" (Inside of entity), or "O" (Outside of any entity).

Identifying entity boundaries:

The "B" tag clearly indicates the start of a named entity, while the "I" tag marks subsequent words belonging to the same entity. For example, beginning of a named entity can be the word “FLIGHT” and subsequent words such as “123” can be the number of the flight indicating which aircraft. A word that may be outside of any entity could be a word like “THE”, which is not essential to an aircraft instruction or command.

The BI-LSTM CRF model 204 is understood as a combination of Bidirectional Long Short-Term Memory (BI-LSTM) networks and Conditional Random Fields (CRF). This model is effective for sequence labelling tasks and would be useful for extracting instructions or commands from text. The combination of BI-LSTM and CRF allows the model to effectively capture the context and dependencies in the text. The process involving use of the BI-LSTM CRF model 204 may be as follows.

A first step may involve converting the spoken voice instructions into a text transcript. This can be done using Automatic Speech Recognition (ASR) systems. It is assumed that the text transcript is perfect and free of ASR errors. Examples of the present disclosure is not about correcting ASR errors.

Next, the text transcript is then tokenized into individual words or tokens. For example, "FLIGHT 123 YOU SHOULD STOP DESCEND FLIGHT LEVEL ONE FOUR ZERO" becomes [“FLIGHT” “123” “YOU” “SHOULD” “STOP” “DESCEND” “FLIGHT” “LEVEL” “ONE” “FOUR” “ZERO”].

Each token then converted into a dense vector representation using word embeddings (e.g., Word2Vec, GloVe, or BERT). For example, the word "STOP" might be represented as a vector like [0.2, 0.1, -0.3 ...].

A BI-LSTM layer processes the sequence of word vectors in both forward and backward directions to capture context from both sides. For example, about the above tokenization example, the forward LSTM processes the sequence from "FLIGHT" to "ZERO", while the backward LSTM processes it from "ZERO" to "FLIGHT".

The outputs from the forward and backward LSTMs are concatenated to form a comprehensive representation of each token. For example, with reference to the above tokenization example, if the forward LSTM output for "FLIGHT" is [0.2, 0.1, -0.3] and

the backward LSTM output is [-0.1, 0.4, 0.2], the concatenated output is [0.2, 0.1, -0.3, -0.1, 0.4, 0.2].

A CRF layer takes the concatenated outputs and models the dependencies between labels to predict the most likely sequence of labels for the tokens. For example, with reference to the above tokenization example, the CRF layer might label B-NAME for "FLIGHT", which means beginning of a flight name, and I-NAME as inside the flight name i.e. it is part of the flight name. "YOU" and "SHOULD" may be labelled as O, which means outside of any entity as they are not essential to the command. Furthermore, "STOP" might be labelled as B-CMD (beginning of an instruction or command; "CMD" means command), "DESCEND" as I-CMD (inside a command i.e. part of the command), and so on. In summary, the CRF layer is used for sequence label prediction of the words in a command or response. It should be appreciated that methods besides CRF suitable such sequence label prediction may be used as well.

Specifically, Conditional Random Field (CRF) is a type of statistical modelling method used for structured prediction in machine learning and natural language processing (NLP). Here are some key points about CRFs:

1. Structured Prediction: Unlike traditional classifiers that predict labels for individual samples independently, CRFs consider the context by modelling the dependencies between predictions. This makes them useful for tasks where the output labels are interdependent, such as sequence labelling.
2. Graphical Model: CRFs are undirected probabilistic graphical models. They represent the conditional probability of a set of output variables given a set of input variables, using a graph structure where nodes represent variables and edges represent dependencies.
3. Discriminative Model: CRFs are discriminative models, meaning they model the conditional probability directly rather than modelling the joint probability of inputs and outputs.

The output of the CRF layer is a sequence of labels that indicate which tokens form part of an instruction or command. For example, ["STOP" "DESCEND" "FLIGHT" "LEVEL" "ONE" "FOUR" "ZERO"] might be labelled as [B-CMD, I-CMD, B-NAME, B-ALT ("ALT" means altitude), I-ALT, I-ALT, I-ALT], where B-CMD indicates the beginning of a command, I-CMD indicates inside a command i.e. it is part of the command, B-ALT indicates the beginning of an (aircraft) altitude, and I-ALT indicate

the inside of the altitude i.e. it refers to the contents of the altitude. The output then identifies "STOP DESCEND" as the command, "FLIGHT LEVEL ONE FOUR ZERO" as the aircraft altitude, which the aircraft has to stop descending to. The output of the CRF layer makes it easy to identify the word or span of each command or instruction to be extracted for further processing later.

Secondly, a fuzzy matching algorithm or system is used to extract information from incomplete flight instructions. After extracting complete instructions from the ATCs' and pilots' conversations using the Bi-LSTM-CRF model 204, the fuzzy matching algorithm or system performs the Fuzzy Matching method 208. This method 208 is grounded on Levenshtein distance and overlooks word order while incorporating a length penalty mechanism and a numeric match reward mechanism. This is done to ensure it does not merely favour perfectly or 100% matched spans and appropriately considers numeric word sequences. In other words, the length penalty mechanism and complete numeric match reward system ensures the algorithm will not favour 100% matched spans that are not in numeric word order or sequence. The method quantitatively assesses similarity with a scoring system where a higher score signifies better matching. After analysis by the Fuzzy Matching method 208, pilots' responses are matched against ATCs' complete instructions. Ultimately, a comprehensive set of both complete and incomplete instructions from the inputted conversations is compiled as output 304. In other words, the output 304 is an integrated dataset of both complete and incomplete instructions. In other examples, other types of edit distances besides Levenshtein distance may be used, for example, Hamming distance, Jaccard Similarity, Jaro distance, etc.

The details of a preferred example of the Fuzzy Matching method 208 are described as follows with reference to Fig. 4.

