United States Patent Application Publication	20250259437
Kind Code	A1
Publication Date	August 14, 2025
Inventor(s)	Yoon; Seunghyun et al.

SYSTEMS AND METHODS FOR IMAGE PROCESSING USING NATURAL LANGUAGE

Abstract

Embodiments of the disclosure provide a machine learning model for generating a predicted executable command for an image. The learning model includes an interface configured to obtain an utterance indicating a request associated with the image, an utterance sub-model, a visual sub-model, an attention network, and a selection gate. The machine learning model generates a segment of the predicted executable command from weighted probabilities of each candidate token in a predetermined vocabulary determined based on the visual features, the concept features, current command features, and the utterance features extracted from the utterance or the image.

Inventors: **Yoon; Seunghyun (San Jose, CA), Bui; Trung Huu (San Jose, CA), Dernoncourt; Franck (San Jose, CA), Kim; Hyounghun (Chapel Hill, NC), Kim; Doo Soon (San Jose, CA)**

Applicant: **Adobe Inc.** (San Jose, CA)

Family ID: **1000008563446**

Appl. No.: **19/098934**

Filed: **April 02, 2025**

Related U.S. Application Data

parent US continuation 17651771 20220218 parent-grant-document US 12293577 child US 19098934

Publication Classification

Int. Cl.: **G06V10/86** (20220101); **G06F40/284** (20200101); **G06N3/044** (20230101); **G06N3/088** (20230101); **G06V10/77** (20220101); **G06V10/80** (20220101); **G06V10/82** (20220101)

U.S. Cl.:

CPC **G06V10/86** (20220101); **G06F40/284** (20200101); **G06N3/044** (20230101); **G06N3/088** (20130101); **G06V10/7715** (20220101); **G06V10/806** (20220101); **G06V10/82** (20220101);

Background/Summary

CROSS-REFERENCE TO RELATED APPLICATIONS [0001] This application is a continuation of U.S. patent application Ser. No. 17/651,771 filed Feb. 18, 2022, entitled, "SYSTEMS AND METHODS FOR IMAGE PROCESSING USING NATURAL LANGUAGE," the content of which is incorporated herein by reference in its entirety.

TECHNICAL FIELD
[0002] The present disclosure relates to systems and methods for processing images using natural language, and more particularly to systems and methods for generating executable commands for controlling an image editing tool using natural language to process an image.
BACKGROUND
[0003] Processing images such as editing and/or searching relative images are widely used, e.g., in project designs such as posters, brochures, flyers, etc. Normally, operations performed on an image are executed through controlling image editing tools. For example, when a user intends to perform a "blur the detail operation" on an image, the user has to find the technical term of the image characteristic (e.g., sharpness) corresponding to his intention (e.g., change the clarity of the detail in the image), and find the command button corresponding to the technical term in the editing tool interface. Existing image editing tools can be complex to operate because the image editing tools presume certain expertise (e.g., knowledge in photo effects) to know the proper technical term to select the correct executable command. Also, the more sophisticated an image editing tool is (e.g., having many possible editing operations), the more complex the operating interface will be.
[0004] Embodiments of the disclosure address these and other problems by providing methods and systems for using natural language to process an image.
SUMMARY
[0005] Embodiments of the disclosure provide a non-transitory computer-readable medium storing instruction that, when executed by one or more processors, cause the one or more processors to perform operations for generating a predicted executable command for an image. The operation includes receiving, via a user interface, an utterance indicating a request associated with an image and generating an utterance feature vector based on utterance features extracted from the utterance. The operation also includes accessing the image corresponding to the utterance and generating a visual feature vector based on visual features extracted from the image. The operation further includes generating a concept feature vector based on concept features extracted from the image and generating a first fused feature vector based on aligning the utterance feature vector and the visual feature vector. The operation additionally includes generating a second fused feature vector based on aligning the first fused feature vector and a current command feature vector and generating a segment of a predicted executable command corresponding to the request associated with image based on the second fused feature vector, the current command feature vector, the utterance feature vector, and the concept feature vector.
[0006] Embodiments of the disclosure provide also provide a computer-implemented method for generating a predicted executable command for an image, the computer-implemented method. The method includes receiving an utterance indicating a request associated with the image and generating an utterance feature vector based on inputting the utterance into a recurrent neural

network. The method also includes receiving the image corresponding to the utterance and generating a visual feature vector and a concept feature vector based on inputting the image into a convolutional network. The method further includes generating a first fused feature vector based on inputting the utterance vector and the visual vector to an attention mechanism; generating a second fused feature vector based on inputting the first fused feature vector and a command feature vector to the attention mechanism. The method yet includes calculating weighted probabilities for each candidate token in a predetermined vocabulary based on the second fused feature vector, the current command feature vector, the utterance feature vector, and the concept feature vector and generating a segment of the predicted executable command based on the weighted probabilities.
[0007] Embodiments of the disclosure provide further provide a machine learning model for generating a predicted executable command for an image. The machine learning model includes an utterance sub-model configured to extract utterance features from an utterance indicating a request associated with the image and generate an utterance feature vector based on the utterance features, and a visual sub-model configured to extract visual features and concept features from the image and generate a visual feature vector and a concept feature vector based on the visual features and the concept features respectively. The machine learning model also includes an attention network configured to generate a first fused feature vector based on aligning the utterance feature vector and the visual feature vector and generate a second fused feature vector based on aligning the first fused feature vector and a current command feature vector. The machine learning model further includes a selection gate configured to generate a segment of the predicted executable command from weighted probabilities of each candidate token in a predetermined vocabulary determined based on the second fused feature vector, the concept features, and the utterance features.
[0008] It is to be understood that both the foregoing general descriptions and the following detailed descriptions are exemplary and explanatory only and are not restrictive of the invention, as claimed.

Description

BRIEF DESCRIPTION OF THE DRAWINGS
[0009] FIG. **1** illustrates a block diagram of an exemplary image editing system, according to embodiments of the disclosure.
[0010] FIG. **2** illustrates a block diagram of an exemplary executable command generation device, according to embodiments of the disclosure.
[0011] FIG. **3** illustrates a block diagram of a machine learning model for generating executable commands for controlling the image editing device, according to embodiments of the disclosure.
[0012] FIG. **4** illustrates a flowchart of an exemplary method for generating executable commend, according to embodiments of the disclosure.
[0013] FIG. **5** illustrates a flowchart of an exemplary method for calculating a total token score for each candidate token in the predetermined vocabulary, according to embodiments of the disclosure.
[0014] FIG. **6** illustrates a conceptual diagram of training a learning model, according to embodiments of the disclosure.
[0015] FIG. **7**. shows a block diagram of an example computing device for implementing the methods, systems, and techniques described herein in accordance with some embodiments.
[0016] FIG. **8** shows an exemplary user interface, according to embodiments of the disclosure.
[0017] FIG. **9** shows a conceptual diagram of an example of the outputs generate by a selection gate, according to according to embodiments of the disclosure.
[0018] FIGS. **10**A and **10**B show exemplary data collection interface for user-annotators and assistant-annotators respectively, according to according to embodiments of the disclosure.
DETAILED DESCRIPTION

[0019] Reference will now be made in detail to the exemplary embodiments, examples of which are illustrated in the accompanying drawings. Wherever possible, the same reference numbers will be used throughout the drawings to refer to the same or like parts.