Using the complete instruction 302 extracted from the ATC's statement as a comparison object, a window-based iteration 408 is performed over all spans of the pilot's response. The specific example of the complete instruction 302 shown in Fig. 4 that is inputted to the fuzzy matching algorithm is the same as conversation 02 in table 1 described above. Underlined text "TWO EIGHT ZERO" in Fig. 4 is extracted by the Bi-LSTM-CRF model (204 in Fig. 2 and 3). The reference text containing the complete ATC instruction is MAINTAIN SPEED TWO EIGHT ZERO KNOTS OR GREATER. The target text to be compared by the fuzzy matching algorithm is TWO EIGHT ZERO.

A span (or text segment) refers to a part of the text in the (pilot's) response and the (ATC's) instruction. A span can have one or more words. For instance, if a window of 2 consecutive words is set, a span would be every two consecutive words in the pilot's response.

Preferably, when comparing spans, the step size of the spans is increased iteratively. Firstly, single-word spans are compared, then move on to two-word spans, followed by three-word spans, and so on. In other words, the step size of the spans for the comparison is increased by one word at a time. The comparison is performed such that the step size is iteratively increased, beginning with one word and progressively moving to more words. See the example below.

Example:
Assume that the ATC instruction is:
"Turn left heading two five zero."
And the pilot's response is:
"Turning left heading two five zero."
The Span Matching Process is explained below:

1. The ATC Instruction is not divided: The complete ATC instruction is kept intact, and the goal is to match substrings from the pilot's response to the entire ATC instruction, finding the longest similar substring.
2. Span Matching on Pilot's Response: Extract substrings of various lengths from the pilot's response. For instance, start by extracting 1-word, 2-word, 3-word, and so on substrings from the pilot's response. For each substring extracted from the pilot's response, compare it to the complete ATC instruction and calculate the edit distance (or similarity). Smaller edit distances indicate that the substring is more similar to the ATC instruction.
3. Span Matching Details:
   For example, extracting 1-word substrings from the pilot's response:
   "Turning"
   "left"
   "heading"
   "two"
   "five"
   "zero"

Next, extract 2-word substrings:

"Turning left"

"left heading"

"heading two"

"two five"

"five zero"

Then, extract 3-word substrings:

"Turning left heading"

"left heading two"

"heading two five"

"two five zero"

Next, extract 4-word substrings:

"Turning left heading two"

"left heading two five"

"heading two five zero"

Finally, extract the 5-word substrings:

"Turning left heading two five"

"left heading two five zero"

4. Select the Longest Matching Substring: By calculating the edit distance for each span with the complete ATC instruction, the most similar and longest matching substring is identified.

In the example above, focusing purely on text-based matching, the Span Matching parts (i.e. the substrings being compared) indicate that the pilot's response and the ATC instruction are fully consistent in the relevant sections (i.e. they are matching).

In the present example, Levenshtein distance is calculated by a Levenshtein algorithm 402 in a manner ignoring the word order between each span of the pilot's response and the comparison object (i.e. a span in the ATC's instruction) by the Fuzzy Matching method 208. This ensures that a thorough comparison is conducted. A similarity score is produced by the Levenshtein algorithm 402 for each pair of compared spans.

After a base similarity score is produced by the Levenshtein algorithm 402, a length penalty mechanism 406 is used for punishing cases where the span of the pilot's response under comparison is longer than that of the ATC's instruction. The length penalty mechanism 406 may also penalize cases where the response span is

significantly shorter than the ATC's instruction, specifically less than 40% of the instruction's length. The percentage of 40% is predetermined and can be adjusted if required.

The Fuzzy Matching method 208 is also configured to use a complete numeric match reward system 408 to adjust the score provided by the Levenshtein distance. As a pilot's repeat-back response is being assessed, there should be strict adherence to numeric word order where possible. Hence, the complete numeric match reward system 408 imposes a penalty when the numeric order of the spans compared is disrupted. However, if the numeric values completely match, the score is multiplied by a reward coefficient.

The Levenshtein algorithm 402, also known as the Levenshtein distance or edit distance, measures the similarity between two text strings by calculating the minimum number of single-character edits (insertions, deletions, or substitutions) required to transform one string into the other. For the example of Fig. 4, a first string (or first span) is the complete ATC instruction (i.e. the reference text) extracted by the Bi-LSTM-CRF model and a second string (or second span) is the pilot's response (i.e. the target text). The Levenshtein algorithm 402 may work as follows.

1. Tokenization: First, the two input strings are tokenized into words or spans. This means breaking down the strings into individual words or segments.
2. Sorting: The words or spans within each string may be sorted in a sequence or in some other predetermined consistent order. This step ensures that the order of words is considered during the comparison. In one example, the words or spans may be sorted according to their order of appearance from left to right in the sentence of the ATC's instruction or pilot's response e.g. in the response "ROGER TWO EIGHT ZERO", ROGER is first, TWO is second, EIGHT is third and so on.
3. Levenshtein Distance Calculation: The Levenshtein algorithm 402 is then applied to compare the lists of words or spans in the ATC's instruction and the pilot's response. For each span, the algorithm calculates the minimum number of single-character edits (insertions, deletions, or substitutions) required to transform one sorted list into the other.

Specifically, the Levenshtein algorithm 402 takes in as input a pair of input words or spans (i.e. two strings) comprising a first set of words or spans taken from the ATC's

instruction, and a second set of words or spans taken from the pilot's response. The Levenshtein algorithm 402 is iterated or looped for all spans of the ATC's instruction and the pilot's response (i.e. step 408). This means that each defined span in the pilot's response is compared with each defined span in the ATC's instruction and a list of scores would tabulated for each comparison. The Levenshtein algorithm 402 performs the following:

1. Initialization: Create a matrix with dimensions (m+1) x (n+1), where m and n are the lengths of the two inputted strings i.e. a pair of input words or spans, one set from the ATC's instruction and the other set from the pilot's response. The matrix is used to store the edit distances between all prefixes of the two strings.
2. Base Case: Initialize the first row and the first column of the matrix. The value at position (i, 0) is set to i, and the value at position (0, j) is set to j. This represents the edit distance between an empty string and a prefix of the other string.
3. Recurrence Relation: Fill in the rest of the matrix using the following recurrence relation:

- If the characters at positions i-1 and j-1 in the two strings are the same, the value at position (i, j) is the same as the value at position (i-1, j-1).
- If the characters are different, the value at position (i, j) is the minimum of the following three values, plus one:
    1) The value at position (i-1, j) (deletion)
    2) The value at position (i, j-1) (insertion)
    3) The value at position (i-1, j-1) (substitution)

4. Result: The value at the bottom-right corner of the matrix represents the Levenshtein distance between the two input strings.