[0020] Existing image editing tools require certain expertise (e.g., expertise in photo effects, color theory, etc.) to make proper editing commends (e.g., color change, brightness change, contrast change, etc.). Also, different image editing tools have different interfaces which are usually complicated. It is especially hard for beginners (e.g., someone not familiar with image processing) to use existing image editing tools to process (e.g., perform operations on) image(s).

[0021] The technical solution disclosed herein allows natural language (e.g., utterance(s)) to be used to perform operations on images (e.g., search an image and/or editing an image). Specifically, a trained machine learning model (e.g., implemented using one or more neural networks) will take utterance(s) as input to generate executable command(s) for controlling an image editing tool (e.g., Adobe Photoshop™, Adobe Lightroom™, etc.) to process the image. According to certain embodiments, the technical solution performs intermediate operations e.g., predicting/generating executable commands indicating an operation request on a relevant image in response to the input utterance. In some scenarios, the relevant image is a search result of a search engine (e.g., by searching the Internet and/or a database) invoked by the utterance, and/or is an image pre-selected by the user. Examples of the executable commands include editing the relevant image (e.g., color change, brightness change, contrast change, rotation, background removal, etc.) or searching an image (e.g., using characteristics of the image to find other images with similar characteristics).

[0022] In some embodiments, in addition to the input utterance indicating the intended action/operation (e.g., editing and/or searching request on the image), other inputs are also fed to the trained machine learning model to generate a current executable command. These other inputs include the image history (e.g., a series of images previously presented simultaneously or sequentially), the utterance history (e.g., the utterances previously input during the same communication session/conversation), and/or the previously generated/predicted executed command (e.g., appended to the last utterance in the utterance history). The current executable command corresponds to the intended action/operation and will be outputted by the trained machine learning model in a segment-by-segment manner (e.g., one segment at a time). A segment of an executable command is also referred to as a token. Accordingly, operations can be performed on an image using natural language (e.g., by inputting utterances) to improve the usability of the image editing tools and to reduce the complexity of the user interface.

[0023] FIG. **1** illustrates a block diagram of an exemplary image processing system **100**, according to embodiments of the disclosure. Image processing system **100** includes an executable command generation device **110** to generate executable command **107** by using a trainable machine learning model (e.g., machine learning model **105**). Executable command **107** is configured to perform operations (e.g., searching and/or editing) on an image (e.g., an image included as part of model input **103**) based on one or more utterance **102**. In some embodiments, machine learning model **105** is trained by a model training device **120** using training data **101** that includes sample utterance(s), sample image(s) and the corresponding sample executable commands manually selected by operators (e.g., pairs of annotators).

[0024] In some embodiments, image processing system **100** will include components shown in FIG. **1**, including an executable command generation device **110**, a model training device **120**, a training database **140**, a database/repository **150**, an input device **160**, and a network **170** to facilitate communications among the various components. In some embodiments, image processing system **100** will optionally connect to an image editing device **130** to implement executable command **107** (e.g., perform operations on the image).

[0025] It is possible that image processing system **100** includes more or less components compared to those shown in FIG. **1**. For example, when the trainable machine learning model (e.g., machine learning model **105**) is pre-trained for generating executable command, image processing system

100 can omit the model training device. One or more of model training device **120**, executable command generation device **110**, image editing device **130**, and/or input device **160** can also be integrated into a single computing device. For example, input device **160** (e.g., a microphone) and image editing device **130** (e.g., processor executing image editing software) can be implemented in a single client computing device.

[0026] In some embodiments, image processing system **100** optionally includes network **170** to facilitate the communication among the various components of image processing system **100**, such as databases **140** and **150**, devices **110** and **120**, and input device **160**. For example, network **170** may be a local area network (LAN), a wireless network, a personal area network (PAN), metropolitan area network (MAN), a wide area network (WAN), etc. In some embodiments, wired data communication systems or devices can be used to implement network **170**.

[0027] In some embodiments, the various components of image processing system **100** are remote from each other or are in different locations and be connected through network **170** as shown in FIG. **1**. In some alternative embodiments, certain components of image processing system **100** are located on the same site or inside a same device. For example, training database **140** can be located on-site with or be part of model training device **120**. As another example, model training device **120** and executable command generation device **110** can be inside the same computer or processing device.

[0028] Image processing system **100** stores training data **101** including sample utterances, sample images to be operated on (e.g., relevant images), and the corresponding sample executable commands, and model input **103** including utterance **102** and images the user intend to operate on. For example, in some embodiments, training data **101** is stored in training database **140**, and the model input **103** is inputted to and stored in database/repository **150**.

[0029] In some embodiments, utterance **102** is received from input device **160** (e.g., input by the user) and transmitted to database/repository **150** and be part of model input **103**. In some embodiments, the image to be operated on (e.g., images related to utterance **102**) is received from a search engine (now shown) by searching the Internet, a local database, and/or is pre-selected by the user and is grouped with utterance **102** in database/repository **150** as part of model input **103**.

[0030] In some embodiments, images to be operated on in training data **101** are received in a manner similar to the image in model input **103**. The sample utterances, and the corresponding sample executable commends included in training data **101** are manually inputted and/or checked by operators and are stored in pairs in training data **101** along with the sample image to be operated on. For example, when collecting training data **101**, the operators (e.g., annotators) are divided into two groups, e.g., user-annotators and assistant-annotators. The user-annotators input sample utterances indicating intended requests (e.g., making requests throughout an operating session/conversation) and the assistant-annotators manually associate corresponding sample executable commends (e.g., associate the intended request) with the input utterances. In some embodiments, if the intended requests indicated by the user is not clear to the utterance assistant-annotators, a clarification request is made and is associated with the sample utterance to be part of training data **101**.