To produce a base similarity score, for example, the Levenshtein distance can be normalised by the Levenshtein algorithm 402 by dividing it by the maximum possible distance (which is the length of the longer string). Subtract this normalized distance from 1 to get a similarity score between 0 and 1, where 1 indicates identical strings and 0 indicates completely different strings.

The calculated similarity score for each comparison is then adjusted by the length penalty mechanism 404 if required. The length penalty mechanism 404 punishes cases where the span (i.e. string, or one or more words) of the pilot's response under

comparison is longer than that of the ATC's instruction. That is, the initially calculated similarity score will be lowered by a predetermined amount if the span of the pilot's response under comparison is longer than the span of the ATC's instruction. The length penalty mechanism 404 may also penalize cases where the response span is significantly shorter than the ATC's instruction, for example, specifically less than 40% of the instruction's length. That is, if the pilot response's span is shorter than the span of the ATC's instruction by a predetermined threshold, the similarity score will also be lowered by a predetermined amount.

Examples of how the length penalty calculation may be performed by the length penalty mechanism 404 are described below.

Firstly, compute length ratio, *LR*, using the following equation:

LR = length of pilot response (words) / length of ATC instruction (words)

The Final Similarity Score after length penalty calculation can be obtained as follows:

Final_score = original_score × length_w × penalty_w

length_w is a weightage apply based on the LR. *length_w can be LR or 1 / LR depending on the LR value.*
penalty_w is a tunable penalty multiplier applied based on the LR. The value of *penalty_w* can set as greater or equal to 1.
"*original_score*" refers to the base or initial similarity score calculated.

For example, if the pilot response has length in a predefined ideal or optimal range of $0.7 \leq LR \leq 1$, both length_w and penalty_w are set as 1. In this case, *Final_score = original_score.* See Example 1 below.

Example 1: Optimal Response Length (No Penalty)

ATC Instruction: *"Turn left heading two five zero."* (6 words)
Pilot Response: *"Left heading two five zero."* (5 words)
LR = 5/6 = 0.83
LR is within range of 0.7 to 1.0, so no penalty is applied.
For no penalty, both *length_w* and *penalty_w* are set as 1.

Initial Similarity Score: 0.85

Final Score: *0.85 × 1 × 1 = 0.85*

When the pilot response is shorter than the ATC instruction, for instance when *LR* < 0.7, *length_w* is set as *LR*. The value of *penalty_w* can be set as a default value of 1.4. The *penalty_w* for shorter response can also be scaled according to the value of *LR* so that for example, there is heavier penalty if the pilot response is much shorter than the ATC instruction and lower penalty if the pilot response is closer to the length of the ATC instruction. See Example 2 below.

Example 2: Shorter Response (Penalty Applied)

ATC Instruction: *"Climb and maintain flight level three three zero."* (8 words)

Pilot Response: *"Climbing three three zero."* (4 words, 50% shorter than ATC length)

LR = 4/8 = 0.5

LR is less than 0.7, so penalty is applied.

As mentioned above, the *penalty_w* for shorter response can be set as 1.4.

Initial Similarity Score: 0.80

Final Similarity Score: *0.80 × 0.5 × 1.4 = 0.56* (pilot response penalized for being too short)

When the pilot response is longer than the ATC instruction, for instance when *LR* > 1, *length_w* is set as *LR*. The value of *penalty_w* can also be set as a default value of 1.4. The *penalty_w* for longer response can also be scaled according to the value of *LR* so that for example, there is heavier penalty if the pilot response is much longer than the ATC instruction and lower penalty if the pilot response is closer to the length of the ATC instruction. See Example 3 below.

Example 3: Longer Response (Penalty Applied)

ATC Instruction: *"Climb to flight level three zero."* (i.e. ATC length = 6 words)

Pilot Response: *"Climb and maintain the flight level three three zero."* (i.e. Pilot length = 9 words; 50% longer than ATC length)

LR = 9/6 = 1.5

LR is more than 1, so penalty is applied.

length_w = 1 / LR = 1 / 1.5 = 0.67

As mentioned above, the *penalty_w* for longer response can be set as 1.4.

Initial Similarity Score: 0.80

Final Similarity Score: *0.80 × 0.67 × 1.4 = 0.75* (pilot response penalized for being too long)

In the above examples, the calculated similarity score for each comparison is also adjusted by the complete numeric match reward system 406. This system 406 imposes a penalty when the numeric order is disrupted. This system 406 is proposed because the numbers spoken and the order of speaking these numbers by the pilot and the ATC instruction are particularly crucial for the aviation sector. That is, if the spoken numbers in the pilot response is not the same as the spoken numbers in the ATC's instruction and/or not spoken in order in the speech, the system 406 will lower the similarity score. For instance, if spoken numbers in the ATC instruction is "two three eight" but the pilot response is "two three six", the similarity score should be penalized. In another example, if a first spoken number in the ATC instruction is "two three eight" and only the second spoken number in the pilot response is the same, the similarity score should also be penalized because the pilot did not say the numbers in the correct order. However, if the numeric order completely matches, for instance, all spoken numbers in both the ATC instruction and the pilot response are identical and are spoken in the same order, the score may be increased by, for instance, multiplying it by a predetermined reward coefficient.