[0031] In some embodiments, machine learning model **105** (the structure of machine learning network model is described in detail in connection with FIG. **3**) includes an utterance sub-model, a visual sub-model, an attention network, and a selection gate. The utterance sub-model is configured to extract utterance features from the input utterance and generate an utterance feature vector based on the utterance features. The visual sub-model is configured to extract visual features and concept features from the image and generate a visual feature vector and a concept feature vector based on the visual features and the concept features, respectively. The attention network is configured to generate a first fused feature vector based on aligning the utterance feature vector and the visual feature vector, and to generate a second fused feature vector based on aligning the first fused feature vector and a current command feature vector. The selection gate is configured to generate a segment of the predicted executable command from weighted probabilities of each candidate token in a predetermined vocabulary, the segment being determined based on the second fused feature vector, the concept features, and the utterance features.

[0032] In some embodiments, the model training process is performed by model training device **120**. As used herein, "training" a learning model refers to determining one or more parameters of at least one layer in the learning model. For example, a convolutional layer of the visual sub-model (e.g., a Faster Region Based Convolutional Neural Networks (RCNN) network) in machine learning model **105** includes at least one filter or kernel. One or more parameters, such as kernel weights, size, shape, and structure, of the at least one filter is determined by e.g., an end-to-end training process. Consistent with some embodiments, machine learning model **105** can be trained based on any suitable methods such as supervised, semi-supervised, or non-supervised methods.

[0033] As show in FIG. **1**, executable command generation device **110** receives trained machine learning model **105** from model training device **120**. In some embodiments, executable command generation device **110** includes a processor and a non-transitory computer-readable medium (not shown). The processor performs instructions of an executable command generation process stored in the medium. In some embodiments, executable command generation device **110** additionally include input and output interfaces to communicate with database/repository **150**, input device **160**, network **170**, and/or a user interface of image editing device **130**. In some embodiments, the input interface is configured to receive model input **103** and/or initiate the executable command generation process, and the output interface is configured to provide the generated/predicted executable command **107**.

[0034] In some embodiments, model training device **120** communicates with training database **140** to receive one or more set of training data **101**. Model training device **120** uses training data **101** received from training database **140** to train a learning model, e.g., machine learning model **105** (the training process is described in detail in connection with FIG. **6**). In some embodiments, model training device **120** is implemented with hardware programmed by software that performs the training process. For example, model training device **120** includes a processor and a non-transitory computer-readable medium (not shown). The processor conducts the training by performing instructions of a training process stored in the computer-readable medium. In some embodiments, model training device **120** additionally includes input and output interfaces to communicate with training database **140**, network **170**, and/or a user interface (not shown). In some embodiments, the user interface is used for selecting sets of training data, adjusting one or more parameters of the training process, selecting or modifying a framework of the learning model, and/or manually or semi-automatically providing the corresponding sample executable command to the sample utterance.

[0035] In some embodiments, image processing system **100** optionally includes image editing device **130** for implementing executable command **107** generated/predicted by executable command generation device **110**. In some embodiments, image editing device **130** communicates with executable command generation device **110** to receive one or more executable command **107**. In some embodiments, image editing device **130** is implemented with hardware programmed by software (e.g., Adobe Photoshop™, Adobe Lightroom™, etc.) that performs image editing. For example, in some embodiments, image editing device **130** includes a processor and a non-transitory computer-readable medium (not shown). The processor implements executable command **107** by performing instructions of image editing stored in the computer-readable medium. In some embodiments, image editing device **130** additionally includes input and output interfaces to communicate with executable command generation device **110** and/or a user interface. For example, FIG. **8** shows an exemplary user interface, according to embodiments of the disclosure. In some embodiments, the user interface is used for manually or semi-automatically select or adjust executable command to improve implementation of the user's intention. In some embodiments, image editing device **130** is part of executable command generation device **110** or is part of the same computing device as executable command generation device **110**.

[0036] FIG. **2** illustrates a block diagram of an exemplary executable command generation device **110**, according to embodiments of the disclosure. As shown in FIG. **2**, executable command generation device **110** includes a communication interface **202**, a processor **204**, a memory **206**, and a storage **208**. In some embodiments, executable command generation device **110** has different modules implemented in a single device, such as an integrated circuit (IC) chip (e.g., implemented as an application-specific integrated circuit (ASIC) or a field-programmable gate array (FPGA)), or separate devices with dedicated functions. In some embodiments, one or more components of executable command generation device **110** is located in the cloud or is alternatively in a single location (e.g., inside a user equipment) or distributed locations. Consistent with the president disclosure, executable command generation device **110** is configured to generate/predicate executable commands based on model input **103** received from database/repository **150**.

[0037] In some embodiments, communication interface **202** sends data to and receive data from components such as database/repository **150**, input device **160**, model training device **120** and/or image editing device **130** via communication cables, a Wireless Local Area Network (WLAN), a Wide Area Network (WAN), wireless networks such as radio waves, a cellular network, and/or a local or short-range wireless network (e.g., Bluetooth™), or other communication methods. In some embodiments, communication interface **202** includes an integrated service digital network (ISDN) card, cable modem, satellite modem, or a modem to provide a data communication connection. In some other embodiments, communication interface **202** includes a local area network (LAN) card to provide a data communication connection to a compatible LAN. Wireless links can also be implemented by communication interface **202**. In such an implementation, communication interface **202** sends and receives electrical, electromagnetic or optical signals that carry digital data streams representing various types of information.

[0038] Consistent with some embodiments, communication interface **202** receives machine learning model **105** from model training device **120**, and model input **103** from database/repository **150**. In some embodiments, communication interface **202** further provides model input **103** and machine learning network **105** to memory **206** and/or storage **208** for storage or to processor **204** for processing.

[0039] In some embodiments, processor **204** includes any appropriate type of general-purpose or special-purpose microprocessor, digital signal processor, or microcontroller. In some embodiments, processor **204** is configured as a separate processor module dedicated to generate/predicate executable commands based on model input **103** using a learning model (e.g., machine learning model **105**). In some other embodiments, processor **204** is configured as a shared processor module for performing other functions in addition to executable commands generation/prediction.

[0040] In some embodiments, memory **206** and storage **208** includes any appropriate type of mass storage provided to store any type of information that processor **204** needs to operate. In some embodiments, memory **206** and storage **208** are any combination of volatile or non-volatile, magnetic, semiconductor, tape, optical, removable, non-removable, or other type of storage device or tangible (i.e., non-transitory) computer-readable medium including, but not limited to, a ROM, a flash memory, a dynamic RAM, and a static RAM. In some embodiments, memory **206** and/or storage **208** is configured to store one or more computer programs that is executed by processor **204** to perform functions disclosed herein. For example, memory **206** and/or storage **208** is configured to store program(s) that is executed by processor **204** to generate/predicate executable commands based on model input **103**.

[0041] In some embodiments, memory **206** and/or storage **208** are also configured to store intermediate data such as the utterance history (e.g., utterances previously entered by the user during the same session/conversation), the image history (e.g., images the user previously operated on), and the previously executed commands (e.g., executable commands previously generated/predicted by executable command generation device **110**) generated by executable command generation device **110**. In some embodiments, memory **206** and/or storage **208** additionally store various machine learning sub-models including their model parameters, such as word embedding models, Bidirectional Long short-term memory (BiLS™) models, Fast Region-based Convolutional Neural Network (RCNN) model, and softmax models that are used for executable command generation. The various types of data may be stored permanently, removed periodically, or disregarded immediately after the data is processed.

[0042] As shown in FIG. **2**, in some embodiments, processor **204** includes multiple modules, such as an utterance feature extraction unit **240**, a visual feature extraction unit **242**, a command feature extraction unit **244**, a feature fusion unit **246**, and a selection gate unit **248**, and the like. In some embodiments, these modules (and any corresponding sub-modules or sub-units) are hardware units (e.g., portions of an integrated circuit) of processor **204** designed for use with other components or are software units implemented by processor **204** through executing at least part of a program. In some embodiments, the program is stored on a computer-readable medium, and when executed by processor **204**, will perform one or more functions. Although FIG. **2** shows units **240-248** all within one processor **204**, it is contemplated that these units may be distributed among different processors located closely or remotely with each other.

[0043] In some embodiments, units **240-248** execute a computer program to apply machine learning model/network (e.g., machine learning model **105**) to automatically generate/predicate executable commands based on model input **103**. In some embodiments, when executing the program, the executable command will be generated in a segment-by-segment manner (e.g., one segment/token at a time). When generating a segment of the executable command, visual features (e.g., representing objects in the image) extracted from the image will first be aligned (e.g., using vector fusion) with utterance features extracted from the utterance to generate first fused features. The first fused features will then be aligned with current command features (e.g., features of executable command generated up to t th time step) generated based on previous command features (e.g., features of executable command generated up to (t−1)th time step) used for generating a previous segment of the predicted executable command to generate second fused features. A total token score (e.g., weighted probabilities) will be calculated for each candidate token in a predetermined vocabulary based on the second fused features, the current command features, the utterance features, and the concept features. The candidate token with the highest total token score will be selected as the segment of the executable command and will be combined with previously generated segments of the executable command to generate the executable command.

[0044] For example, FIG. **3** illustrates a schematic diagram of machine learning model **105** for generating/predicating executable commands based on model input **103**, according to embodiments of the disclosure. In some embodiments, machine learning model **105** includes an utterance sub-model **310**, a visual sub-model **320**, a command sub-model **330**, an attention network **340**, and a selection gate **350**. FIG. **3** will be described together with units **240-248** in FIG. **2**.

[0045] In some embodiments, units **240-248** of FIG. **2** execute computer instructions to perform the prediction/generation. For example, FIG. **4** illustrates a flowchart of an exemplary method **400** for generating/predicating executable commands, according to embodiments of the disclosure. In some embodiments, method **400** is implemented by executable command generation device **110** and particularly processor **204** or a separate processor not shown in FIG. **2**. In some embodiments, a computer-readable apparatus including a storage medium stores computer-readable and computer-executable instructions that are configured to, when executed by at least one processor apparatus, cause the at least one processor apparatus or another apparatus (e.g., the computerized apparatus) to perform the operations of the method **400**. Example components of the computerized apparatus are illustrated in FIG. **7**, which are described in more detail below.

[0046] In some embodiments, method **400** includes operations **402-416** as described below. It is to be appreciated that some of the operations may be performed simultaneously, or in a different order than shown in FIG. **4**. FIG. **4** will be described together with FIGS. **2** and **3**.

[0047] In operation **402**, communication interface **202** receives model input **103** from database/repository **150**. In some embodiments, model input **103** includes an image a user intends to operate on, and utterance(s) acquired by input device **160**, indicating the operation the user intended to perform. In some embodiments, the utterance history (e.g., utterances previously entered by the user during a same operation session/conversation/dialogue), the image history (e.g., images the user previously operated on), and/or previously executed commands (e.g., executable commands previously generated/predicted by executable command generation device **110** and/or manually selected by the user) along with model input **103** are fed to processor **204** (e.g., to different units in units **240-248** and the corresponding sub-models in machine learning network **105**) for generating/predicting executable commands.

[0048] In operation **404**, utterance feature extraction unit **240** extracts utterance features from a "dialogue" that includes the utterance included in model input **103**, the utterance history, and/or the previously executed commands to generate an utterance feature vector based on the utterance features. For example, when forming the "dialogue," the utterance and the utterance history are combined, and the previously executed commands are appended to the last utterance in the combination.

[0049] For example, as illustrated in FIG. **3**, the utterance is represented by an utterance feature vector U after being processed by utterance sub-model **310**. Specifically, in some embodiments, utterance sub-model **310** includes a bidirectional LSTM sub-network for encoding sequences of tokens (e.g., segments in the utterance) from the utterance W.sup.u to generate utterance features .Math.ϵR.sup.M×N×d according to equation (1):

[00001] $$\hat{U} = \mathrm{BiLSTM}(\mathrm{Embed}(W^u)) \quad (1)$$

where M denotes the dialogue length (e.g., the number of utterances/previously executed commands in the "dialogue"), N denotes the utterance length (e.g., the number of words in each utterance), and d denotes the number of feature dimensions. In some embodiments, utterance sub-model **310** also includes a LSTM sub-network where the last forward hidden state and the first backward hidden state of utterance features .Math. are extracted and concatenated, and are fed to the LSTM sub-network to generate the utterance feature vector U according to equation (2):

[00002] $$U = \mathrm{LSTM}([\hat{U}^f_{N-1}; \hat{U}^b_0]) \quad (2)$$

[0050] In operation **406**, visual feature extraction unit **242** extracts visual features from the image included in model input **103** to generate an image feature vector based on the image features extracted from the image. For example, as illustrated in FIG. **3**, the visual features {circumflex over (V)} representing objects in the image and the corresponding bounding box features B are extracted from the image by visual sub-model **320** and are combined to generate image features V. Specifically, in some embodiments, visual sub-model **320** includes a Faster RCNN sub-network for extracting the visual features {circumflex over (V)} and the corresponding bounding box features B from the image. In some embodiments, visual sub-model **320** also includes a linear layer for combining {circumflex over (V)} and B to generate image features vector V according to equation (3):

[00003] $$\hat{V}, B, W^c = \mathrm{FRCNN}(I), V = \mathrm{PE}(\hat{V}; B)) \quad (3)$$

where PE denotes positional encoding which is applied to the image (e.g., the same encoding value is applied to the visual feature from the same image).