An example of how the numeric order penalty works is described below.

The complete numeric match reward system 406 adjusts the similarity score based on how well the numbers spoken in the pilot's response match those in the ATC's instruction. If more than one numbers are spoken, the order of their appearance in the ATC instruction and pilot response will be taken into consideration. The Formula Representation is as follows.

Numeric Match Score = Fuzzy Similarity (ATC Numbers, Pilot Numbers) × numpenalty_weight

"Fuzzy Similarity (ATC Numbers, Pilot Numbers)" is a function taking in only the ATC spoken numbers and the Pilot spoken numbers as input. Hence, this function is only concerned with the spoken numbers in the ATC instruction and the pilot response. This function measures how well the spoken numbers of the ATC and Pilot match in content.

Levenshtein algorithm or other suitable algorithm can be used to perform this function. When there are more than one spoken numbers in the ATC instruction and pilot response, this function is used to compare these spoken numbers in the order of their appearance as well as compare them without regard to their order of appearance. This function outputs an average matching score taking into consideration of both the similarity of the spoken numbers in the order of their appearance as well as their similarity when order of their appearance is disregarded. In this case, highest matching score of this function would be computed if the spoken numbers match both in the order of their appearance and when they are not compared in order of their appearance. Lowest matching score of this function would be computed if the spoken numbers do not match both in the order of their appearance and when they are not compared in order of their appearance.

"numpenalty_weight" is a tunable parameter that determines the impact of numeric mismatches. The default value can be set as 0.02. "numpenalty_weight" is also called Numeric Match Score.

Examples of application of Numeric Order Penalty are below.

Example 1: Perfect Numeric Match (Full Reward Applied)

ATC Instruction: *"Turn left heading two five zero."*

Pilot Response: *"Roger, turning left heading two five zero"*

ATC spoken number: two five zero

Pilot spoken number: two five zero

Numeric Match Score = 100 (perfect match) × 0.02 (numpenalty_weight is set as 0.02 as default) = 2

Final Similarity Score = Initial Similarity Score × 2 (no penalty is applied and a reward is given for perfect match)

The higher the Final Similarity Score, the higher the similarity between the ATC instruction and the Pilot Response.

Example 2: Mismatched Order of Numbers (Penalty Applied)

ATC Instruction: *"Turn left heading two five zero, then descend to flight level one eight zero."*

Pilot Response: *"Turning left heading one eight zero, descending to flight level two five zero."* (The order of the spoken numbers "two five zero" and "one eight zero" in the response are swapped, resulting in different meaning)
The spoken numbers are extracted and compared in order of appearance.
ATC first spoken number: two five zero
Pilot first spoken number: one eight zero
There is a mismatch in the first spoken number.
ATC second spoken number: one eight zero
Pilot second spoken number: two five zero
There is also a mismatch in the second spoken number.
Applying the Fuzzy Similarity function, Numeric Match Score (for two spoken numbers) = 40 (low match due to incorrect order of both spoken numbers in the pilot response) × 0.02 (numpenalty_weight = 0.02 as default) = 0.8
Final Similarity Score = Initial Similarity Score × 0.8 (penalty is applied)

For Example 2 above, the matching score of the Fuzzy Similarity function for the two spoken numbers is not zero but has a value of 40 because the function also calculate scores (for averaging or aggregating) for spoken number comparisons where order of appearance is not considered. When order of appearance is not considered, ATC first spoken number would match with Pilot second spoken number and ATC second spoken number would match with Pilot first spoken number.

The output after processing by the length penalty mechanism 404 and the complete numeric match reward system 406 will be a list of compared pilot spans and the respective final similarity scores, which are adjusted as decided by the length penalty mechanism 404 and the complete numeric match reward system 406. The span in the list with the highest score meeting a predefined threshold is indicative that the read-back in the pilot's response correctly reflects what the ATC instructed in the ATC's instruction. In the specific example of Fig. 4, the span (comprising 3 words) with best final score in the pilot's response (i.e. the target text) is TWO EIGHT ZERO.

The Data and Experimental results of an example of an apparatus or system configured according to the description of Fig. 1 to 4 is described below.

The dataset used for experiment comprises 200 hours of ATC-Pilot radar voice conversations recorded on seven channels over a 44-day period. Close to 8000

utterances are manually annotated for the dataset using the BIO method. The data of the close to 8000 annotated utterances is first used to train the instruction extraction model i.e. the BI-LSTM-CRF model. After analyzing the bad cases of the BI-LSTM-CRF model in the instructions it extracted, it was discovered that this model performs poorly in recognizing incomplete instructions. However, after using the BI-LSTM-CRF model with fuzzy matching method, the observations show that almost all incomplete instructions are successfully extracted.

Table 2 - The instruction extraction performance for using only BI-LSTM-CRF model and using BI-LSTM-CRF model with Fuzzy Matching method:

	Precision		Recall		F1-Score		Support
	BI-LSTM-CRF	**BI-LSTM-CRF WITH FUZZY MATCH**	BI-LSTM-CRF	**BI-LSTM-CRF WITH FUZZY MATCH**	BI-LSTM-CRF	**BI-LSTM-CRF WITH FUZZY MATCH**	
Macro avg	0.92	**0.93**	0.93	**0.97**	0.92	**0.94**	162
Weighted avg	0.92	**0.94**	0.89	**0.96**	0.90	**0.94**	162

From the tabulated data in table 2, it can be clearly observed that Weighted Avg F1-Score achieves a 4% improvement after the introduction of the fuzzy matching method. This result confirms the effectiveness of the fuzzy matching method.

With reference to Fig. 5, table 502 shows the actual number (which is 7926) of pilot-ATC utterances (i.e. ATC instructions and pilot responses) used for training the Bi-LSTM-CRF model. The utterances cover instructions and responses pertaining to various aspects or categories of flight control, including altimeter, flight level, speed, taxi etc. Table 504 in Fig. 5 also shows that Weighted Avg F1-Score achieves a 4% improvement after the introduction of fuzzy matching method.

Furthermore, table 504 shows another model using only BERT and CRF being tested and the results are not as promising as using only Bi-LSTM-CRF and using Bi-LSTM-CRF with the fuzzy matching method. That said, BERT-CRF with fuzzy matching

method could be a viable option and would be considered as one of the examples of the present disclosure.

BERT is a transformer-based model that uses a bidirectional approach to understand the context of words in a sentence by looking at both the left and right sides of a word simultaneously. BiLSTM is a type of recurrent neural network (RNN) that processes data in both forward and backward directions, capturing dependencies from both past and future contexts. BERT and BiLSTM share some characteristics, despite their different architectures:

1. Contextual Understanding: Both models are designed to capture the context of words within a sentence, allowing for more accurate understanding and generation of text.
2. Bidirectional Processing: They both process text in a bidirectional manner. BERT looks at the entire sentence from both directions simultaneously, while Bi-LSTM processes the text in both forward and backward directions.
3. Deep Learning: Both models are based on deep learning techniques, leveraging multiple layers to learn complex patterns in data. Deep learning is a subset of machine learning that uses artificial neural networks with multiple layers to process information and learn from data, allowing computers to recognize complex patterns and make decisions similar to how humans do, often by analyzing large amounts of data to improve accuracy over time.

CRF may be used for sequence labelling to label text into specific categories or classes. CRF is one technique for sequence labelling. It should appreciated that other techniques for sequence labelling may also be used. For example, use of a Softmax layer, attention mechanisms, etc.

The Strategic test set selection stated in table 504 comprises 162 samples meticulously chosen to represent a wide spectrum of cases encountered in real-world scenarios. They include a mix of sentence lengths, ensuring both short and complex sentences are tested, which can demonstrate the tested system's robustness across varying lengths. That is, challenging cases are included and most scenarios are covered.

The performance evaluation described above with reference to tables 2 and Fig. 5 demonstrates significant improvements in precision, recall, and F1-score across

various flight control instruction categories. This enhanced accuracy is pivotal for real-time error detection and aviation safety. The results underscore the effectiveness of the proposed solution according to examples of the present disclosure in enhancing aviation safety through improved instruction extraction and read-back error detection.

In general, a trained neural network that processes text in a bidirectional manner to extract or capture text resembling an instruction is preferred. In addition, it is preferred to perform sequence labelling to label each element in the inputted text. With regard to flight command extraction application, the complete instruction of the ATC is extracted and preferably labelled by the trained neural network. Furthermore, in general, the fuzzy matching algorithm or method is a technique used to compare strings to determine their similarity. Specifically, with regard to flight command extraction application, the objective is to extract repeat-back flight commands from a pilot's response to ensure that they are the same as what was instructed in the ATC's flight command. The fuzzy matching method compares the text of the completion instruction of the ATC's flight command (which was extracted by the trained neural network) and the text of the pilot's response to the flight command to find text that are similar. Thereafter, the fuzzy matching method extracts the text found in the pilot's response that is most similar to the text in extracted instruction of the flight command.

The Fuzzy matching algorithm or method (e.g. 208) described in the present disclosure can involve machine learning, but they do not always have to. The approach described in the present disclosure does not require machine learning but are based on edit distance (e.g., Levenshtein distance), Jaccard similarity, or Soundex that do not inherently involve machine learning. These methods use predefined rules to measure the similarity between strings. If required, machine learning can be used to enhance fuzzy matching by learning from data to improve accuracy. For example, machine learning models can be trained to optimize the weights of different features or to predict the likelihood of a match based on historical data.

Few-shot learning (FSL) is a machine learning approach where models are trained to recognize new classes with only a few examples. This is useful when collecting large amounts of labelled data is impractical. In FSL, the model to be trained is provided with a small set of labelled examples (support set) for each new class. When a new, unlabelled example is presented, the model compares it with the support set to

determine the closest match. FSL is useful in Natural Language Processing for tasks like text classification with limited labelled examples.

Zero-shot learning (ZSL) enables models to recognize and classify objects or concepts they have never seen before. This is achieved by leveraging semantic information about the classes. In ZSL, the model to be trained uses descriptions or attributes of unseen classes to make predictions. By understanding the relationships between seen and unseen classes, the model can infer the characteristics of the unseen classes.

The Bi-LSTM-CRF model may be trained by few-shot learning (FSL) and/or zero shot learning (ZSL). The fuzzy match algorithm may also be trained via few-shot learning (FSL) and/or zero shot learning (ZSL) if machine learning is involved.

Fig. 6 shows an example of an apparatus or system 600 that can be used to implement the apparatus or system of Fig. 1 and the features illustrated by Figures 2 to 4. The apparatus 600 can be a local computer or remote server (e.g. cloud server) in which users can submit an audio input (or a video input containing audio) to be transcribed by an Automatic Speech Recognition engine. With regard to flight command extraction application, such audio input would contain ATC flight commands and pilot responses or pilot commands and ATC responses. The transcription of the audio input would be processed by the apparatus or system 600 to extract flight commands according to the manner described with reference to Fig. 1 to 4.

The apparatus 600 may comprise a processing unit (or processor) 602 for processing software including one or more programs for running one or more computer/server/device applications to enable a backend logic flow or the method or methods for carrying out the relevant software steps required to implement the features of the earlier described Figures.

Furthermore, the processing unit 602 may include user input modules such as a computer mouse 636, keyboard/keypad 604, and/or a plurality of output devices such as a display device 608. The display device 608 may incorporate technologies like LCD, LED, OLED, and the like. It may also be a touch screen capable of receiving user input.