[0051] In operation **408**, visual feature extraction unit **242** also extracts concept features from the image included in model input **103** to generate a concept feature vector based on the concept features extracted from the image. For example, as illustrated in FIG. **3**, the concept features WC (e.g., consists of tokens/text labels of the objects from the image I) are extracted from the image by visual sub-model **320** and are encoded to generate the concept feature vector C. In some embodiments, concept features extracted from the image includes object names and/or other attributes (e.g., size, color, location, etc.) and are in a textual semi-symbolic format (e.g., text labels).

[0052] Specifically, in some embodiments, visual sub-model **320** includes a word embedding layer and a bidirectional LSTM sub-network for encoding the concept features W.sup.e to generate the concept feature vector C according to equation (4):

[00004] $\hat{C} = (\mathrm{Embed}(W^c), C = \mathrm{PE}(\mathrm{BiLSTM}(\hat{C}))$ (4)

[0053] In operation **410**, command feature extraction unit **244** generates a current command feature vector (e.g., in t th time step) based on previous command feature vector(s) used for generating previously generated segment(s) of the executable command (e.g., from 1 stto (t−1)th time step). For example, as illustrated in FIG. **3**, command sub-model **330** includes an embedding layer and a LSTM layer. Tokens from previous segment(s) of the predicted executable command {w.sub.t}.sub.t=1.sup.T are embedded in the embedding layer and are fed to the LSTM layer sequentially to generate current command feature vector h.sub.t according to equation (5):

[00005] $\hat{w}_{t-1} = \mathrm{Embed}(w_{t-1}), h_t = \mathrm{LSTM}(\hat{w}_{t-1}, h_{t-1})$ (5)

[0054] In operations **412** and **414**, feature fusion unit **246** generate a first fused feature vector based on aligning the utterance feature vector U and the visual feature vector V, and generate a second fused feature vector based on aligning the first fused feature with the current command feature vector h.sub.t. For example, as illustrated in FIG. **3**, attention network **340** calculates a similarity matrix SϵR.sup.O×M between the utterance feature vector U and the visual feature vector V, where S.sub.ij=V.sub.i.sup.TU.sub.j and O is the total number of all features extracted from the images. In some embodiments, attention network **340** includes two structurally identical attention sub-networks **342** and **344** with different parameters (e.g., trainable parameters) for implementing the attention mechanism between the utterance feature vector U and the visual feature vector V, and between the first fused feature vector (e.g., fused based on the utterance feature vector U and the visual feature vector V) and current command feature vector h.sub.t respectively.

[0055] Specifically, attention sub-network **342** calculates fused utterance features Ū and fused visual features V (e.g., the first fused feature vector) based on the similarity matrix S according to equation (6):

[00006] $\bar{U} = \mathrm{softmax}(S)$ .Math. $U, \bar{V} = [V; \bar{U}; V \odot \bar{U}]$ .Math. W_v (6)

Where W.sub.vϵR.sup.3d×d is the trainable parameter, ⊚ denotes the element-wise product, · denotes matrix multiplication. Attention sub-network **344** calculates the second fused feature vector e.sub.t according to equation (7):

[00007] $e_i = \mathrm{Attn}(h_t, \bar{V})$ (7)

[0056] In operation **416**, selection gate unit **248** calculates a total token score (e.g., a weighted probability) for each candidate token/segment in the predetermined vocabulary (e.g., a vocabulary including all the candidate tokens that is possible to present in the executable commend). For example, FIG. **5** illustrates a flowchart of an exemplary method **500** for calculating a total token score for each candidate token in the predetermined vocabulary, according to embodiments of the disclosure. In some embodiments, method **500** is implemented by executable command generation device **110** and particularly selection gate unit **248** of processor **204** or a separate processor not shown in FIG. **2**. In some embodiments, method **500** includes operations **502-514** as described below.

[0057] In some embodiments, as illustrated in FIG. **3**, selection gate **350** includes a concept extractor **352**, a generator **354**, and an utterance extractor **356** for calculating a first token score a.sup.c.sub.t (e.g., a first probability), a second token score a.sup.g.sub.t (e.g., a second probability), and a third token score a.sup.u.sub.t (e.g., a third probability) based on the channels' own vocabulary (e.g., candidate tokens), and the corresponding adjusted weight (e.g., g.sub.t,0 for a.sup.g.sub.t, gt.sub.t,1 for a.sup.u.sub.t, and gt.sub.t,2 for a.sup.c.sub.t). For example, concept extractor **352** has a vocabulary that includes candidate tokens that are determined to be possible image concept features. For example, when defining concept extractor **352**'s vocabulary, a prediction is made ahead of time regarding all the tokens that could potentially become part of the image concept features (e.g., possible text labels associated with images, such as blue, apple, right top corner, etc.). Concept extractor **352** determines first token score act based on concept extractor **352**'s vocabulary and the second fused features (disclosed in detail below). Similarly, utterance extractor **356** has utterance extractor **356**'s vocabulary and determines third token score a.sup.u.sub.t based on the vocabulary and the current command features (disclosed in detail below). In some embodiments, generator **354** has the largest vocabulary that includes all candidate tokens that are possible in the "dialogue" (e.g., possible image concept features, possible utterance features, and/or command features), and generator **354** determines the second token score a.sup.g.sub.t based on generator **354**'s vocabulary and the second fused features (disclosed in detail below).

[0058] Accordingly, selection gate **350** calculates a total token score based on (e.g., a weighted sum of) the first token score act, the second token score a.sup.g.sub.t, and the third token score a.sup.u.sub.t, weighted by the corresponding adjusted weight (e.g., g.sub.t,0 for a.sup.g.sub.t, gt.sub.t,1 for a.sup.u.sub.t, and gt.sub.t,2 for a.sup.c.sub.t) for every candidate token in generator **354**'s vocabulary. The candidate token with the highest total token score will be selected as the token for the segment of the executable command. In some embodiments, the segment of the executable command will be combined with (e.g., appended to the end of) the existing (e.g., previously generated/predicted) segment(s) of the executable command. The segment of the executable command will be generated/predicted one at a time until the executable command is complete (e.g., a last segment of the executable command is generated).

[0059] Specifically, for a first candidate token in the generator **354**'s vocabulary, in operations **502**, selection gate unit **248** calculates the first token score a.sup.c.sub.t of the first candidate token. For example, concept extractor **352** (e.g., shown in in FIG. **3**) calculates the first token score act based on the concept feature vector C and the second fused feature vector e according to equation (8):

[00008] $(A_t^c)_i = e_t^\top C_i, \; {}^{c}_{t} = \mathrm{softmax}(A_t^c)$ (8)

[0060] Accordingly, in some embodiments, concept extractor **352** can directly obtain useful information from the concept features since the concept features provides object names/attributes in a textual semi-symbolic format.