The processing unit 602 may be connected to a computer network 612 via a suitable transceiver device 614 (i.e. a network interface), to enable access to e.g. the Internet or other network systems such as a wired Local Area Network (LAN) or Wide Area Network (WAN). The processing unit 602 may be connected to one or more external wireless communication enabled devices 634 via a suitable wireless transceiver device 632, e.g. a WiFi transceiver, Bluetooth module, Mobile telecommunication transceiver suitable for Global System for Mobile Communication (GSM), 3G, 4G, 5G telecommunication systems, and the like. Through the computer network (i.e. internet) 612, the processing unit 602 can gain access to one or more storages i.e. data storages, databases, data servers and the like connectable to the computer network 612 to retrieve and/or store data in the one or more storages. The external wireless communication enabled devices 634 may be user computers or mobile devices (e.g. tablet computer, smartphones etc.) and users may use such devices to connect to the processing unit 602.

The processing unit 602 may include a microprocessor 618, a Random Access Memory (RAM) 620 and a Read Only Memory (ROM) 622. In the case of a server, the processing unit 602 may also include a number of Input/Output (I/O) interfaces, for example I/O interface 638 to the computer mouse 636, a memory card slot 616, I/O interface 624 to the display device 608, and I/O interface 626 to the keyboard/keypad 604.

The components of the processing unit 602 typically communicate via an interconnected bus 628 and in a manner known to the person skilled in the relevant art.

The programs may be supplied to the user of the processing unit 602, or the processor (not shown) of one of the one or more external wireless communication enabled devices 634, encoded on a data storage medium such as a CD-ROM, on a flash memory carrier, Solid State Drive, or a Hard Disk Drive, and are to be read using a corresponding data storage medium drive of a data storage device 630. Such computer or application programs may also be downloaded from the computer network 612. The application programs are read and controlled in its execution by the processor 618. Intermediate storage of program data may be accomplished using RAM 620.

In more detail, one or more of the computer or application programs may be stored on any non-transitory machine- or computer-readable medium. The machine- or computer-readable medium may include storage devices such as magnetic or optical disks, memory chips, or other storage devices suitable for interfacing with a general-purpose computer. The machine- or computer- readable medium may also include a hard-wired medium such as that exemplified in the Internet system, or wireless medium such as that exemplified in the Wireless LAN (WLAN) system and the like. The computer program when loaded and executed on such a general-purpose computer effectively results in an apparatus that implements the steps of the computing methods in examples herein described.

Specifically, examples of the present disclosure may include a method of extracting flight commands between a controller and a pilot. The method comprising:

(a) Obtaining, from the controller, a first speech input containing a flight command;

(b) Obtaining, from the pilot, a second speech input containing a readback of the flight command;

(c) extracting instructions from the first speech input and the second speech input using a Bidirectional Long Short-Term Memory-Conditional Random Field (Bi-LSTM-CRF) model; and

(d) comparing instructions extracted from the second speech input with instructions extracted from the first speech input using a fuzzy algorithm and extracting the instructions from the second speech input taking into consideration results of the comparison.

In one embodiment, the fuzzy algorithm returns a score by calculating the Levenshtein distance between text extracted from the second speech and text extracted from the first speech input, and wherein the fuzzy algorithm is adapted to overlook word order and to modify the score by including a length penalty mechanism and a numeric match reward mechanism.

In another embodiment, the score is lowered if a length of the text extracted from the second speech input is less than 40% of a length of the extracted from the first speech input, and wherein the score is increased if there is a match between numeric word order of the text extracted from the second speech input and numeric word order of the text extracted from the first speech input.

In another embodiment, the Bi-LSTM-CRF model is trained by a dataset annotated using a Begin-Inside-Outside (BIO) method.

In another embodiment, the fuzzy algorithm involves Artificial Intelligence involving few-shot learning (FSL) and/or zero shot learning (ZSL) and/or the Bi-LSTM-CRF model is trained by few-shot learning (FSL) and/or zero shot learning (ZSL).

In yet another embodiment, the method includes employing a comprehensive dual strategy that is able to balance the effective extraction of both complete and incomplete instructions. Bi-LSTM-CRF model performs well in the recognition of complete instructions for both ATC and pilots. The fuzzy matching method compensates for the lack of extraction of incomplete instructions for pilots.

The proposed solution provides an Integrated Extraction System for Complete and Incomplete Instructions that leverages the Bi-LSTM-CRF model to proficiently extract both complete instructions from ATC and pilot conversations and subsequently apply a Fuzzy Matching method to identify and incorporate incomplete instructions.

The Advanced Fuzzy Matching Methodology uniquely disregards word order and implements a penalty-reward mechanism for length variations and numeric matches respectively, optimizing for real-world aviation dialogues where perfect matches might not be the norm.

Comprehensive Data Compilation for Enhanced Flight Safety and Error Detection is provided by synthesizing a dataset of both complete and incomplete directives post extraction and matching, the system not only facilitates clearer communication between ATCs and pilots but also provides a robust foundation for "read-back error detection". This integration serves as a pivotal tool for identifying discrepancies in repeated instructions, thus promoting heightened operational safety in the aviation sector.

The proposed solution represents a significant advancement in enhancing flight safety and operational efficiency. By addressing the limitations of current methods, we offer a scalable, accurate system for instruction extraction in critical communication scenarios.

Generally, examples of the present disclosure covering not just the aviation sector but other relevant sectors may include the following features. Reference numerals of the elements in the figures of the present disclosure that are examples of the features discussed are provided in brackets.

A method (e.g. 300) for extracting commands between a controller and an operator, the method comprising:

receiving a command and a response to the command (e.g. 212);

extracting an instruction (e.g. 302 and specifically the reference text shown in Fig. 4) from the command using a trained neural network model (e.g. 204) that processes text in a bidirectional manner;

comparing the extracted instruction from the command and the response to the command to find text that is similar (e.g. find the target text in Fig. 4); and

extracting text in the response to the command found to be most similar to the text in the extracted instruction of the command (e.g. 304),

wherein a fuzzy matching algorithm (e.g. 208) is used for comparing the extracted instruction of the command and the response to the command to find text that is similar, wherein the fuzzy matching algorithm compares a span of the extracted instruction of the command with a span of the response to the command and calculates a similarity score for the two compared spans.

The trained neural network may be a Bidirectional Long Short-Term Memory (Bi-LSTM-CRF) model that comprises a Conditional Random Field layer for labelling one or more elements in a sequence in the text inputted to the Bi-LSTM-CRF model.

The similarity score may be calculated based on Levenshtein distance (e.g. 402).

The fuzzy matching algorithm may compare each of a plurality of spans determined for the extracted instruction of the command with each of a plurality of spans determined for the response to the flight command to obtain a plurality of corresponding similarity scores for each pair of compared spans (e.g. 408).

Each span may comprise one or more words, and comparison of spans of the command with spans of the response is performed iteratively, increasing by a step size

of one word, beginning with spans containing one word and progressively moving to spans containing more words.

A length penalty mechanism (e.g. 404) may be used to penalise a calculated similarity score of a pair of compared spans if the length of the span of the response to the command of the pair of compared spans is different from the length of the span of the extracted instruction of the command of the pair of compared spans.

The length penalty mechanism may be used to penalise the calculated similarity score of the pair of compared spans if the length of the span of the response to the command of the pair of compared spans is less than the length of the span of the extracted instruction of the command of the pair of compared spans by a predetermined percentage of 40% or more.

A numeric order match reward mechanism (e.g. 406) may be used to process spoken numbers in the extracted instruction and response, and reward or penalise the calculated similarity score of a pair of compared spans depending on whether there is a match in the spoken numbers and order of appearance of the spoken numbers in the extracted instruction and the response

The trained neural network model may be trained by a dataset annotated using a Begin-Inside-Outside (BIO) method.

The trained neural network model may be trained using few-shot learning (FSL) and/or zero shot learning (ZSL).

An apparatus (e.g. 600) for extracting commands between a controller and an operator, wherein the apparatus comprises:

a processor (e.g. 602) configured to execute instructions in a memory to operate the apparatus to:

receive a command and a response to the command (e.g. 212);

extract an instruction (e.g. 302 and specifically the reference text shown in Fig. 4) from the command using a trained neural network model (e.g. 204) that processes text in a bidirectional manner;

compare the extracted instruction from the command and the response to the command to find text that is similar (e.g. find the target text in Fig. 4); and

extract text in the response to the command found to be most similar to the text in the extracted instruction of the command (e.g. 304),

wherein a fuzzy matching algorithm (e.g. 208) is used for comparing the extracted instruction of the command and the response to the command to find text that is similar, wherein the fuzzy matching algorithm compares a span of the extracted instruction of the command with a span of the response to the command and calculates a similarity score for the two compared spans.

With regard to the apparatus, the trained neural network may be a Bidirectional Long Short-Term Memory (Bi-LSTM-CRF) model that comprises a Conditional Random Field layer (e.g. 204) for labelling one or more elements in a sequence in the text inputted to the Bi-LSTM-CRF model.

With regard to the apparatus, wherein the similarity score may be calculated based on Levenshtein distance (e.g. 402).

With regard to the apparatus, the fuzzy matching algorithm may compare each of a plurality of spans determined for the extracted instruction of the command with each of a plurality of spans determined for the response to the command to obtain a plurality of corresponding similarity scores for each pair of compared spans (e.g. 408).

With regard to the apparatus, wherein each span may comprise one or more words, and comparison of spans of the command with spans of the response is performed iteratively, increasing by a step size of one word, beginning with spans containing one word and progressively moving to spans containing more words.

With regard to the apparatus, wherein a length penalty mechanism (e.g. 404) may be used to penalise a calculated similarity score of a pair of compared spans if the length of the span of the response to the command of the pair of compared spans is different from the length of the span of the extracted instruction of the command of the pair of compared spans.

With regard to the apparatus, wherein the length penalty mechanism may be used to penalise the calculated similarity score of the pair of compared spans if the length of the span of the response to the command of the pair of compared spans is less than

the length of the span of the extracted instruction of the command of the pair of compared spans by a predetermined percentage of 40% or more.

With regard to the apparatus, wherein a numeric order match reward mechanism (e.g. 406) may be used to process spoken numbers in the extracted instruction and response, and reward or penalise the calculated similarity score of a pair of compared spans depending on whether there is a match in the spoken numbers and order of appearance of the spoken numbers in the extracted instruction and the response.

With regard to the apparatus, wherein the trained neural network model may be trained by a dataset annotated using a Begin-Inside-Outside (BIO) method.

With regard to the apparatus, wherein the trained neural network model may be trained using few-shot learning (FSL) and/or zero shot learning (ZSL).

A non-transitory computer-readable medium storing a command extraction program including instructions that, when executed by a processor (e.g. 602), causes an apparatus (e.g. 600) to:

receive a command and a response to the command (e.g. 212);

extract an instruction (e.g. 302 and specifically the reference text shown in Fig. 4) from the command using a trained neural network model (e.g. 204) that processes text in a bidirectional manner;

compare the extracted instruction from the command and the response to the command to find text that is similar (e.g. find the target text in Fig. 4); and

extract text in the response to the command found to be most similar to the text in the extracted instruction of the command (e.g. 304),

wherein a fuzzy matching algorithm (e.g. 208) is used for comparing the extracted instruction of the command and the response to the command to find text that is similar, wherein the fuzzy matching algorithm compares a span of the extracted instruction of the command with a span of the response to the command and calculates a similarity score for the two compared spans.

In the present disclosure, unless the context clearly indicates otherwise, the term "comprising" has the non-exclusive meaning of the word, in the sense of "including at least" rather than the exclusive meaning in the sense of "consisting only of". The same

applies with corresponding grammatical changes to other forms of the word such as "comprise", "comprises" and so on.

While the invention has been described in the present disclosure in connection with a number of examples, embodiments and implementations, the invention is not so limited but covers various obvious modifications and equivalent arrangements, which fall within the purview of the appended claims. Although features of the invention are expressed in certain combinations among the claims, it is contemplated that these features can be arranged in any combination and order.