[0061] In operation **504**, selection gate unit **248** calculates the second token score a.sup.g.sub.t, of the first candidate token. For example, generator **354** (e.g., shown in in FIG. **3**) calculates the second token score a.sup.g.sub.t based on the second fused feature vector e according to equation (9):

[00009] $l_t = \mathrm{Linear}(e_t), a_t^g = \mathrm{softmax}(l_t)$ (9)

[0062] In operation **506**, selection gate unit **248** calculates the third token score a.sup.u.sub.t of the first token. For example, utterance extractor **356** (e.g., shown in in FIG. **3**) calculates third token score a.sup.u.sub.t based on the current command feature vector h.sub.t and the utterance feature vector U according to equation (10):

[00010] $(A_t^u)_i = h_t^\top \bar{U}_i, a_t^u = \mathrm{softmax}(A_t^u)$ (10)

[0063] Since the utterance(s) (e.g., including the utterance history) contain direct clues for generating segments of the executable commands, in some embodiments, utterance extractor **356** can benefit directly from extracting keywords from the context of the utterance(s).

[0064] In operation **508**, selection gate unit **248** calculates the adaptive weights gtϵR.sup.1×3 for each of the first the second and the third token scores (e.g., gt includes the value of g.sub.t,0, g.sub.t,1, and g.sub.t,2, where g.sub.t,0 corresponds to a.sup.g.sub.t, gt.sub.t,1 corresponds to a.sup.u.sub.t, and gt.sub.t,2 corresponds to a.sup.c.sub.t). For example, selection gate **350** (e.g., shown in in FIG. **3**) calculates the adaptive weights gt based on the second fused feature vector e according to equation (11):

[00011] $g_t = \text{softmax}(W_g^{\top} e_i) \quad (11)$

where W.sub.g is the trainable parameter.

[0065] In operation **510**, selection gate unit **248** calculates the total token score p(W.sub.t|W.sub.1:t−1, I,D) (e.g., the weighted sum of each of the first, the second, and the third token scores) of the first candidate token. For example, selection gate **350** (e.g., shown in in FIG. **3**) calculates the total token score p(W.sub.t|W.sub.1:t−1, I,D) according to equation (11):

[00012]

$$p(w_t \text{ .Math. } w_{1:t-1}, I, D) \text{ .Math. } = g_{t,0} \text{ .Math. } a_t^g + g_{t,1} \text{ .Math. } a_t^u + g_{t,2} \text{ .Math. } a_t^c \quad (11)$$

[0066] In some embodiments, the calculation of the total token score is optimized by minimizing a loss. The loss is defined as:

[00013] $L = - \text{.Math.}_{t=1}^{T} \log p(w_t^* \text{ .Math. } w_{0:t-1}, \text{ .Math. } I, D) \quad (12)$

[0067] In operation **512**, selection gate unit **248** determines whether to calculate a total token score for another candidate token in the generator **354**'s vocabulary. If yes, method **500** proceeds back to operation **502** to determine a first token score for the another candidate token. If no (e.g., all candidate tokens have a total token score determined using method **500**), method continues to operation **514**.

[0068] In operation **514**, selection gate unit **248** selects the candidate token with the highest total token score to be the segment of the executable command. In some embodiments, as illustrated in FIG. **3**, the segment of the executable command will be combined with the existing/previously generated/predicted segment(s) of the executable command. Method **500** will generating new segment of the executable command until the executable command is complete (e.g., a last segment of the executable command is generated).

[0069] FIG. **9** shows exemplary selection gate **350** outputs, according to some embodiments of the disclosure. As illustrated in FIG. **9**, each predicted segments in a first executable command **901** (e.g., segments **910**, **920**, and **930**) and in a second executable command **902** (e.g., segments **940** and **950**) has a first, a second, and a third token score calculated by one of concept extractor **352**, generator **354**, and utterance extractor **356** of selection gate **350** (e.g., shown in in FIG. **3**). Take segment **910** as an example, first token score **914**, second token score **916**, and third token score **912** are generated by concept extractor **352**, generator **354**, and utterance extractor **356** based on method **500** disclosed above. Specifically, since the token “search” is only in generator **354**'s vocabulary (e.g., possible for potential command features) and not in concept extractor **352**'s and utterance extractor **356**'s vocabularies, first token score **914** and second token score **916** are both determined as 0, and third token score **912** is calculated as 1 according to equation (9) shown above.

[0070] FIG. **6**. illustrates a schematic diagram of an exemplary method **600** for learning model (e.g., machine learning network **105**) training, according to embodiments of the disclosure. Consistent with some embodiments as shown in FIG. **6**, model training device **120** uses training data **101** as an input for model training. In some embodiments, training data **101** includes sample utterances **602**, sample image(s) **604**, and corresponding sample executable command(s) **606** that are grouped together. When performing training, operators (e.g., annotators) manually associate the sample executable commends with the sample utterance and the sample image. For example, in some embodiments, annotators are split in two groups, e.g., the user-annotators and the assistance-annotators. In some embodiments, user-annotators will give request (e.g., intended operations) by making utterances (e.g., make four or more requests throughout a conversation). Assistant-annotators will perform the operations intended by the user-annotators' utterance (e.g., select the corresponding sample executable commend). For example, FIGS. **10**A and **10**B show exemplary data collection interface for user-annotators and assistant-annotators respectively, according to according to embodiments of the disclosure.

[0071] Based on sample utterances **602**, sample image(s) **604**, and corresponding sample executable command(s) **606**, model training device **120** determines/trains one or more parameters in at least one sub-network/model in machine learning network **105** (e.g., trainable parameters of W.sub.v and W.sub.g). For example, a convolutional layer in machine learning network **105** may include at least one filter or kernel. One or more parameters, such as kernel weights, size, shape, and structure, of the at least one filter may be determined by e.g., an end-to-end manner, or a backpropagation-based training process using training data **101** that includes grouped sample utterances **602**, sample image(s) **604**, and corresponding sample executable command(s) **606**. Consistent with some embodiments, machine learning network **105** may be trained using supervised, non-supervised, or semi-supervised method. Using method **600**, model training device **120** generates a trained learning model (e.g., machine learning network **105**) as an output. Executable command generation device **110** can then use the trained learning model for executable command generation/prediction.

[0072] In some embodiments, a computer-readable apparatus including a storage medium stores computer-readable and computer-executable instructions that are configured to, when executed by at least one processor apparatus, cause the at least one processor apparatus or another apparatus (e.g., the computerized apparatus) to perform the operations/operations of the method **600**. Example components of the computerized apparatus are illustrated in FIG. **7**, which are described in more detail below.

[0073] FIG. **7** shows a schematic diagram of components of a computing device **700** that is implemented in a computing system in accordance with some implementations. As illustrated, computing device **700** includes a bus **712** that directly or indirectly couples one or more processors(s) **702**, a memory subsystem **704**, a communication interface **706**, an input/output (I/O) interface **708**, and/or one or more user interface components **710**. It should be noted that, in some embodiments, various other components are included in a computing device that are not shown in FIG. **7**, and/or one or more components shown in FIG. **7** are omitted.

[0074] In some embodiments, computing device **700** includes or is coupled to a memory subsystem **704**. Memory subsystem **704** includes a computer-readable medium (e.g., non-transitory storage medium) or a combination of computer-readable media. Examples of computer-readable media include optical media (e.g., compact discs, digital video discs, or the like), magnetic media (e.g., hard disks, floppy disks, or the like), semiconductor media (e.g., flash memory, dynamic random access memory (DRAM), static random access memory (SRAM), electrically programmable read-only memory (EPROM), electrically erasable programmable read-only memory (EEPROM), or the like), or a combination thereof. In some embodiments, the computer-readable media includes non-volatile memory, volatile memory, or a combination thereof. In some embodiments, memory subsystem **704** also includes one or more hardware devices such as a solid-state memory, one or more hard drives, one or more optical disk drives, or the like. In some embodiments, memory subsystem **704** stores content files such as text-based files, audio files, image files, and/or video files, etc. In some implementations, the content files include documents, pictures, photos, songs, podcasts, movies, etc. In some embodiments, memory subsystem **704** stores one or more computer program products that are each implemented as a set of instructions (e.g., program code) stored on a computer-readable medium.

[0075] A computer program product (e.g., a program stored in or downloadable onto a computer readable medium) includes instructions or program code that are executable by one or more processors (e.g., processor(s) **702**, or processor(s) of another computing device communicatively coupled to computing device **700**) to perform various operations or functions such as those described with reference to FIGS. **2-6**. In some embodiments, a computer program product is referred to as a non-transitory computer readable medium storing or comprising instructions to perform certain operations or functions. Examples of a computer program product include firmware, software driver, operating system, or software application. Examples of a software application include data management application (e.g., file management application, document management application, media management application, database application, etc.), communication application (e.g., email application, messaging application, teleconference or meeting application, social media application, etc.), productivity application (e.g., document viewer application, document creation or editing application, etc.), media or interactive application (e.g., web browser, image or photo viewer, audio or video playback application, gaming application, virtual or augmented reality application, shopping application, recommendation or review application, etc.), creativity application (e.g., image, drawing, photo, audio, or video creation or editing application, web page development application, virtual or augmented reality creation or editing application, graphic design application, etc.), or the like.

[0076] In some embodiments, a computer program product such as any of the example software application are implemented using one or more neural network or machine learning models. In such embodiments, one or more neural network or matching learning models are trained using computing device **700** (or a computing system that includes computing device **700**). Furthermore, in some implementations, computing device **700** (or a computing system include computing device **700**) executes the one or more neural network or machine learning models as part of the computer program product to perform inference operations. It should be noted, in some embodiments, the neural network or matching learning model(s) are trained using a computing device or system that is the same as, overlaps with, or is separate from the computing device or system performing inference operations.

[0077] Communication interface **706** is used by computing device **700** to communicate with one or more communication networks, and/or other electronic device(s). Example types of communication networks include wired communication networks and/or wireless communication networks. Example types of communication networks include the Internet, a wide-area network, a local-area network, a virtual private network (VPN), an Intranet, or the like. In some embodiments, communication interface **706** utilizes various drivers, wireless communication circuitry, network interface circuitry, or the like to enable communication via various communication networks.

[0078] I/O interface **708** includes various drivers and/or hardware circuitry for receiving input from various input devices, providing output to various output devices, or exchanging input/output with various input/output devices. Examples of devices coupled to I/O interface **708** include peripheral devices such as a printer, a docking station, a communication hub, a charging device, etc. In some implementations, some devices coupled to I/O interface **708** are used as user interface component(s) **710**. In one example, a user operates input elements of user interface component(s) **710** to invoke the functionality of computing device **700** and/or of another device communicatively coupled to computing device **700**; a user views, hears, and/or otherwise experiences output from computing device **700** via output elements of user interface component(s) **710**. Some user interface component(s) **710** provide both input and output functionalities. Examples of input user interface component include a mouse, a joystick, a keyboard, a microphone, a camera, or the like. Examples of output user interface component include a display screen (e.g., a monitor, an LCD display, etc.), one or more speakers, or the like. Examples of a user interface components provide both input and output functionalities include a touchscreen, haptic feedback controllers, or the like.

[0079] Various embodiments are described herein which are intended to be illustrative. Alternative embodiments may be apparent to those of ordinary skill in the art without departing from the scope of the disclosure. In one example, one or more features from one embodiment are combined with another embodiment to form an alternative embodiment. In another example, one or more features

are omitted from an embodiment to form an alternative embodiment without departing from the scope of the disclosure. Additionally, it should be noted that, in some implementations, certain features described herein are utilized without reference to other features described herein.
[0080] With reference to the various processes described above, it should be understood that the order in which operations are performed is not limited to the order described herein. Moreover, in some embodiments, two or more operations are performed concurrently and/or substantially in parallel. In some embodiments, what is described as a single operation is split into two or more operations (e.g., performed by the same device, performed by two or more different devices, etc.). In some embodiments, what is described as multiple operations is combined into a single (e.g., performed by the same device, etc.). Descriptions of various blocks, modules, or components as distinct should not be construed as requiring that the blocks, modules, or components be separate (e.g., physically separate) and/or perform separate operations. For example, in some implementations, two or more blocks, modules, and/or components are merged. As another example, a single block, module, and/or components is split into multiple blocks, modules, and/or components.
[0081] The phrases “in one embodiment,” “in an embodiment,” “in one example,” and “in an example” are used herein. It should be understood that, in some cases, these phrases refer to the same embodiments and/or examples, and, in other cases, these phrases refer to different embodiments and/or examples. The terms “comprising,” “having,” and “including” should be understood to be synonymous unless indicated otherwise. The phases “A and/or B” and “A or B” should be understood to mean {A}, {B}, or {A, B}. The phrase “at least one of A, B, or C” and “at least one of A, B, and C” should each be understood to mean {A}, {B}, {C}, {A, B}, {A, C}, {B, C}, or {A, B, C}.