CLAIMS

1. A method for extracting commands between a controller and an operator, the method comprising:

receiving a command and a response to the command;

extracting an instruction from the command using a trained neural network model that processes text in a bidirectional manner;

comparing the extracted instruction from the command and the response to the command to find text that is similar; and

extracting text in the response to the command found to be most similar to the text in the extracted instruction of the command,

wherein a fuzzy matching algorithm is used for comparing the extracted instruction of the command and the response to the command to find text that is similar, wherein the fuzzy matching algorithm compares a span of the extracted instruction of the command with a span of the response to the command and calculates a similarity score for the two compared spans.

2. The method of claim 1, wherein the trained neural network is a Bidirectional Long Short-Term Memory (Bi-LSTM-CRF) model that comprises a Conditional Random Field layer for labelling one or more elements in a sequence in the text inputted to the Bi-LSTM-CRF model.

3. The method of claim 1 or 2, wherein the similarity score is calculated based on Levenshtein distance.

4. The method of claim 1, 2 or 3, wherein the fuzzy matching algorithm compares each of a plurality of spans determined for the extracted instruction of the command with each of a plurality of spans determined for the response to the flight command to obtain a plurality of corresponding similarity scores for each pair of compared spans.

5. The method of claim 4, wherein each span comprises one or more words, and comparison of spans of the extracted instruction with spans of the response is performed iteratively, increasing by a step size of one word, beginning with spans containing one word and progressively moving to spans containing more words.

6. The method of any one of claims 1 to 5, wherein a length penalty mechanism is used to penalise a calculated similarity score of a pair of compared spans if the length of the span of the response to the command of the pair of compared spans is different from the length of the span of the extracted instruction of the command of the pair of compared spans.

7. The method of claim 6, wherein the length penalty mechanism is used to penalise the calculated similarity score of the pair of compared spans if the length of the span of the response to the command of the pair of compared spans is less than the length of the span of the extracted instruction of the command of the pair of compared spans by a predetermined percentage of 40% or more.

8. The method of the preceding claims, wherein a numeric order match reward mechanism is used to process spoken numbers in the extracted instruction and response, and reward or penalise the calculated similarity score of a pair of compared spans depending on whether there is a match in the spoken numbers and order of appearance of the spoken numbers in the extracted instruction and the response.

9. The method of any one of the preceding claims, wherein the trained neural network model is trained by a dataset annotated using a Begin-Inside-Outside (BIO) method.

10. The method of any one of the preceding claims, wherein the trained neural network model is trained using few-shot learning (FSL) and/or zero shot learning (ZSL).

11. An apparatus for extracting commands between a controller and an operator, wherein the apparatus comprises:

a processor configured to execute instructions in a memory to operate the apparatus to:

receive a command and a response to the command;

extract an instruction from the command using a trained neural network model that processes text in a bidirectional manner;

compare the extracted instruction from the command and the response to the command to find text that is similar; and

extract text in the response to the command found to be most similar to the text in the extracted instruction of the command,

wherein a fuzzy matching algorithm is used for comparing the extracted instruction of the command and the response to the command to find text that is similar, wherein the fuzzy matching algorithm compares a span of the extracted instruction of the command with a span of the response to the command and calculates a similarity score for the two compared spans.

12. The apparatus of claim 11, wherein the trained neural network is a Bidirectional Long Short-Term Memory (Bi-LSTM-CRF) model that comprises a Conditional Random Field layer for labelling one or more elements in a sequence in the text inputted to the Bi-LSTM-CRF model.

13. The apparatus of claim 11 or 12, wherein the similarity score is calculated based on Levenshtein distance.

14. The apparatus of claim 11, 12 or 13, wherein the fuzzy matching algorithm compares each of a plurality of spans determined for the extracted instruction of the command with each of a plurality of spans determined for the response to the command to obtain a plurality of corresponding similarity scores for each pair of compared spans.

15. The apparatus of claim 14, wherein each span comprises one or more words, and comparison of spans of the extracted instruction with spans of the response is performed iteratively, increasing by a step size of one word, beginning with spans containing one word and progressively moving to spans containing more words.

16. The apparatus of any one of claims 11 to 15, wherein a length penalty mechanism is used to penalise a calculated similarity score of a pair of compared spans if the length of the span of the response to the command of the pair of compared spans is different from the length of the span of the extracted instruction of the command of the pair of compared spans.

17. The apparatus of claim 16, wherein the length penalty mechanism is used to penalise the calculated similarity score of the pair of compared spans if the length of the span of the response to the command of the pair of compared spans is less than the length of the span of the extracted instruction of the command of the pair of compared spans by a predetermined percentage of 40% or more.

18. The apparatus of any one of claims 11 to 17, wherein a numeric order match reward mechanism is used to process spoken numbers in the extracted instruction and response, and reward or penalise the calculated similarity score of a pair of compared spans depending on whether there is a match in the spoken numbers and order of appearance of the spoken numbers in the extracted instruction and the response.

19. The apparatus of any one of claims 11 to 18, wherein the trained neural network model is trained by a dataset annotated using a Begin-Inside-Outside (BIO) method.

20. The apparatus of any one of the claims 11 to 19, wherein the trained neural network model is trained using few-shot learning (FSL) and/or zero shot learning (ZSL).

21. A non-transitory computer-readable medium storing a command extraction program including instructions that, when executed by a processor, causes an apparatus to:
receive a command and a response to the command;
extract an instruction from the command using a trained neural network model that processes text in a bidirectional manner;
compare the extracted instruction from the command and the response to the command to find text that is similar; and
extract text in the response to the command found to be most similar to the text in the extracted instruction of the command,
wherein a fuzzy matching algorithm is used for comparing the extracted instruction of the command and the response to the command to find text that is similar, wherein the fuzzy matching algorithm compares a span of the

extracted instruction of the command with a span of the response to the command and calculates a similarity score for the two compared spans.