Claims

1. A computer-implemented method for predicting and using an executable command, the computer-implemented method comprising: receiving, via a user interface, an utterance indicating a request associated with a content; predicting, from the utterance, the executable command comprising one or more segments by, for each candidate token of each vocabulary: determining a total token score based on a combination of one or more token scores calculated by respective one or more machine learning models; and selecting, from the vocabulary, a candidate token having the highest total token score to be a segment of the executable command; and based on the one or more segments of the executable command, enabling performance of one or more operations with respect to the content, and providing the content to the user interface.
2. The computer-implemented method of claim 1, wherein: the request comprises an image search request; and the one or more operations comprise providing a search result of an image based on the image search request.
3. The computer-implemented method of claim 1, wherein: the content comprises an input image; the request comprises an image search request or an image editing request; and the one or more operations comprise retrieving one or more images based on characteristics of the input image, editing the input image or the one or more retrieved images, or a combination thereof.
4. The computer-implemented method of claim 1, wherein: the one or more machine learning models comprise a first machine learning model, a second machine learning model, and a third machine learning model; the one or more token scores comprise a first token score generated by the first machine learning model, a second token score generated by the second machine learning model, and a third token score generated by the third machine learning model; and the total token score comprises a weighted sum of the first token score, the second token score, and the third token score.
5. The computer-implemented method of claim 4, wherein: the first machine learning model

comprises a first vocabulary that includes candidate tokens corresponding to possible image concept features; the second machine learning model comprises a second vocabulary that includes candidate tokens corresponding to possible command features; and the third machine learning model comprises a third vocabulary that includes the image concept features and the possible command features.

6. The computer-implemented method of claim 4, wherein: the content comprises an input image; the first token score is generated by the first machine learning model based on a concept feature vector associated with the input image, and a fused feature vector derived from a command feature vector and visual features of the input image; the second token score is generated by the second machine learning model based on the fused feature vector; and the third token score is generated by the third machine learning model based on the command feature vector, and an utterance feature vector derived from utterance features extracted from the utterance.

7. The computer-implemented method of claim 1, wherein the one or more segments of the executable command comprise one or more words semantically related to a plurality of words within the utterance indicating the request.

8. The computer-implemented method of claim 1, wherein: each vocabulary comprises a plurality of candidate tokens that are usable in the executable command; and the one or more segments of the executable command comprise at least a first segment having the highest total token score from a first vocabulary, and a second segment selected having the highest total token score from a second vocabulary.

9. A command generation device comprising: a user interface; one or more machine learning models; at least one processor; and at least one non-transitory computer-readable apparatus comprising a storage medium, the storage medium comprising a plurality of instructions configured to, when executed by the at least one processor, cause the command generation device to: receive, via the user interface, an utterance indicating a request associated with a content; predict, from the utterance, the executable command comprising one or more segments by, for each candidate token of each vocabulary: determining a total token score based on a combination of one or more token scores calculated by the one or more machine learning models; and selecting, from the vocabulary, a candidate token having the highest total token score to be a segment of the executable command; and based on the one or more segments of the executable command, enable performance of one or more operations with respect to the content, and provide the content to the user interface.

10. The command generation device of claim 9, wherein: the request comprises an image search request; and the one or more operations comprise providing, to the command generation device, a search result of an image based on the image search request.

11. The command generation device of claim 9, wherein: the content comprises an input image; the request comprises an image search request or an image editing request; and the enabling performance of the one or more operations comprises: enabling retrieval of one or more images based on characteristics of the input image, editing the input image or the one or more retrieved images, or a combination thereof; and enabling provision of at least one image to the command generation device subsequent to the retrieval, the edit, or both.

12. The command generation device of claim 9, wherein: the one or more machine learning models comprise a first machine learning model, a second machine learning model, and a third machine learning model; and the one or more token scores comprise a first token score generated by the first machine learning model, a second token score generated by the second machine learning model, and a third token score generated by the third machine learning model.

13. The command generation device of claim 12, wherein: the first machine learning model comprises a first vocabulary that includes candidate tokens corresponding to possible image concept features; the second machine learning model comprises a second vocabulary that includes candidate tokens corresponding to possible command features; and the third machine learning model comprises a third vocabulary that includes the image concept features and the possible command features.

14. The command generation device of claim 12, wherein: the content comprises an input image; the first token score is generated by the first machine learning model based on a concept feature vector associated with the input image, and a fused feature vector derived from a command feature vector and visual features of the input image; the second token score is generated by the second machine learning model based on the fused feature vector; and the third token score is generated by the third machine learning model based on the command feature vector, and an utterance feature vector derived from utterance features extracted from the utterance.

15. A non-transitory computer-readable apparatus comprising a storage medium, the storage medium comprising a plurality of instructions configured to, when executed by at least one processor, cause a device to: receive, via a user interface, an utterance indicating a request associated with a content; predict, from the utterance, the executable command comprising one or more segments by, for each candidate token of each vocabulary: determining a total token score based on a combination of one or more token scores calculated by respective one or more machine learning models; and selecting, from the vocabulary, a candidate token having the highest total token score to be a segment of the executable command; and based on the one or more segments of the executable command, enable performance of one or more operations with respect to the content, and provide the content to the user interface.

16. The non-transitory computer-readable apparatus of claim 15, wherein: the request comprises an image search request; and the one or more operations comprise providing, to the device, a search result of an image based on the image search request.

17. The non-transitory computer-readable apparatus of claim 15, wherein: the content comprises an input image; the request comprises an image search request or an image editing request; and the enabling performance of the one or more operations comprises: enabling retrieval of one or more images based on characteristics of the input image, editing the input image or the one or more retrieved images, or a combination thereof; and enabling provision of at least one image to the device subsequent to the retrieval, the edit, or both.

18. The non-transitory computer-readable apparatus of claim 15, wherein: the one or more machine learning models comprise a first machine learning model, a second machine learning model, and a third machine learning model; and the one or more token scores comprise a first token score generated by the first machine learning model, a second token score generated by the second machine learning model, and a third token score generated by the third machine learning model.

19. The non-transitory computer-readable apparatus of claim 18, wherein: the first machine learning model comprises a first vocabulary that includes candidate tokens corresponding to possible image concept features; the second machine learning model comprises a second vocabulary that includes candidate tokens corresponding to possible command features; and the third machine learning model comprises a third vocabulary that includes the image concept features and the possible command features.

20. The non-transitory computer-readable apparatus of claim 18, wherein: the content comprises an input image; the first token score is generated by the first machine learning model based on a concept feature vector associated with the input image, and a fused feature vector derived from a command feature vector and visual features of the input image; the second token score is generated by the second machine learning model based on the fused feature vector; and the third token score is generated by the third machine learning model based on the command feature vector, and an utterance feature vector derived from utterance features extracted from the utterance.